ALMIR DE JESUS COSTA

UMA PROPOSTA DE ESPECIALIZAÇÃO DE ÁREAS DE PROCESSO DO CMMI-SE/SW EVIDENCIANDO AS PARTICULARIDADES DO DESENVOLVIMENTO DE SOFTWARE BASEADO EM COMPONENTES

Monografia de graduação apresentada ao Departamento de Ciência da Computação da Universidade Federal de Lavras como parte das exigências do Curso de Ciência da Computação para obtenção do título de Bacharel em Ciência da Computação.

LAVRAS
MINAS GERAIS – BRASIL
2007

ALMIR DE JESUS COSTA

UMA PROPOSTA DE ESPECIALIZAÇÃO DE ÁREAS DE PROCESSO DO CMMI-SE/SW EVIDENCIANDO AS PARTICULARIDADES DO DESENVOLVIMENTO BASEADO EM COMPONENTES

Monografia de graduação apresentada ao Departamento de Ciência da Computação da Universidade Federal de Lavras como parte das exigências do Curso de Ciência da Computação para obtenção do título de Bacharel em Ciência da Computação.

Área de concentração:

Engenharia e Qualidade de Software

Orientador:

Prof. Guilherme Bastos Alvarenga

LAVRAS
MINAS GERAIS – BRASIL
2007

Ficha Catalográfica preparada pela Divisão de Processo Técnico da Biblioteca Central da UFLA

Costa, Almir de Jesus

Uma Proposta de Especialização de Áreas de Processo do CMMI-SE/SW Evidenciando Particularidades do Desenvolvimento Baseado em Componentes./ Almir de Jesus Costa – Minas Gerais, 2007.

Monografia de Graduação – Universidade Federal de Lavras. Departamento de Ciência da Computação.

1. Engenharia de Software. 2. Qualidade de Software. 3. Desenvolvimento baseado em componentes 4. CMMI. I. COSTA, A. de J. II. Universidade Federal de Lavras. III. Título.

ALMIR DE JESUS COSTA

UMA PROPOSTA DE ESPECIALIZAÇÃO DE ÁREAS DE PROCESSO DO CMMI-SE/SW EVIDENCIANDO AS PARTICULARIDADES DO DESENVOLVIMENTO BASEADO EM COMPONENTES

Monografia de graduação apresentada ao Departamento de Ciência da Computação da Universidade Federal de lavras como parte das exigências do Curso de Ciência da Computação para obtenção do título de Bacharel em Ciência da Computação.

Aprovada em 23 de Março de 2007.

_____
Prof. André Luiz Zambalde

_____
Prof. Marluce Rodrigues Pereira

_____
Prof. Guilherme Bastos Alvarenga
(Orientador)

LAVRAS
MINAS GERAIS - BRASIL

Dedico este trabalho a querida vovó Clara que nos deixou.

Agradecimentos

Agradeço,

Aos meus pais por todo amor, sem eles não chegaria até aqui.

Aos grandes amigos que colaboraram para realização deste trabalho: Leandro Silva, Vanessa Maira e Rafael Vargas.

As titias Elena e Neuza.

As famílias Swfactory e Bokts.

Aos amigos de república: Bagas, Dema, Breno e Daniel Masseli.

A todos os novos irmãos que ganhei nesta universidade.

Ao meu orientador, o professor Guilherme Bastos Alvarenga por toda confiança deixa em mim.

A professora Marluce Rodrigues e professor André Luiz Zambalde por participarem da avaliação deste trabalho.

UMA PROPOSTA DE ESPECIALIZAÇÃO DE ÁREAS DE PROCESSO DO CMMI-SE/SW EVIDENCIANDO AS PARTICULARIDADES DO DESENVOLVIMENTO BASEADO EM COMPONENTES

RESUMO

O presente trabalho apresenta uma proposta de adequação do modelo de maturidade de capacidade CMMI-SE/SW v1.1, em sua versão estagiada, para o contexto específico da melhoria de processo de desenvolvimento de software baseado em componentes. Procurou-se explorar as áreas de processo Desenvolvimento de Requisitos, Solução Técnica e Integração de Produto pertencentes à área de engenharia do modelo e que se relacionaram com o reuso quando tomadas para essa perspectiva. A estratégia usada buscou especializar as três áreas de processo, anteriormente referidas, através da análise de dois modelos de processo de desenvolvimento, o UML *Components* e o Processo de DBC da Unicamp, junto a um modelo de maturidade de capacidade de processos, o FAA iCMM. Explorou-se práticas relacionadas à diretrizes e especificação de processos abordadas nestes modelos.

Palavras-Chave: Engenharia de Software, Qualidade de software, Desenvolvimento baseado em componentes, CMMI, UML *Components*, Processo de DBC da Unicamp e FAA iCMM.

A PROPOSAL OF SPECIALIZATION OF AREAS OF PROCESS OF THE CMMI-SE/SW EVIDENCING THE PARTICULARITITIES OF THE DEVELOPMENT OF SOFTWARE BASED ON COMPONENTS

ABSTRACT

The present work presents a proposal of adequacy of the model of capacity of maturity CMMI-SE/SW v1.1, in its served as apprentice version, for the specific context of the improvement of process of development of software based on components. It was looked to explore the process areas Development of Requirements, Solution Technique and Integration of Product pertaining the area of engineering of the model and that they had become related with reuse when taken for this perspective. The used strategy searched to specialize the three areas of process, previously related, through the analysis of two models of process of development, the UML Components and the Process of DBC of the Unicamp, together a model of capacity of maturity of processes, FAA iCMM. It was explored practical related the guidelines and boarded specification of processes in these models.

Key-Words: Software Engineering, Software Quality, Component Based Development, CMMI, UML Components, Process of DBC of the Unicamp and FAA iCMM.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS

AB - Habilitação

ABNT – Associação Brasileira de Normas Técnicas

BEA – *Bureau of Economic Analysis*

CBSE – *Component Based Software Engineering* (Engenharia de Software Baseada em Componentes)

CMM – *Capability Maturity Model (*Modelo de Maturidade da Capacidade)

CMMI-SE/SW - *Capability Maturity Model Integration for Systems Engineering and Software Engineering (*Modelo de Maturidade de Capacidade Integrado para Engenharia de Sistemas e Engenharia de Software)

COMPGOV – Componentes para Governo Eletrônico

CORBA - *Common Object Request Broker Architecture*

DBC – Desenvolvimento Baseado em Componentes

DI – Implementação

EIA - *Electronic Industries Alliance* (Aliança das Indústrias Eletrônicas)

FINEP – Financiadora de Estudos e Projetos

IBM - *International Business Machines*

IEC - *International Electrotechnical Commission* (Comissão Internacional Eletrotécnica)

IEEE - *Institute of Electrical and Electronics Engineers* – Instituto de Engenheiros Elétricos e Eletrônicos

ISO - *International Organization for Standardization (*Organização Internacional para Padronização)

MCT – Ministério de Ciência e Tecnologia

NBR – Norma Brasileira

NQI - *National Quality Institute*

PMI – *Project Management Institute* (Instituto de Gerenciamento de Projetos)

PRO2PI - *Process Capability Profile to Process Improvement* (Perfil de Capacidade de Processo para Melhoria de Processo).

PRO2PI-CYCLE – Ciclo de Melhoria com PRO2PI

REU - Reuso

SEI - *Software Engineering Institute* (Instituto de Engenharia de Software)

SPICE - *Software Process Improvement and Capability dEtermination* (Melhoria de Processo de Software e Determinação da Capacidade)

SUP - Suporte

SW-CMM – *Capability Maturity Model for Software* (Modelo de Maturidade de Capacidade para Software)

UML – *Unified Modeling Language* – (Linguagem de Modelagem Unificada)

1. INTRODUÇÃO

Neste primeiro capítulo é apresentada a introdução ao trabalho proposto, juntamente com as motivações que levaram a sua realização, os objetivos e as justificativas.

1.1. *Contextualização e Motivações*

Cada vez mais o desenvolvimento de software está sujeito a fortes restrições de prazos e custos com exigência de garantia de alta qualidade. Em busca do cumprimento de objetivos tão antagônicos, o desenvolvimento baseado em componentes vem surgindo como opção para redução de custos e prazos por proporcionar a reutilização de código (Brito *et al.* 2005).

Um componente de software é um programa de computador com função claramente definida, modularizado, integrável com outros a partir de suas interfaces bem definidas (MCT, 2005).

Segundo Rossi (2004), as organizações atualmente estão cada vez mais dependentes de software para realização de seus negócios. Com isto, uma das preocupações, na área de desenvolvimento de software, é a obtenção cada vez mais rápida de softwares que atendam às necessidades atuais e que sejam flexíveis para acompanhar as mudanças de tecnologia e práticas de negócio. A reutilização de componentes de software tem sido considerada uma das formas para obter redução dos custos e do tempo de desenvolvimento e aumento da produtividade e da qualidade do produto de software.

Com a popularização do Desenvolvimento Baseado em Componentes - DBC, a necessidade de novos processos voltados para esse paradigma é uma realidade. Isso acontece porque os processos de desenvolvimento tradicionais não são totalmente adequados ao desenvolvimento de software baseados em componentes. Mais especificamente, esses processos devem conter fases e métodos que ofereçam entre outras coisas, técnicas que permitam o empacotamento de componentes com o objetivo específico de serem reutilizados (Brito *et al.* 2005).

Uma potencialidade do uso de componentes é a criação de um novo nicho de mercado: o mercado de componentes. Este mercado origina-se na interação de empresas especializadas no desenvolvimento, manutenção e atualização de componentes com seus clientes imediatos que, na maior parte dos casos, também são desenvolvedores de software.

Os clientes têm a opção de adquirir componentes de diversos fornecedores, desde que sigam um mesmo modelo de componentes, um mesmo contrato e uma mesma interface.

No entanto, para que este mercado se estabeleça de forma definitiva é necessário que produtores e consumidores de componentes alcancem uma massa crítica de componentes disponíveis. Uma vez atingida a massa crítica, isto é, um patamar de qualidade e quantidade de componentes de software aplicáveis em determinado segmento, seu uso naquele segmento é inevitável (Szyperski, 2002).

De acordo com o Ministério de Ciência e Tecnologia – MCT (2005), um produto que utiliza componentes se beneficia da produtividade e inovação de todos os fornecedores de componentes. Uma empresa de software que não estiver preparada para utilizar componentes nesse possível cenário correrá sérios riscos, uma vez que a Engenharia de Software Baseada em Componentes - ESBC efetivamente se estabelecer no mercado. Uma estimativas realizadas para o Departamento de Defesa Norte Americano, (Boehm, 1999) constatou um ganho de 47% de produtividade com a utilização do reuso. Na mesma pesquisa foi constatado que trabalhando de forma organizada é possível obter um ganho de 17% de produtividade, logo, utilizando-se de processo junto a reuso é obtido uma produtividade considerável.

A implementação de programas de melhoria de processo tem se mostrado um elemento crucial para o sucesso das organizações desenvolvedoras de software.

Por outro lado, dentre os mecanismos disponíveis e capazes de contribuir com a melhoria de processos, destaca-se no mercado atual o CMMI-SE/SW (*Capability Maturity Model Integrated for Systems Engineering and Software Engineering),* cujo propósito consiste em estabelecer um guia para a melhoria de processos da organização e sua capacidade para gerenciar o desenvolvimento, aquisição ou manutenção de produtos ou serviços. Entretanto, o modelo atualmente se mostra deficitário no que concerne a capacitação de processo para o DBC uma vez que não se faz referência a uma política de reuso.

1.2. *Objetivos e Justificativas*

Segundo Silva (2006), o modelo de maturidade CMMI-SE/SW é aplicado a projetos de desenvolvimento de software, mas não contempla o desenvolvimento baseado em componentes. Ele possui algumas áreas de processo relacionadas ao desenvolvimento

baseado em componentes, mas não é o suficiente para estabelecer diretrizes relacionadas ao reuso de componentes.

Nesse sentido, o objetivo deste trabalho é adequar o modelo CMMI, através de uma proposta, de tal maneira que dê suporte aos processos da ESBC através da especialização das áreas de processo:

- Desenvolvimento de Requisitos;
- Solução Técnica, e;
- Integração de Produto.

Estas três áreas de processos foram escolhidas por pertencerem às áreas de processo de engenharia do CMMI.

É pretendido nortear os processos de desenvolvimento de software para grupo de desenvolvedores que utilizam o reuso de componentes, através das áreas de processo reformuladas.

1.3. *Organização do Trabalho*

O trabalho está estruturado em cinco capítulos, a saber:

- Capítulo 1: apresenta uma introdução do trabalho realizado, descrevendo o contexto e as motivações no qual o mesmo se encontra e também os objetivos deste trabalho;
- Capítulo 2: discute os conceitos de engenharia e qualidade, normas e modelos de qualidade de software e desenvolvimento baseado em componentes;
- Capítulo 3: mostra a metodologia utilizada para concepção deste trabalho;
- Capítulo 4: são apresentados os resultados e discussões acerca do trabalho realizado;
- Capítulo 5: são discutidas as conclusões deste trabalho.

2. REFERENCIAL TEÓRICO

Este capítulo apresenta conceitos para o melhor entendimento do trabalho desenvolvido. Nas seções seguintes são apresentados os conceitos de engenharia e qualidade de software, os conceitos de DBC e a abordagem PRO2PI (*Process Capability Profile to Process Improvement*) para Melhoria de Processo.

2.1. *Engenharia e Qualidade de Software*

Segundo Vasconcelos (2003), há alguns anos, desenvolvia-se software de uma maneira artesanal. A partir de uma simples definição dos requisitos do software, partia-se para a implementação do mesmo. Hoje em dia, ainda há muitas empresas que desenvolvem software dessa maneira, mas várias outras estão mudando suas formas de trabalho.

Sendo assim surgiu a proposta que o desenvolvimento de software deixasse de ser puramente artesanal e passasse a ser baseado em princípios de Engenharia, ou seja, seguindo um enfoque estruturado e metódico (Vasconcelos, 2003).

Portanto, o termo Engenharia de Software, que se refere ao desenvolvimento de software por grupos de pessoas, usando princípios de engenharia e englobando aspectos técnicos e não-técnicos, de modo a produzir software de qualidade, de forma eficaz e dentro de custos aceitáveis, passou a ser utilizado no mundo da informática, afirma Vasconcelos (2003).

A Sociedade de Computação IEEE - *Institute of Electral and Eletronic Engineers* (IEEE, 1990) define engenharia de software como a aplicação de uma abordagem quantitativa, sistemática e disciplinada ao desenvolvimento, operação e manutenção de software; que é a aplicação da engenharia de software.

De acordo com Vasconcelos (2003), as organizações de software, para se tornarem mais competitivas, vêm investindo cada vez mais na qualidade de seus produtos e serviços de software.

Souza (2004), diz que para alcançar a qualidade, a engenharia de software utiliza-se de melhoria de processos, implementada através de modelos abstratos ou formais que permitem aos engenheiros especificar, projetar, implementar e manter sistemas de software, avaliando e garantindo suas qualidades. Além disso, a engenharia de software deve oferecer mecanismos para planejar e gerenciar o processo de desenvolvimento.

2.1.1. Qualidade do Processo de Software

O conceito de processo é quase que intuitivo. As engenharias comumente descrevem processos como sendo diversas operações pelas quais passa um produto até ele ficar pronto, afirma Souza (2004).

Segundo a Associação Brasileira de Normas Técnicas – ABNT (1994), processo é definido como um conjunto de atividades inter-relacionadas, que transforma entradas em saídas. Para o IEEE (1990), processo é uma seqüência de passos realizados para um determinado propósito.

O conceito de processo de software pode ser definido como um conjunto de atividades, métodos, práticas e tecnologias que as pessoas utilizam para desenvolver e manter o software e seus produtos relacionados (Paulk *et al.* 1995. A Figura 2.1, ilustra esta definição.

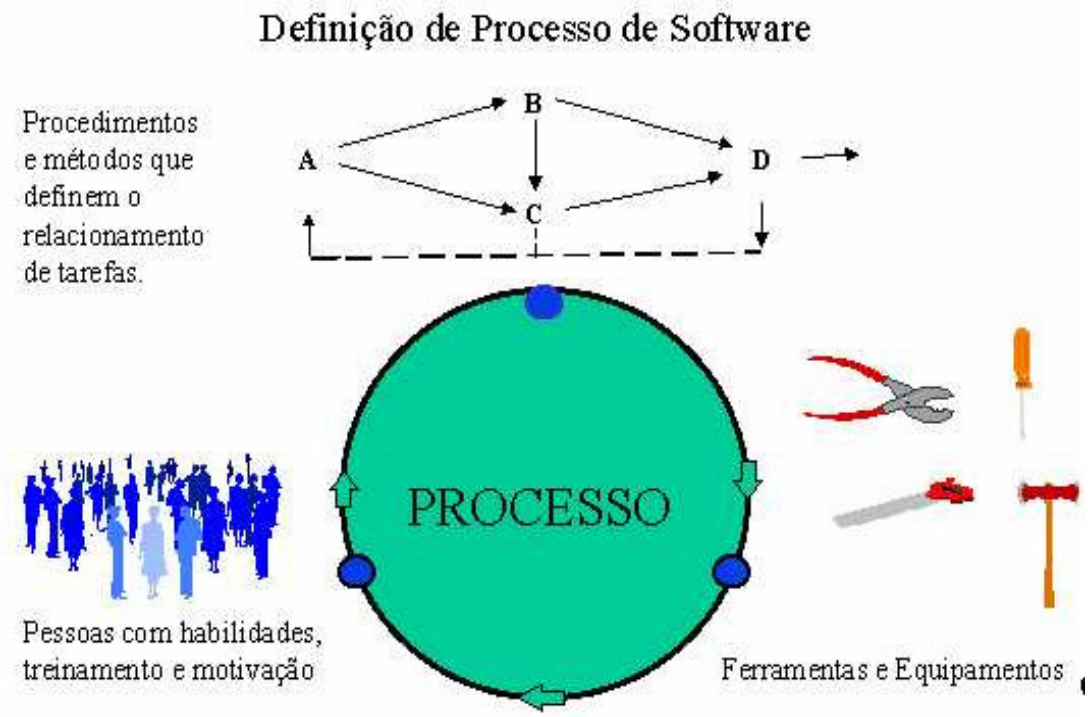

Figura 2.1 - O Processo de Software e seus Componentes. Fonte: (Pêssoa, 2004).

Pessôa (2004) considera que um processo de software para funcionar perfeitamente, deve ter seus procedimentos e métodos descrevendo a relação entre as tarefas, deve possuir ferramentas e equipamentos que dão suporte à realização das tarefas, simplificando e automatizando o trabalho e as pessoas com perfil adequado que foram treinadas nos métodos e nas ferramentas para, assim poder realizar as atividades adequadamente. Esse conjunto deve estar integrado harmoniosamente para funcionar de forma eficaz.

Pessoa (2004), afirma ainda que o software é resultado do processo de projeto, adota-se como premissa que a qualidade de um software é altamente influenciada pela qualidade do processo utilizado no seu desenvolvimento e manutenção. Essa premissa implica tanto em foco no processo quanto no produto.

Segundo Sommerville (2003), durante os últimos anos, ampliou-se o interesse por parte das organizações que desenvolvem software pela melhoria de seus processos. A melhoria do processo significa compreender os processos existentes e modificá-los, a fim de melhorar a qualidade do produto e/ou reduzir os recursos aplicados no desenvolvimento.

2.1.2. Qualidade de Produto de Software

Segundo Tsukumo (1997), a qualidade de software é amplamente determinada pela qualidade dos processos utilizados para o desenvolvimento. Deste modo, a melhoria da qualidade de software é influenciada pela melhoria da qualidade dos processos. Avaliar a qualidade de um produto de software é verificar, através de técnicas e atividades operacionais o quanto os requisitos definidos são atendidos.

De acordo com Moreira (2004), pessoas com diferentes interesses sobre um produto têm visões diferentes sobre o conceito de qualidade. Por exemplo, clientes (mercado) usualmente consideram que o software tem qualidade se possui características que atendam suas necessidades. Desenvolvedores usualmente vêem a qualidade através das medidas de suas propriedades que são comparadas com indicadores de qualidade preestabelecidos. Para o setor de software um produto de qualidade é aquele com custo mínimo associado ao retrabalho durante o desenvolvimento e após a entrega do produto.

2.1.3. Engenharia de Domínio

A engenharia de domínio consiste numa técnica bastante utilizada para a estruturação de componentes propícios a serem reutilizados. O fundamento básico dessa técnica é a análise da lógica do negócio nas quais os componentes se inserem . Sendo assim, os componentes que contiverem as funcionalidades das entidades básicas do domínio serão considerados candidatos a reutilização. Por exemplo, se o domínio é varejo os elementos do domínio serão caixas, estoque, vendas etc.

2.2. *Modelos de Maturidade*

2.2.1. CMMI-SE/SW

A sigla CMMI representa as iniciais de *Capability Maturity Model Integration* e nomeia tanto um projeto, quanto os modelos resultantes deste projeto. O projeto CMMI, está sendo executado pelo *Software Engineering Institute* (SEI), em cooperação com a indústria mundial e o governo dos Estados Unidos. O objetivo é consolidar um *framework* para modelos, evoluir e integrar modelos desenvolvidos pelo SEI (inicialmente os modelos SW-CMM, SE-CMM e IPD-CMM), e gerar seus produtos associados, incluindo material de treinamento e método de avaliação. Estes são os três modelos que foram evoluídos e integrados inicialmente: a versão 2.0 do *Capability Maturity Model for Software* (SW-CMM), o *System Engineering Capability Maturity Model*: *Electronic Industries Alliance* - EIA 731 (SE-CMM) e o *Integrated Product Development Capability Maturity Model* (IPD-CMM).

Segundo Salviano (2003), esta integração e evolução tiveram como objetivo principal a redução do custo da implementação de melhoria de processo multidisciplinar baseada em modelos. Multidisciplinar porque, além da engenharia de software, o CMMI cobre também a engenharia de sistemas, aquisição e a cadeia de desenvolvimento de produto. Esta redução de custo é obtida por meio da eliminação de inconsistências, redução de duplicações, melhoria da clareza e entendimento, utilização de terminologia comum, utilização de estilo consistente, estabelecimento de regras de construção uniformes, manutenção de componentes comuns, garantia da consistência com a Norma ISO/IEC 15504.

O nome do modelo CMMI pode ser traduzido para "Modelo Integrado de Maturidade da Capacidade". Em agosto de 2000 foi lançada a versão 1.0 do CMMI, também chamada de CMMI-SE/SW v1 (SEI, 2004). Após outras versões intermediárias, em 10 de março de 2002 foi lançada a versão atual denominada de CMMI-SE/SW/IPPD/SS *version* 1.1 (CMMISM *for Systems Engineering* / *Software Engineering* / *Integrated Product and Process Development* / *Supplier Sourcing*, *version* 1.1, em português CMMISM para Engenharia de Sistemas / Engenharia de Software / Desenvolvimento Integrado de Produtos e Processos / Fornecimento).

2.2.1.1. Arquitetura do Modelo

A arquitetura do modelo CMMI é composta pela definição de um conjunto de áreas de processo - PA, organizadas em duas representações diferentes: uma como um modelo por estágio, semelhante ao SW-CMM, e outra como um modelo contínuo (semelhante à ISO/IEC 15504). A versão atual é composta por 25 áreas de processo, conforme ilustrado na Figura 2.2.

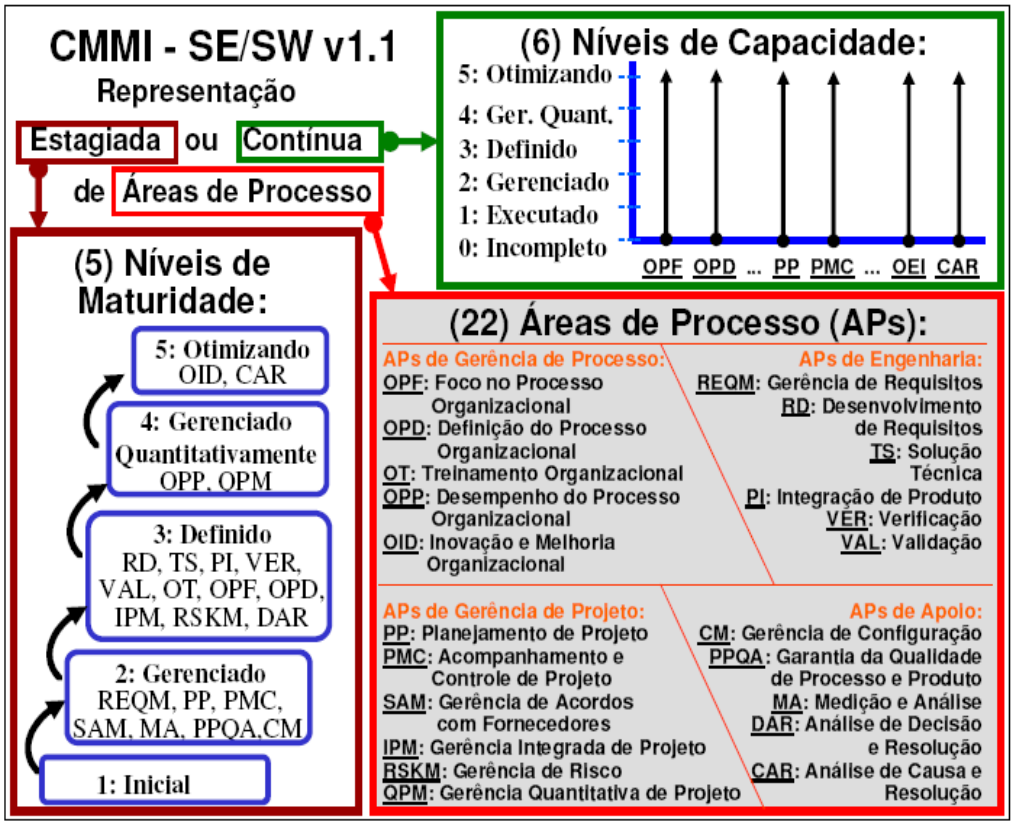

Figura 2.2 - CMMI - Áreas de Processo em duas representações. Fonte: (Salvino, 2006).

Cada área de processo é definida no modelo por meio da descrição de seu propósito, notas introdutórias, relacionamentos com outras áreas, metas específicas, metas genéricas, práticas e subpráticas específicas, práticas e subpráticas genéricas, produtos de trabalho típicos e referências para outros elementos do modelo relacionados.

Na representação por estágio, as 25 áreas de processo estão agrupadas em 4 níveis de maturidade: níveis 2, 3, 4 e 5. O nível 1 não contém nenhuma área de processo e a única exigência para que a empresa de software seja qualificada neste nível é a sua própria existência. Em relação a esta representação, o processo, de desenvolvimento e manutenção, de software ou sistema de uma unidade organizacional, pode estar classificado em um dos seguintes cinco níveis de maturidade:

- Nível 1: Inicial (*Initial*)
- Nível 2: Gerenciado (*Managed*)
- Nível 3: Definido (*Defined*)
- Nível 4: Gerenciado Quantitativamente (*Quantitatively Managed*)
- Nível 5: Otimizando (*Optimizing*)

Cada nível de maturidade é definido pelo conjunto de áreas de processo. Na representação contínua, as mesmas 25 áreas de processo estão agrupadas em quatro grupos (gerência de processos, gerência de projetos, engenharia e suporte) e são definidos seis níveis de capacidade de processo. Nesta representação, o conjunto de atividades correspondente a cada uma destas áreas de processo, pode ter sua capacidade de execução classificada em um dos seguintes seis níveis de capacidade de processo:

- Nível 0: Incompleto (*Incomplete*)
- Nível 1: Executado (*Performed*)
- Nível 2: Gerenciado (*Managed*)
- Nível 3: Definido (*Defined*)
- Nível 4: Gerenciado Quantitativamente (*Quantitatively Managed*)
- Nível 5: Otimizando (*Optimizing*)

Cada nível de capacidade é definido por um conjunto de características que o processo deve satisfazer para estar naquele nível. A área de Planejamento de Projetos, por exemplo, contém 8 metas, 8 práticas, 31 subpráticas e ocupa 29 páginas.

Como o objetivo do CMMI é representar no modelo metas e recomendações genéricas para orientar a melhoria dos processos em geral, não são descritas soluções prontas para serem executadas nas organizações. Cabe a cada organização entender e interpretar estas orientações no contexto, objetivo e estratégia de negócio da organização para obter melhorias relevantes em seu processo de desenvolvimento de software.

Em relação ao SW-CMM, o CMMI por estágio é uma revisão deste modelo, com ajustes. No nível 2 de maturidade, por exemplo, foi incluída a área de processo de medição e análise como uma ampliação deste assunto, que já era coberto em parte em cada área do SW-CMM. No nível 3, por exemplo, a área de engenharia de produto do SW-CMM foi mais bem descrita por meio de cinco áreas de processo: Desenvolvimento de Requisitos, Solução Técnica, Integração de Produto, Verificação, e Validação. Outras mudanças ocorrem para refletir melhor a orientação para atendimento dos níveis de maturidade (Salviano, 2003).

Em relação à utilização para melhoria de processo, o CMMI por estágio sugere abordagens semelhantes àquelas utilizadas com sucesso com o SW-CMM. Em relação ao CMMI contínuo, a tendência é toda a experiência de utilização da ISO/IEC 15504 ser aproveitada para ajustar as abordagens já utilizadas com sucesso pelo SW-CMM.

2.2.1.2. Elementos do Modelo

Os elementos que compõe o modelo CMMI são agrupados em três categorias que refletem como eles devem ser interpretados (SEI, 2002):

- **Elementos Requeridos:** metas específicas e genéricas são elementos requeridos do modelo. Estes elementos devem ser estabelecidos pelos processos definidos e implementados pela organização. São essenciais para avaliar a satisfação de uma área de processo. O estabelecimento (ou satisfação) das metas é usado em avaliações como base para satisfação da área de processo e maturidade organizacional. Apenas a declaração das metas é um elemento requerido do modelo, o título e qualquer nota associada são considerados elementos informativos do modelo;

- **Elementos Esperados:** práticas específicas e as genéricas são elementos esperados no modelo. Estes elementos descrevem o que uma organização tipicamente irá implementar para satisfazer um componente requerido. Servem como guia para aqueles que implementam melhorias e/ou executam avaliações. Espera-se que as práticas descritas, ou as alternativas aceitáveis para elas, estejam presentes nos processos planejados e implementados da organização, antes que as metas sejam consideradas satisfeitas. Apenas a declaração da prática é um elemento esperado do modelo, o título e qualquer nota associada são considerados elementos informativos do modelo;
- **Componentes Informativos:** sub-práticas, produtos de trabalho típicos, particularidades da disciplina, elaborações de práticas genéricas, títulos, notas e referências são elementos informativos do modelo. Estes elementos ajudam o usuário do modelo a entender as metas e práticas e como elas podem ser estabelecidas, fornecendo detalhes para ajudar a pensar como estabelecer as práticas e metas. Características comuns são elementos não classificados do modelo, apenas constituem um grupo que fornece uma maneira de apresentar as práticas genéricas. Quando se usa um modelo CMMI como guia, processos são planejados e implementados em conformidade com os elementos esperados e requeridos das áreas de processo. Conformidade com uma área de processo significa que nos processos planejados e implementados existe um processo associado (ou processos) que endereça as práticas específicas e genéricas da área de processo, ou alternativas, que geram um resultado de acordo com a meta associada com aquela prática específica ou genérica.

2.2.1.3. Esquema de Numeração

Cada meta específica tem um número começando com SG, do inglês *Specific Goal* (SG 1, por exemplo). Cada meta genérica tem um número começando com GG, do inglês *Generic Goal* (GG 2, por exemplo). Práticas específicas começam com SP, do inglês *Specific Practice*, seguido por um número no formato x.y. O x é o número da meta específica à qual a prática corresponde e y é o número de seqüência da prática. As práticas genéricas estão numeradas de forma semelhante, começando com GP, do inglês *Generic Practice.*

2.2.2. FAA iCMM

Em 1997, a Administração Federal de Aviação Norte-Americana desenvolveu o *FAA integrated Capability Model (iCMM) version* 1.0 (Ibrahim et al. 2001) para prover melhoria nos seus processos de engenharia, gerência e aquisição em uma maneira integrada, efetiva e eficiente. O modelo integrou três disciplinas do CMM que estavam sendo utilizadas separadamente em diferentes diretórios do FAA: o Modelo de Maturidade da Capacidade de Aquisição de Software - SA-CMM 96, o Modelo de Maturidade da Capacidade para Software - SW-CMM 93 e o Modelo de Maturidade da Capacidade de Engenharia de Sistemas - SE-CMM (Ibrahin *et al* 2001).

Os métodos do modelo FAA iCMM foram destinados para serem usados por organizações com intuito de poderem avaliar suas próprias práticas e processos organizacionais e definir melhorias para os mesmos. O modelo pode ser usado para um auto-aperfeiçoamento da organização, desde que seus usuários entendam seu propósito e suas limitações, sendo estruturado para fornecer uma representação contínua e estagiada para capacidade de processo.

O modelo fornece um guia prático em processos usados em empreendimentos ou organizações engajadas na aquisição, fornecimento, engenharia, desenvolvimento, operação, evolução, suporte e gerenciamento de produtos e serviços. Integra as disciplinas engenharia de software, engenharia de sistemas, aquisição e desenvolvimento integrado de processos e produtos. Inclui processos de liderança e estratégia para assegurar a correta compreensão dos projetos. Baseada na implementação com sucesso da versão 1.0, a FAA revisou e ampliou o modelo para vir com seu novo lançamento, o FAA iCMM v2.0.

A Figura 2.3, ilustra a relação da estrutura simplificada do modelo iCMM versão 2.0, com a identificação e nome das três categorias de processo, dos vinte e três processos, dos seis níveis de capacidade e de nove dos outros modelos integrados nesta versão.

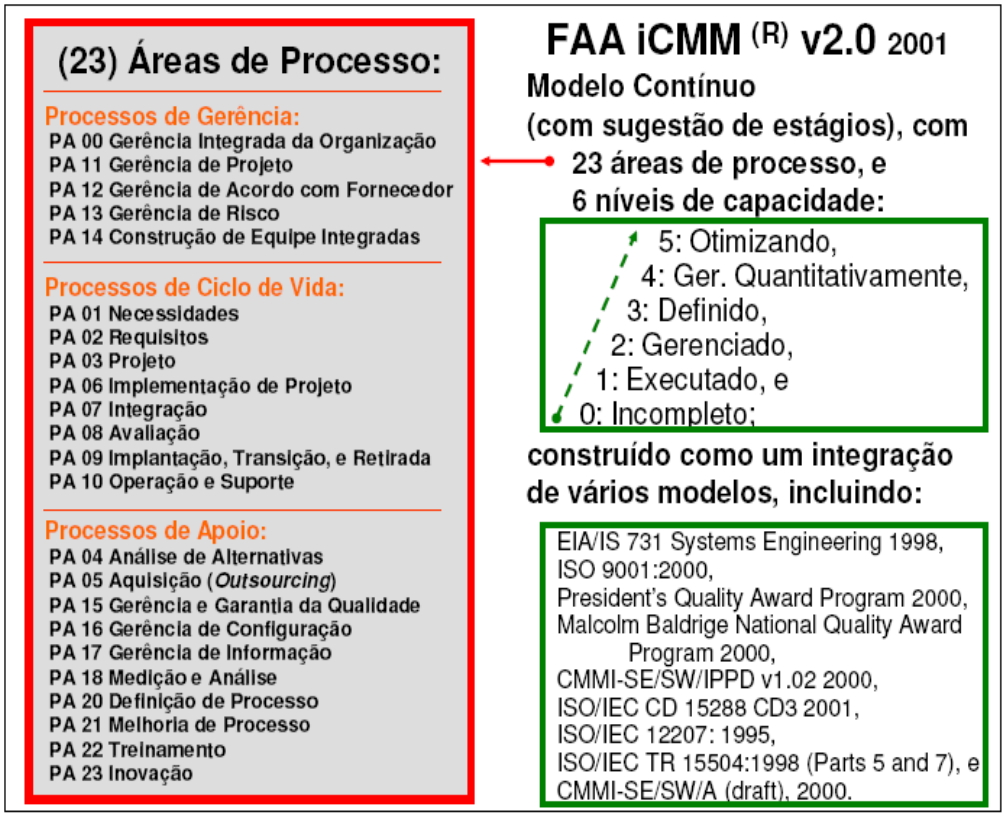

Figura 2.3 - Composição sintética do modelo FAA iCMM. Fonte: (Brito et. al.,2005).

O FAA tem intensivamente se envolvido com o projeto CMMI, provendo material de referência e relato de experiência de lições aprendidas pelo modelo. Em contrapartida, produtos de trabalho do CMMI também foram utilizados no desenvolvimento da versão 2.0.

Como apresentado na Figura 2.4, o modelo FAA-iCMM possui duas dimensões, uma voltada ao desempenho do processo (*Process Dimension*) e outra voltada à capacitação do processo (*Capability Process*). Os níveis de capacidade contêm metas a serem atingidas, que serão alcançadas por meio de práticas genéricas. Além disso, os níveis de capacidade aplicam áreas de processo, que contêm metas a serem atingidas e promovem um nível maior de detalhes em relação às práticas genéricas.

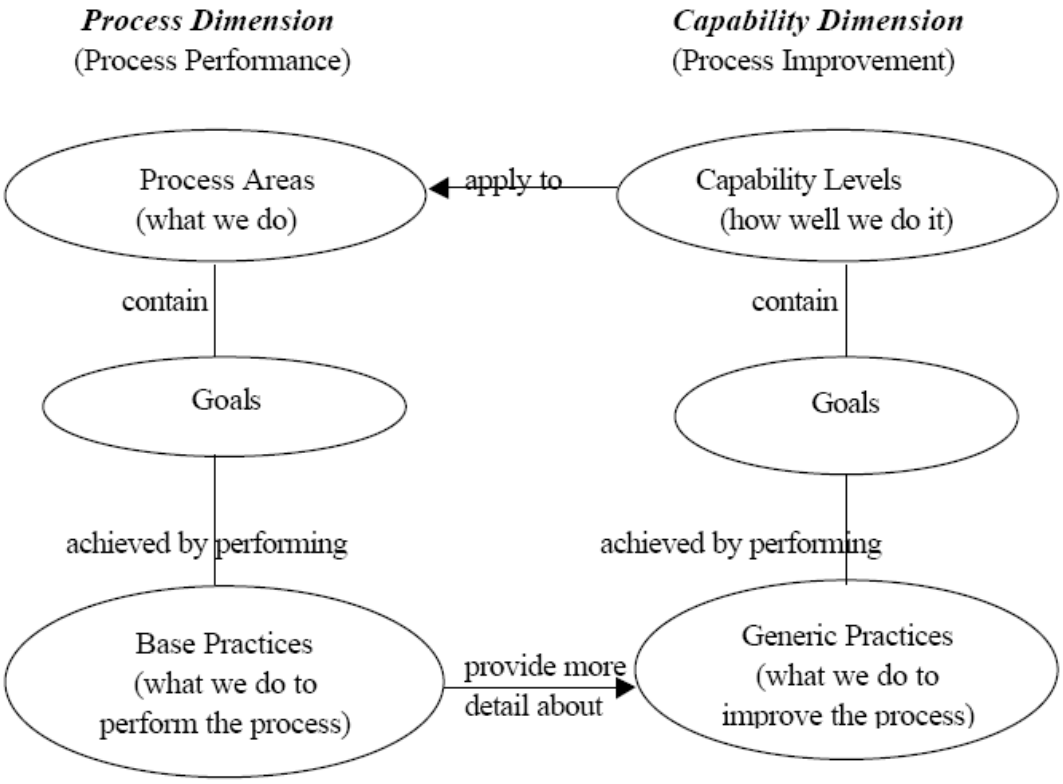

Figura 2.4 - Estrutura do modelos FAA iCMM. Fonte: (Ibrahim et. al., 2001).

2.2.2.1. Componentes do Modelo

Os seguintes critérios foram usados para integrar e derivar as áreas de processo (PAs, do inglês *process areas*) do FAA iCCM (Brito et. al.,2005):

- Devem agrupar atividades relacionadas em um local de fácil uso;
- Devem ser adequadas para implementação em múltiplas organizações;
- Geralmente são mais simples de serem melhoradas, quando tomados como um processo distinto;
- Geralmente são mais simples de serem melhoradas por um grupo com interesses similares no processo;
- Incluem todas as práticas *base pratices* (PB – práticas base) requeridas para alcançar seus os objetivos;

Em geral, uma prática básica resume as principais características da execução da área de processo. Os seguintes critérios são utilizados para as práticas básicas:

- A sobreposição entre práticas básicas devem ser minimizadas;
- As práticas básicas devem ser aplicadas utilizando múltiplos métodos em múltiplos contextos de negócio. Não podem, porém especificar um método ou ferramenta particular;
- Os resultados de uma prática básica devem ser objetivamente observados utilizando métodos razoáveis de investigação (ex.: entrevistas, revisão de documentos).

2.2.2.2. Áreas de Processo Relevantes para Reuso

Nesta subseção são descritas as áreas de processo consideradas relevantes para o escopo do trabalho possuindo de certa maneira relacionamento com o desenvolvimento baseado em reuso.

As áreas de processo são (Brito *et. al*.,2005):

- **PA 01 Necessidade:** O propósito desta PA é evidenciar, analisar, esclarecer e documentar as necessidades e outras expectativas dos clientes e de outros *stakeholders*[1]. Estabelecer e manter a comunicação com clientes e com os *stakeholders* relevantes por todo o ciclo de vida. Assegurar o contínuo entendimento do que satisfará estas necessidades.
- **PA 02 Requisitos:** O propósito desta PA é desenvolver requisitos que reúnam as necessidades dos clientes, análise dos produtos, serviços e outros requisitos, produzindo, assim, um conjunto detalhado de requisitos e gerenciar estes requisitos por todo o ciclo de vida.
- **PA 03 Design:** O propósito desta PA é estabelecer e manter uma arquitetura, determinando as soluções para as necessidades e requisitos dos clientes e outros *stakeholders*.
- **PA 06 Implementação de Design:** O propósito desta PA é produzir um componente específico para uma certa solução.
- **PA 07 Integração:** O propósito desta PA é garantir que os elementos do produto ou serviço funcionarão em conjunto com outros elementos.

[1] Stakeholder: é um grupo ou um indivíduo que é afetado ou de alguma maneira é responsável pelo resultado de alguma empreitada. Os stakeholders podem incluir membros do projeto, fornecedores, clientes, usuários finais e outros.

2.3. Desenvolvimento Baseado em Componentes

Para uma melhor compreensão do presente trabalho, este capítulo apresenta conceitos fundamentais sobre a utilização de componentes de software abordando seus processos de desenvolvimento.

2.3.1. Introdução

Na maioria dos projetos de software, ocorre algum reuso de software. Isso, em geral, acontece informalmente, quando as pessoas que trabalham no projeto conhecem projetos ou código similares àquele exigido. Eles recorrem a esses produtos, fazem as modificações conforme necessário e as incorporam em seu software e/ou sistemas. O reuso é visto frequentemente como essencial para o rápido desenvolvimento de sistemas (Somerville, 2003).

Sistemas de informação enfrentam vários problemas (Short, 1997). Entre eles podemos citar:

- Responder às requisições de implementar rapidamente e efetivamente aplicações que suportem novos *workflow* em processos de negócios reestruturados e com funções cruzadas;
- Forjar ligações entre pacotes de software, aplicações construídas nas próprias empresas e sistemas legados;
- Adaptar software de prateleira com os requisitos únicos corporativos;
- Encontrar caminhos para fazer com que aplicações de níveis departamentais trabalhem em maior escala e integradas em nível corporativo;
- Atualizar e melhorar aplicações existente, tornando-as rápidas e economicas, sem sacrificar qualidade e usabilidade.

De acordo com Short (1997), estes problemas levam à crença de que o desenvolvimento baseado em componentes é “a resposta que anuncia um processo de desenvolvimento de aplicações reformado”. Pode-se dividir as vantagens do desenvolvimento baseado em componentes em duas categorias, baseadas e não baseadas em reuso.

Entre as vantagens daqueles que são baseadas em reuso podemos encontrar (Short, 1997):

- **Contenção de complexidade –** projetos em desenvolvimento com foco em um ou em pequeno número de componentes tem seu escopo reduzido e riscos mais facilmente gerenciáveis. Equipes de projetos menores são geralmente mais produtivas;

- **Oportunidade para massivo desenvolvimento em paralelo –** fronteiras do projeto definidas em redor de estáveis definições de componentes encorajam desenvolvimento externo (*outsourcing*).

  Outsourcing de manutenção também pode ocorrer uma vez que quem publica os componentes podem revender e manter componentes desenvolvidos internamente;

- **Estáveis definições de componentes também encorajam a flexibilidade –** um componente que garante satisfazer um requisito pode ser substituído por outro que satisfaça aquele e alguns novos requisitos;

- **Teste incremental –** componentes facilitam teste de unidade e suportam a construção de testes progressivos;

- **Componentes encapsulados agem como um *firewall* (sistema de segurança) para mudanças –** efeitos das mudanças são limitados e a manutenção se torna mais fácil e barata.

- **Redução do tempo de entrega –** pela consulta de catálogos de componentes, algumas funcionalidades são normalmente encontradas. Componentes selecionados podem ser produzidos externamente;

- **Redução de custos de desenvolvimento –** o uso de componentes externos faz com que sejam reduzidos os custos de desenvolvimento uma vez que as equipes internas e focam em novas e únicas funcionalidades;

- **Aumento de produtividade –** equipes de desenvolvimento focam mais na montagem do sistema do que em desenvolvimento. Novos desenvolvedores podem ser mais produtivos mais cedo;

- **Reduzido riscos de problemas –** uma vez que pré-fabricados componentes são certificados e testados. Interações válidas são bem especificadas e documentadas;

- **Maior consistência no reuso –** componentes impõem uma arquitetura padrão para as aplicações;
- **Seleção do melhor em um conjunto de soluções –** o mercado de componentes se desenvolve e o fornecimento de pontos de funções empacotados como componentes cresce. Competição entre fornecedores faz com que o preço diminua.

2.3.2. Desenvolvimento com e para Reuso

Com a popularização do DBC, a necessidade de novos processos voltados para esse paradigma é uma realidade. Isso acontece porque os processos de desenvolvimento tradicionais não são totalmente adequados ao desenvolvimento de sistemas baseados em componentes. Mais especificamente, esses processos devem conter fases e métodos que ofereçam, entre outras coisas, técnicas que permitam o empacotamento de componentes com o objetivo específico de serem reutilizados.

É importante notar aqui que há questões distintas relacionadas ao uso de componentes no desenvolvimento de software (desenvolvimento **com** reuso) e ao desenvolvimento de componentes (desenvolvimento **para** reuso) propriamente ditos. As principais vantagens atribuídas à tecnologia de componentes são relativas ao desenvolvimento com reuso (ganho de produtividade, redução de *time to market*[2], etc.). O desenvolvimento do componente em si tende a ser mais complexo e demorado devido à preocupação extra com reuso (MCT, 2005).

[2] Time to market: tempo para lançamento no mercado

2.4. *Componentes*

2.4.1. O que são Componentes?

O que exatamente vem a ser um componente? Esta é uma questão difícil de responder, mesmo que seja restrita a noção de componente às tecnologias atuais, porque a palavra componente é usada informalmente para significar uma variedade de coisas, cada uma com validade igual (Cheesman & Daniels, 2001).

Na literatura encontramos diferentes definições de componentes. Por exemplo, na visão geral técnica da Microsoft de *Component Object Model* (COM), o componente é definido como "um pedaço de software compilado, o qual está oferecendo um serviço", de acordo com Box, citado por Crnkovic & Larson (2002). D'Souza & Wills definem componentes como parte de software reusável a qual é desenvolvida independentemente e pode ser combinada com outros componentes para construir maiores unidades. Estas partes podem ser adaptadas, mas não modificadas, citados por Crnkovic & Larson (2002).

Segundo o MCT (2005), uma analogia que pode auxiliar na compreensão sobre componentes de software é o brinquedo Lego (2006). As peças de Lego, clássico brinquedo de montar, podem simbolizar a idéia de componentes de software em um sentido mais geral. As peças do Lego permitem a montagem de automóveis, casas, aviões, navios e muitas outras mais, limitada apenas pela criatividade dos montadores. A montagem no Lego consiste em reunir diversas peças de diferentes formatos, encaixá-las entre si e obter, no final, uma figura totalmente nova. Existem algumas peças que podem e provavelmente serão utilizadas na construção de qualquer uma das montagens. As pequenas peças retangulares (aquelas que vêm em maior número) provavelmente estarão presentes em todas as montagens feitas com o brinquedo. Outras peças, por sua vez, têm funções mais específicas e serão utilizadas dentro de sua especialidade, dificilmente se adaptando a outros contextos. Um eixo, por exemplo, pode ser utilizado em automóveis, aviões, mas provavelmente não o será na montagem de casas. Nada impede, porém, que o montador visualize uma nova função para o eixo em um novo contexto e o utilize em uma montagem. Este exemplo, embora simples, mostra que componentes são reutilizáveis, podem possuir uso mais geral ou específico, embora todos sejam genéricos dentro do escopo considerado, e componentes se interagem com outros componentes.

O MCT (2005), apresenta as propriedades de um componente de software. A Tabela 2.1 exibe algumas destas propriedades.

Tabela 2.1- Propriedades de Componentes de Software.

Propriedade	Descrição	Vantagem
Uso múltiplo	Utilizável em diversos projetos	Compartilhamento do custo do desenvolvimento entre diversos projetos. Ampliação paulatina da confiança no componente, ao ser empregado em diversos projetos
Sem contexto específico (genérico)	Desenvolvido independentemente de qualquer projeto específico ou contexto de sistema. No caso de necessidades específicas, estas devem estar explícitas.	Aumento da probabilidade do componente ser reutilizado.
Composição	Pode ser combinado com outros componentes, com base nos contratos das interfaces.	Alta produtividade no desenvolvimento alcançada através de montagem de sistemas complexos a partir de componentes.
Encapsulamento	Apenas as interfaces são visíveis e a implementação não pode ser alterada.	Evita variações múltiplas; todos os usos de um componentes se beneficiam de um mesmo esforço de manutenção e atualização.
Unidades independentes	Componentes podem ser configurados e instalados como unidades atômicas independentemente do resto do sistema	Permite que clientes alterem, atualizem e combinem componentes mesmo com o sistema em produção, aumentando a competição entre os fornecedores de componentes.

Fonte: (MTC, 2005).

São encontradas, comumente, nas definições de componentes as seguintes características (Crnkovic & Larson, 2002):

- Componentes são unidades de composição;

- Componentes precisam ser especificados de um modo que é possível sua composição com outros componentes;
- A composição entre componentes e sua integração em sistemas precisa ocorrer de um modo previsível.

Os princípios fundamentais para componentes são os mesmos da Orientação a Objetos (Cheesman & Daniels, 2001):

- Unificação de dados e função – um objeto de software consiste de dados (ou estados) e funções que processam estes dados;
- Encapsulamento – o cliente do objeto é isolado do modo em que os dados do objeto de software é armazenado ou como as funções são implementadas. O cliente depende da especificação e não da implementação. Esta separação é muito importante e a chave para gerenciar dependências e reduzir o acoplamento;
- Identidade – cada objeto de software possui a sua identidade independente do seu estado.

E o que difere o componente de um objeto de software é a explícita separação de sua especificação da sua implementação e a divisão da sua especificação em interfaces (Cheesman & Daniels, 2001).

2.4.2. Forma de um Componente

Um componente pode aparecer em várias e diferentes formas (Cheesman & Daniels, 2001). São elas:

- Padrão de componente (*Component Standard*) – ambiente padrão que serve como um modelo onde os componentes se enquadram. Como exemplos de tais padrões, temos o EJB (*Enterprise Java Beans*) e o COM+ (*Component Object Model*) da Microsoft. Grandes organizações podem possuir definidos os seus próprios padrões;
- Especificação do componente (*Component Specification*) – é a especificação de uma unidade de software que descreve o comportamento de um conjunto de bjetos do componente e define uma unidade de implementação;

- Interface do componente (*Component Interface*) – define o conjunto de comportamentos que podem ser oferecidos pelo componente;
- Implementação do componente (*Component Implementation*) – consiste na realização da especificação do componente. A implementação pode ser instalada e substituída independentemente de outros componentes. Não é necessariamente um único item físico como um único arquivo;
- Componente instalado (*Installed Component*) – representa uma cópia da implementação do componente registrada em um ambiente de execução. Este registro permite ao ambiente identificar os componentes instalados e criar uma nova instância do mesmo;
- Objeto do componente (*Component Object*) – instância de um componente instalado. É um conceito de tempo de execução. Um objeto com seus próprios dados e única identidade. Um componente instalado pode possuir muitos objetos do componente (o que requer uma identificação explícita) ou um único (cuja identificação pode ser implícita).

As formas que podem ser tomadas pelo componente são exibidas na **Erro! Fonte de referência não encontrada.**.

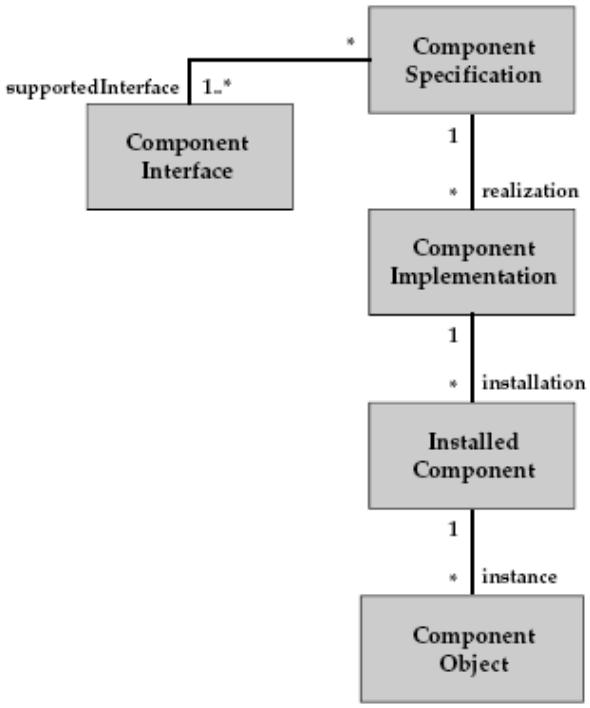

Figura 2.5 - Formas assumidas por um Componente. Fonte: (Chessman & Daneils, 2001).

2.4.3. Arquitetura de Componentes e de Sistemas

Chessman & Daniels (2001) definem arquitetura de sistemas como o conjunto de partes que compõe uma completa instalação de software, incluindo as responsabilidades destas partes, suas interconexões e mesmo a tecnologia apropriada; e a arquitetura de componentes como o conjunto de componentes de software ao nível da aplicação, seus relacionamentos estruturais e suas dependências comportamentais.

Em seus estudos, Chessman & Daniels (2001) dividem a arquitetura de um sistema em quatro camadas: Interface com o Usuário, Diálogo com o Usuário, Serviços de Sistema e Serviços de Negócio. Uma breve descrição destas camadas é apresentada a seguir:

- **Interface com o usuário –** compreende a apresentação das informações ao usuário e captura suas entradas de dados. Sistemas externos podem ser utilizados também;
- **Diálogo com o usuário –** gerencia os diálogos do usuário em uma sessão;
- **Serviços de sistema –** representação externa do sistema, provendo acesso aos serviços do sistema. Esta camada funciona como uma fachada para os serviços providos pela camada inferior, provendo um contexto no qual serviços de negócio

mais genéricos são utilizados de acordo com as necessidades de um sistema em particular;

- **Serviços de negócio –** implementação do núcleo das informações do negócio, regras e transformações. Os componentes desta camada são reutilizados em diferentes sistemas.

2.4.4. Engenharia de Software Baseada em Componentes (ESBC)

Segundo Crnkovic e Larsson (2002), clientes e fornecedores têm esperado muito do desenvolvimento em componentes, mas suas expectativas não têm sido sempre alcançadas. A experiência tem mostrado que o desenvolvimento baseado em componentes requer uma abordagem sistemática com foco nos aspectos de componentes no desenvolvimento de software. As disciplinas tradicionais da engenharia de software precisam ser ajustadas para a nova abordagem, e novos procedimentos precisam ser desenvolvidos. A engenharia de software baseada em componentes, tem sido reconhecida como uma nova subdisciplina da engenharia de software. As maiores metas da engenharia de software baseada em componentes são:

- Prover suporte ao desenvolvimento de sistemas como montagem de componentes;
- Auxiliar no desenvolvimento de componentes como entidades reusáveis;
- Facilitar a manutenção e atualização de sistemas pela customização e substituição de seus componentes.

A construção de sistemas a partir de componentes e a construção de componentes e a construção de componentes para diferentes sistemas requerem metodologias e processos estabelecidos não apenas em relação aos aspectos de desenvolvimento e manutenção, mas também ao inteiro ciclo de vida do componente e do sistema, incluindo aspectos organizacionais, de mercado e legal, entre outros. Em adição aos temas específicos da engenharia de software baseada em componentes como especificações de componentes e tecnologias, algumas disciplinas e processos da engenharia de software requerem metodologias específicas para aplicação em DBC. Muitas destas metodologias ainda não têm sido estabelecidas na prática, e algumas ainda não têm sido refinadas teoricamente. O progresso do desenvolvimento de software em um futuro próximo dependerá muito do estabelecimento com sucesso da ESBC, um ponto que é reconhecido pela indústria e academia (Crnkovic & Larson, 2002).

De acordo com Crnkovic e Larson (2002), DBC e ESBC estão iniciando suas fases de expansão. O DBC é reconhecido como uma nova abordagem poderosa que melhorará significativamente – se não revolucionar – o uso e desenvolvimento de software em geral. Pode-se esperar que componentes e serviços baseados em componentes sejam amplamente utilizados por pessoas que não são programadores quando construindo suas aplicações. Ferramentas para construir tais aplicações pela montagem de componentes serão desenvolvidas. Atualização automática de componentes através da internet, já presente em muitas aplicações hoje, será o meio padrão de melhoria das aplicações.

Outra tendência observada é a padronização de componentes específicos de domínio no nível de interfaces. Isto facilita a construção de aplicações a partir de componentes adquiridos de diferentes vendedores. Crnokovic & Larson (2002) afirmam que a padronização de componentes específicos de domínios requer a padronização dos processos específicos de domínio. O suporte à troca de informações entre componentes, aplicações e sistemas distribuídos através da internet está sendo desenvolvido e tecnologias como o XML (*Extensible Markup Language*) são utilizadas para troca de informações através da internet.

Crnokovic & Larson (2002) também apresentam alguns dos desafios da CBSE. Tais desafios são: a especificação dos componentes, os modelos dos componentes, o ciclo de vida do software baseado em componentes, a previsibilidade da composição, componentes confiáveis (*trusted components*), certificação de componentes, configuração dos componentes, suporte de ferramentas e sistemas confiáveis (*dependable systems*).

Embora a especificação dos componentes tenha sido um problema endereçado desde o início do desenvolvimento de modelos de componentes, não se tem chegado a um consenso sobre como deve ser especificado um componente. Dos modelos de componentes, pode-se dizer que mesmo que os modelos de desenvolvimento existentes demonstrem tecnologias poderosas, eles possuem muitas características ambíguas, são incompletos e difíceis de usar. As relações entre a arquitetura do sistema e os modelos de componentes não estão definidas precisamente.

O ciclo de vida de software baseado em componentes está se tornando mais complexo porque muitas fases são separadas em atividades não sincronizadas. Por exemplo, o desenvolvimento do componente pode ser totalmente independente do desenvolvimento do sistema que irá usar o componente. O processo de engenharia dos requisitos é muito mais complexo porque os possíveis componentes candidatos, normalmente, faltam uma ou mais características que o sistema requer.

Ainda relacionado ao ciclo de vida do software baseado em componentes, mesmo que alguns componentes estejam individualmente bem encaixados no sistema, não é óbvio que eles irão funcionar de modo ótimo em combinação com os outros no sistema. Estas restrições podem requerer uma outra abordagem na engenharia de requisitos – uma análise da praticidade dos requisitos em relação aos componentes disponíveis e da conseqüente modificação dos requisitos. Uma estratégia para gerenciar riscos na seleção e evolução dos componentes é necessária por causa das muitas incertezas que cercam os processos de seleção e evolução dos componentes.

Similarmente, muitas questões restam nas fases posteriores do ciclo de vida de sistemas baseados em componentes. Desde que os sistemas baseados em componentes incluem componentes com diferentes ciclos de vida, o problema da evolução do sistema se torna significativamente mais complexo. Muitos tipos diferentes de problemas ainda não foram resolvidos. Existem questões técnicas (Pode o sistema ser tecnicamente atualizado pela substituição de componentes?), questões administrativas e organizacionais (Quais componentes podem ser atualizados, quais componentes poderiam e quais precisam ser atualizados?), questões legais (Quem é responsável pela falha do sistema? Quem produziu o sistema ou quem produziu os componentes?) e inúmeras outras. ESBC é uma abordagem nova e pequena experiência tem sido adquirida considerando-se a manutenção de tais sistemas.

Mesmo que seja assumido que todos os atributos relevantes do componente estejam especificados, não há necessariamente o conhecimento de que estes atributos sejam os correspondentes do sistema ao qual eles irão fazer parte. A abordagem ideal – derivar atributos do sistema a partir de atributos dos componentes – ainda é tema de pesquisa.

Desde que a tendência é entregar componentes na forma binária e o processo de desenvolvimento do componente está fora do controle do usuário do componente, questões relacionadas à confiabilidade do componente se tornam muito importantes. Um meio de classificação dos componentes é a certificação dos mesmos. Apesar da existência de uma crença comum de que certificação significa absoluta confiabilidade, de fato isto meramente provê os resultados de testes realizados e uma descrição do ambiente em que os testes foram realizados. Embora a certificação seja um procedimento padrão em muitos domínios, ela ainda não tem sido estabelecida em software em geral e especificamente não para componentes de software.

Sistemas complexos podem incluir muitos componentes, os quais, por sua vez, incluem outros componentes. Em muitos casos, composições de componentes são tratadas como componentes. Quando estruturas complexas começam a ser trabalhadas, problemas envolvendo configurações estruturais começam a aparecer. Por exemplo, duas composições podem possuir o mesmo componente. Este componente será tratado como duas entidades ou o sistema aceitará o componente como uma única instância, comum para as duas composições? O que ocorre se duas versões diferentes do componente são incorporadas nas duas composições? Qual versão será selecionada? O que ocorre se as duas versões não são compatíveis? Os problemas de atualização dinâmica de componentes já são conhecidos, mas suas soluções ainda são temas de pesquisas. Um meio de manusear estes tipos de sistemas complexos com muitos componentes é fazendo uso de arquiteturas de linhas de produto que impõem regras para a configuração de componentes.

O propósito da engenharia de software é prover soluções práticas para problemas práticos, e a existência de ferramentas apropriadas é essencial para o desempenho da CBSE. Ferramentas, como o *Visual Basic* da *Microsoft*, têm provado ser extremamente bem sucedidas, mas muitas outras ferramentas ainda têm que aparecer: ferramentas de seleção e avaliação de componentes, repositórios de componentes e ferramentas para gerência dos repositórios, ferramentas de teste de componente, ferramentas para projeto baseado em componentes, ferramentas de análise de sistemas em tempo de execução e ferramentas de configuração de componentes, entre outras. O objetivo da ESBC é construir sistemas de modo simples e eficiente, e isto pode ser alcançado unicamente com um extensivo suporte de ferramentas.

O uso de DBC em domínios de segurança ou críticos, sistemas de tempo real e diferentes sistemas de controle de processos, no qual requisitos de confiabilidade são particularmente rigorosos, é muito desafiante. Um grande problema com a DBC é a possibilidade limitada de garantir a qualidade e outros atributos não funcionais dos componentes e, deste modo, uma incapacidade de garantir atributos específicos do sistema.

2.5. Modelos de Processo de Desenvolvimento

2.5.1. UML Components

O UML *Components* (Cheesman & Daniels, 2001) é um processo de DBC voltado para o desenvolvimento de sistemas de informacão. Devido a sua simplicidade, o UML Components adota uma arquitetura pré-definida em quatro camadas. Duas dessas camadas são destacadas durante o desenvolvimento: camada de sistema e camada de negócio. A camada de sistema é responsável por agrupar os componentes que implementam as funcionalidades especificadas para o sistema. Porém, para que essas funcionalidades possam ser implementadas, esses componentes podem necessitar de algumas funcionalidades comuns ao domínio. Os componentes que implementam essas funcionalidades gerais são posicionados na camada de negócio. (Brito *et al,* 2005).

As quatro camadas da arquitetura e o papel de cada uma podem ser vistos na Figura 2.6.

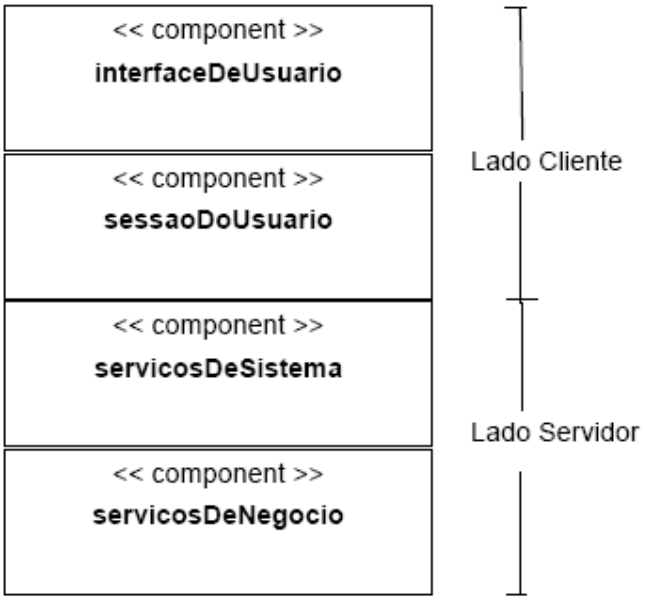

Figura 2.6 - Camadas da Arquitetura do Sistema. Fonte: (MCT, 2005).

A Figura 2.7 mostra como no UML *Components*, o desenvolvimento é dividido em 6 fases: (i) especificacão de requisitos *(requirements denition)*; (ii) especificacão dos componentes *(specication)*; (iii) suprimento dos componentes *(provisioning)*; (iv) montagem do sistema *(assembly)*; (v) testes *(test)*; e (vi) implantação *(deployment)*. Esse processo é iterativo e enfatiza basicamente a fase de especificação dos componentes. Por essa razão essa fase é detalhada em três sub-fases: (i) identificação dos componentes *(component identification)*; (ii) interação entre os componentes *(component interaction)*; e (iii) especificação final *(component specication)*.

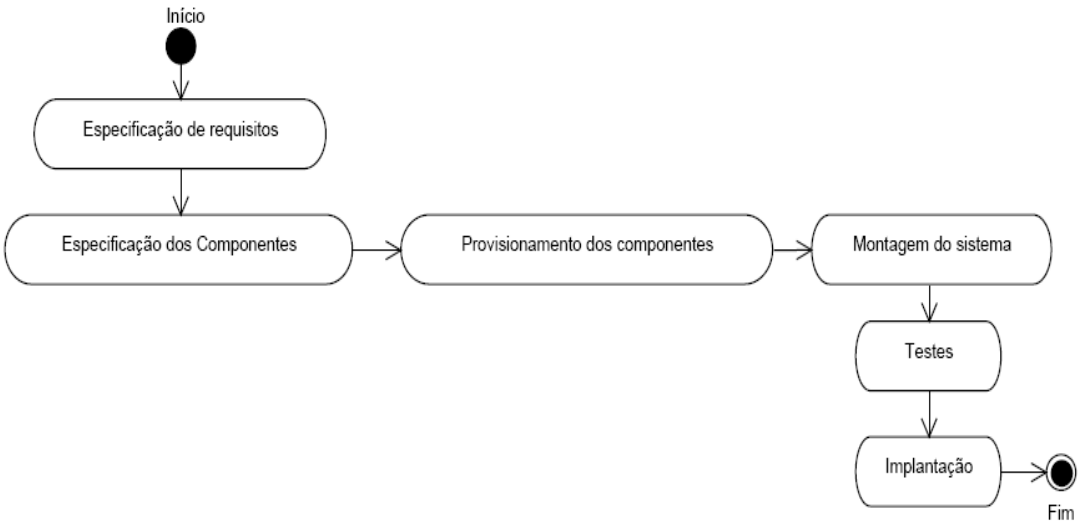

Figura 2.7 - Fases do Modelo. Fonte: (Cheesman & Daniels, 2001).

2.5.1.1. Especificação de Requisitos

Apesar de deixar bem claro o papel desta fase, inclusive descrevendo os artefatos que devem ser produzidos nela, o UML *Components* não detalha as atividades da sua execução. Na fase de especificação dos requisitos, as funcionalidades do sistema são representadas através de casos de uso. Além disso, é especificado o modelo conceitual do negócio, que representa as entidades básicas da lógica do mesmo. Dadas as suas importâncias, esses artefatos são considerados as principais saídas desta fase.

O modelo conceitual do negócio representa o domínio do problema. Por essa razão ele necessita ser entendido e firmado entre os interessados no sistema. O propósito principal desse modelo é proporcionar uma linguagem comum entre os envolvidos no projeto. Além disso, esse modelo é a base para a identificação dos componentes da camada de negócio.

2.5.1.2. Especificação de Componentes

Essa especiação acontece tomando como entrada os artefatos produzidos na fase de especificação de requisitos: (i) modelo conceitual do negócio; e o (ii) modelo de casos de uso.

Os principais artefatos de saída da fase de especificação dos componentes são três: (i) especificação das interfaces (refinamento das assinaturas); (ii) especificação dos componentes (interfaces providas e requeridas); e (iii) a arquitetura do sistema, como conseqüência da definição das dependências entre os componentes. Como pode ser visto na **Erro! Fonte de referência não encontrada.** para produzir esses artefatos, o UML *Components* divide essa fase em três estágios: (i) identicação dos componentes; (ii) interação entre os componentes; e (iii) especicacão final.

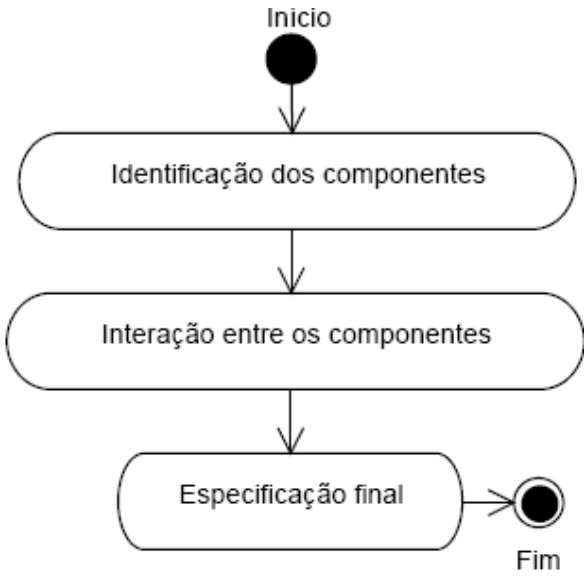

Figura 2.8 - Especificação dos Componentes no UML Components. Fonte: (Brito et al., 2005).

Com a especificação dos componentes finalizada, é chegada a hora de providenciar cada um dos componentes para em seguida compor o sistema, testá-lo e entregá-lo ao cliente para utilização. Apesar da importância dessas fases, o UML *Components* não detalha a sua execução.

2.5.1.3. Provisionamento de Componentes

A etapa de provisionamento deve garantir a materialização dos componentes especificados. Essa materialização pode ocorrer de duas formas: (i) reutilização de componentes já existentes; e (ii) implementação de componentes novos.

Para o caso de reutilizacão de componentes, por causa de possíveis inconsistências entre as interfaces especificadas e reutilizadas, pode ser necessário implementar adaptadores (Somerville, 2001). No caso dos componentes implementados, _é necessário especificar as suas estruturas internas.

2.5.1.4. Montagem do Sistema

A fase de montagem é responsável pela materialização da configuração da arquitetura do sistema. Sendo assim, ela consiste na integração dos componentes para construção da aplicação. Basicamente, existem duas atividades desempenhadas durante esta fase: (i) a construção dos conectores, que implementam um código de mapeamento entre os componentes do sistema; e (ii) a construção do programa principal, que deve conter o código de instanciação dos componentes, dos conectores, além da ligação" entre eles, que é efetuada dinamicamente.

2.5.1.5. Testes

Apesar da importância dos testes, este restringem ao aspecto puramente de desenvolvimento, não contendo atividades sistemáticas de testes que tratem essa questão desde a especificação dos requisitos. Em parte isso é uma vantagem, uma vez que possibilita a utilização conjunta com algum processo de testes distintos.

2.5.1.6. Implantação

A conclusão da fase de implantação não significa que o ciclo de desenvolvimento do sistema foi finalizado. Pelo contrário, muito provavelmente serão necessários alguns ciclos de manutenções corretivas para que o sistema possa ser considerado estável (Pressman, 2001). Além disso, mesmo com a estabilidade do sistema, nada impede que novas mudanças de requisitos sejam solicitadas por parte do cliente. As atividades de evolução da especificação do sistema, decorrentes dessas mudanças de requisitos, são conhecidas como manutenções perfectivas (ou evolutivas) (Sommerville, 2001; Pressman, 2001).

2.5.2. Processo de Desenvolvimento Baseado em Componentes da Unicamp

É proposto um processo de DBC voltado para a integração de sistemas. O objetivo é promover a reutilização de componentes já existentes, de forma sistemática e garantindo a qualidade requerida para o sistema. Outra característica importante desse processo é o fato dele ser genérico e facilmente adaptável a outros processos de desenvolvimento existentes, tais como UML *Components* (Cheesman & Daniels, 2001) e *Catalysis* (D'Souza & Wills).

Como mostrado na Figura 2.9, o processo de desenvolvimento apresentado aqui é composto de oito passos básicos: (i) especificação de requisitos; (ii) projeto arquitetural; (iii) reutilização de COTS; (iv) implementação de componentes novos; (v) implementação dos conectores e adaptadores; (vi) integração dos componentes do sistema; (vii) validação do sistema; e (viii) correções do sistema.

Figura 2.9 - Visão Geral do Processo. Fonte: (Brito et al., 2005).

A partir das descrições textuais iniciais, os requisitos do sistema são especificados. Em seguida, a visão funcional abstrata já começa a dar lugar a uma visão mais específica de implementação, através da estruturação inicial da arquitetura de software. Essa especificação precoce da arquitetura deve modularizar a estrutura do sistema através da identificação dos seus principais componentes abstratos. Com os componentes arquiteturais identificados, o processo passa por uma fase de busca por candidatos a desempenhar o papel de cada um deles. Vale a pena salientar o fato de que o principal fator a ser considerado na escolha dos componentes adequados é o equilíbrio entre a qualidade das funcionalidades implementadas (proximidade dos requisitos) e o custo de implementação do sistema.

Dependendo da granularidade do componente, ou seja, do seu grau de detalhe, pode ser necessário executar o processo recursivamente, no intuito de reutilizar as partes que o constituem. O fato do processo ser recursivo o torna adequado tanto ao desenvolvimento de sistemas de grande porte, quanto ao desenvolvimento de sistemas pequeno.

As seções seguintes detalham cada uma das oito atividades do processo, que foram apresentadas na Figura 2.9.

2.5.2.1. Especificação de Requisitos

Os requisitos funcionais, que representam o comportamento esperado pelo sistema, pode ser mapeado para os serviços, tarefas ou funções que o sistema deve oferecer. Já os requisitos não-funcionais, apesar de não representarem funcionalidades diretamente, eles podem interferir na maneira como o sistema deve executá-las. Dessa forma, a organizacão estrutural do sistema, materializada na arquitetura de software, é diretamente relacionada a esses requisitos.

Conforme mostrado na Figura 2.10, a fase de especificação de requisitos abrange basicamente sete atividades: (i) análise e representação do domínio do problema; (ii) identificação inicial dos requisitos funcionais; (iii) construção do protótipo; (iv) refinamento dos requisitos funcionais; (v) especificação dos requisitos de qualidade; (vi) especicação das restrições de desenvolvimento; e (vii) definir restrições de reutilização.

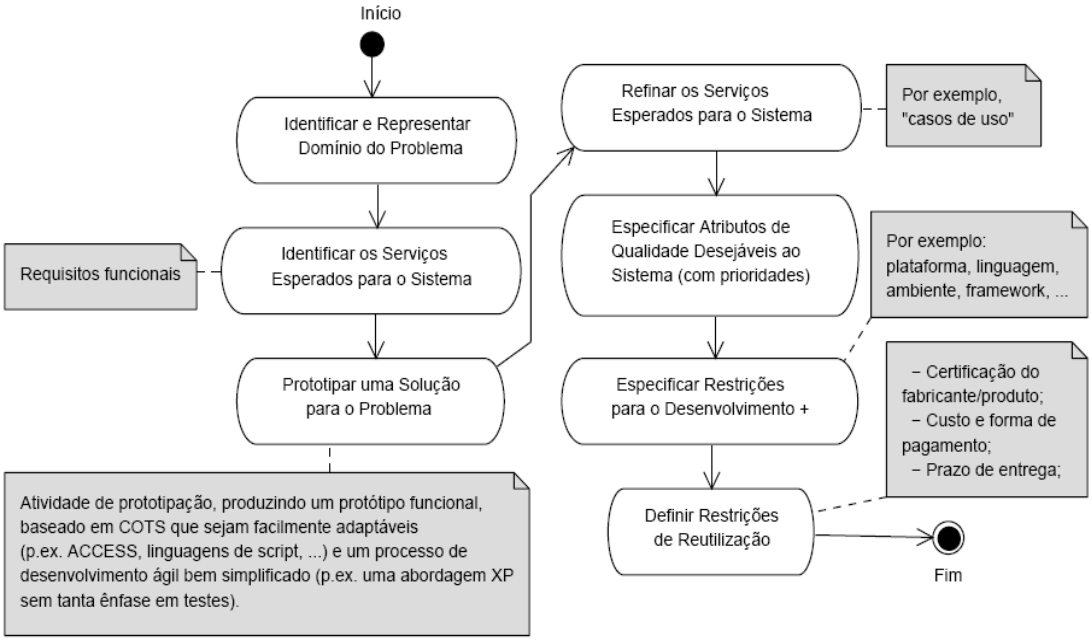

Figura 2.10 - Especificação de Restriçoes de Desenvolvimento. Fonte: (Brito et al., 2005).

2.5.2.2. Projeto Arquitetural

As principais vantagens de se enfatizar o papel arquitetural do sistema no modelo são: (i) ter uma estruturação abstrata, que facilita o entendimento; (ii) alta granularidade de reutilização, que aliado ao DBC pode ser determinante para a redução do tempo de desenvolvimento; e (iii) maior facilidade para adaptar, manter, evoluir e portar o sistema para outras plataformas, como conseqüências do baixo acoplamento proporcionado pelo uso de conectores arquiteturais, que explicitam a comunicação entre os componentes da arquitetura. Essa comunicação explicita facilita a substituição dos componentes, uma vez que torna possível a adaptação das mensagens que fluem entre eles [7, 22].

2.5.2.3. Reutilização de COTS

O método proposto define três formas de se materializar um componente: (i) reutilização de um componente já utilizado pela empresa; (ii) aquisição do componente a partir de um catálogo de terceiros; e (iii) implementação do componente. O *workflow* de execução dessa atividade de busca é apresentado na Figura 2.11. Nele, após a identificação dos COTS candidatos, é iniciado um processo de negociação e ajuste dos requisitos.

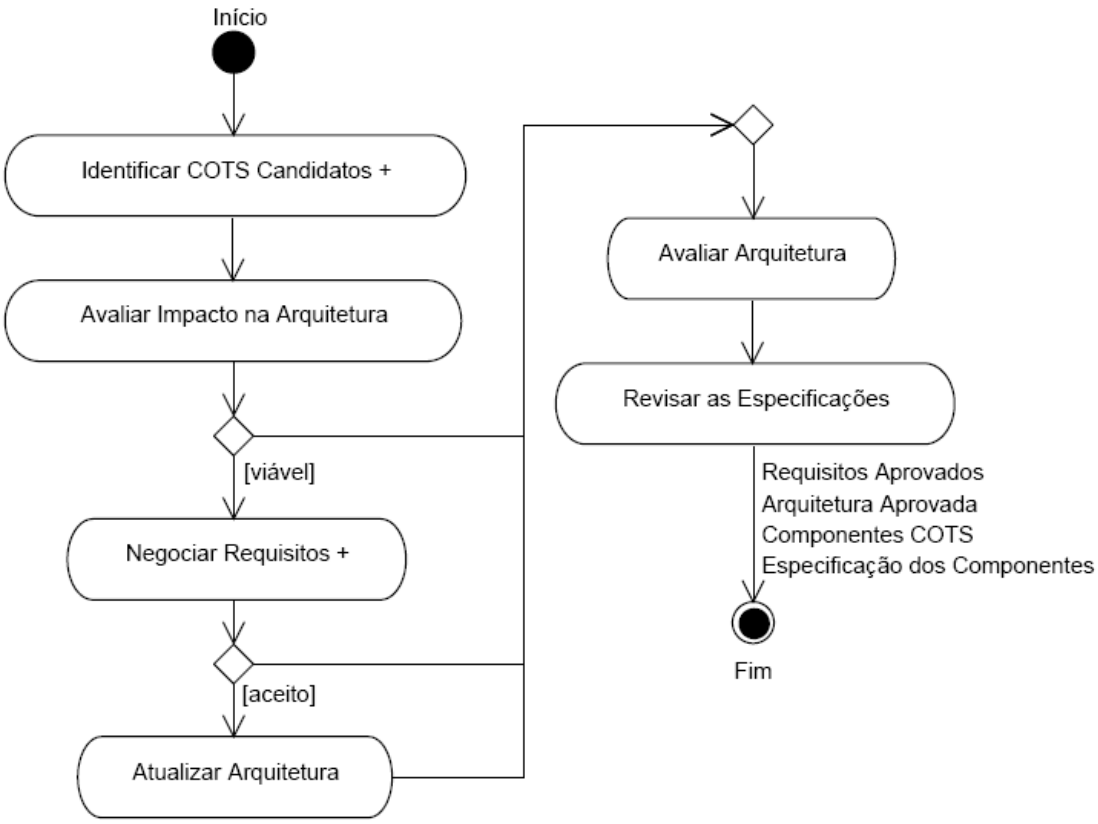

Figura 2.11 - Reutilização de COTs. Fonte: (Brito et al., 2005).

2.5.2.4. Implementação de Componentes Novos

Devido a característica recursiva desse processo de desenvolvimento, para a implementação de um componente com granularidade alta, a execução pode ser executada recursivamente, a partir da fase de especicacão dos requisitos. O desenvolvedor pode optar por desenvolver o componente a partir de sua fase inicial. Essa opção é válida para os casos onde a granularidade do componente já atingiu o limite da relação custo x benefício.

2.5.2.5. Implementação dos Conectores e Adaptadores de Integração dos Componentes de Sistema

A materialização das conexões dos componentes são realizadas através da ligação das interfaces requeridas de um componente _as respectivas interfaces providas de outros. Porém, como mostrado na Figura 2.12, dependendo da similaridade ou não entre as duas interfaces, pode ser necessário implementar algum procedimento de adaptação entre elas. Nos casos onde a adaptação é necessária, o conector passa a ser chamado de adaptador [20],

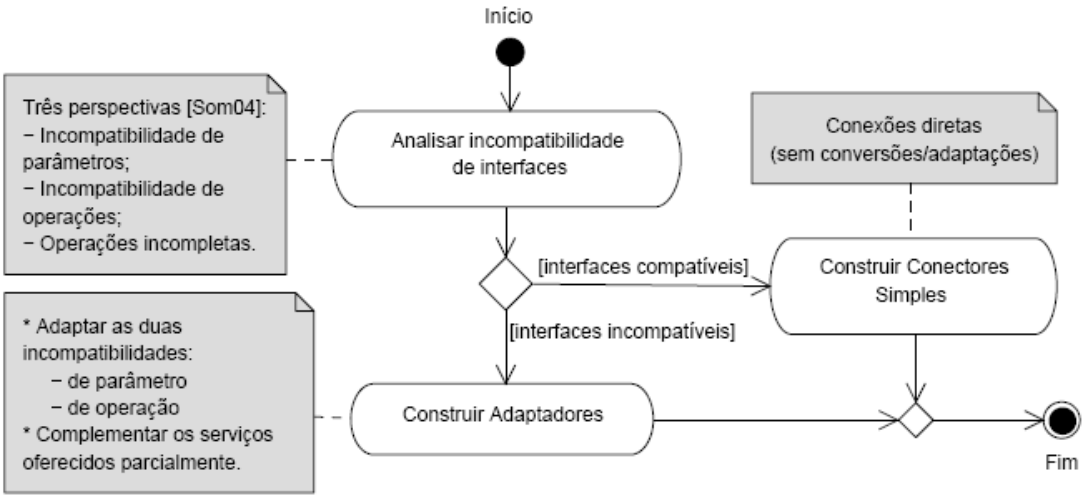

Figura 2.12 - Composição de Componentes. Fonte: (Brito et al., 2005).

Por representar o elo de comunicação entre componentes arquiteturais, os conectores são os únicos componentes do sistema que possuem o conhecimento dos seus fluxos interativos. Logo, são considerados os locais ideais para a implementação dos requisitos de qualidade do sistema, como tolerância a falhas.

Após a finalização da especificação dos conectores do sistema, deve-se prosseguir com a sua implementação, conforme o modelo de componentes adotado. Além disso, _é necessário implementar as rotinas de ligação dos componentes. Esse código é responsável por instanciar cada um dos pares de componentes cliente e servidor, assim como o seu respectivo conector. Em seguida, as interfaces requeridas do componente cliente devem ser inicializada com a implementação da interface provida do conector. Finalmente, a interface requerida do conector deve ser inicializada com a classe que implementa a interface provida do componente servidor. Para ficar mais claro, a Figura 2.13 mostra a estruturação de dois componentes (A e B), unidos por um conector AB.

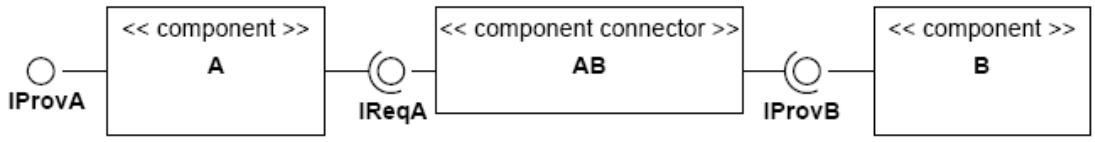

Figura 2.13 - Estrutura de dois componentes conectados. Fonte: (Brito et al., 2005).

Esse código de ligação dinâmica dos componentes faz parte da implementação do programa principal do sistema.

2.5.2.6. Validação e Correção do Sistema

Basicamente, a validação do software consiste em uma seqüência de testes, elaborados a partir dos documentos de requisitos. Em sistemas reais, devido ao grande número de serviços que podem ser oferecidos, essa atividade deve ser executada automaticamente, com o auxílio de ferramentas CASE.

2.5.3. Abordagem PRO2PI para melhoria de Processo

Segundo Salviano (2006), a abordagem PRO2PI (Perfil de Capacidade de Processo para Melhoria de Processo, em inglês, *Process Capability Profile to Process Improvement*) é uma abordagem para uma melhoria de processo orientada a perfis de capacidade de processo. Um PRO2PI pode conter elementos de vários modelos de capacidade de processo. Esses modelos podem ser estagiados ou contínuos e também podem ser modelos de outros tipos que não de capacidade de processo. Todos os modelos fontes de elementos para um PRO2PI são vistos como se fossem modelos de capacidade de processo segundo a arquitetura contínua.

A abordagem PRO2PI é baseada nos seguintes pressupostos:

a) A melhoria de processo de software baseada nos níveis de capacidade dos modelos estagiados, como, por exemplo, os modelos SW-CMM, CMMI-SE/SW e MR-MPS, podem ser considerados como um exemplo de utilização de modelos contínuos, considerando cada nível de maturidade como um perfil de capacidade de processo;

b) O processo de uma organização pode ser considerado como o conjunto dos processos mais relevantes da organização;

c) O processo de uma organização em um determinado instante no tempo pode ser modelado, segundo o aspecto de capacidade de processo, em um perfil de capacidade de processo, e esse perfil pode ser chamado de perfil atual, podendo ser obtido por meio de uma avaliação de processo;

d) O processo alvo para um ciclo de melhoria de processo em uma organização também pode ser modelado, segundo o aspecto de capacidade de processo, em um perfil de capacidade de processo, e esse perfil pode ser chamado de perfil alvo;

e) Uma organização tende a utilizar mais de um modelo de capacidade de processo ou mesmo de outro tipo de modelo de referência para orientar um ciclo de melhoria de processo;

f) Existe uma tendência para a ampliação da abrangência da melhoria de processo de software para a melhoria de qualquer processo tecnológico que seja intensivo em atividade criativa humana; e

g) As abordagens de melhoria de processo de software atuais, como, por exemplo, as abordagens IDEAL e ciclo de melhoria da ISO/IEC 15504-4, devem ser evoluídas para uma engenharia de processo, de software e qualquer outra área tecnológica intensiva em atividade humana, orientada a perfis de capacidade de processo.

A descrição de PRO2PI é feita em Salviano (2006) com a visão geral da utilização de PRO2PI e a descrição dos quatro elementos que compõem a abordagem PRO2PI. Um dos quatro elementos que compõem a abordagem PRO2PI é o ciclo de melhoria PRO2PI-CYCLE, composto por fases correspondentes a ciclos de melhoria convencionais tais como o FAA iCMM.

2.6. Considerações Finais do Capítulo

Neste capítulo foi apresentada uma visão geral sobre Engenharia e Qualidade de Software, qualidade do produto e processo de software, normas e modelos de qualidade de software, uma abordagem para melhoria de processo e também sobre o desenvolvimento baseado em componentes.

Baseado nas características dos modelos de processo e maturidade apresentados neste capítulo, foram identificadas algumas as áreas de processo do CMMI-SE/SW que de alguma forma contemplam o reuso, mas de maneira insatisfatória. Essas áreas serão reformuladas para se tornarem especializadas para tal propósito. Essa reformulação se dará a partir da adequação das áreas de processo, consideradas relevantes para o reuso, junto às práticas proposta pelos modelos de processo apresentados no referencial teórico deste trabalho.

3. METODOLOGIA

Neste capítulo é apresentada a metodologia de trabalho utilizada para a especialização das áreas de processo voltadas para o reuso do CMMI-SW/SE v1.1.

Para alcançar este objetivo foram planejadas três atividades principais.

A primeira atividade realizada foi um estudo bibliográfico relacionado ao desenvolvimento de software com reuso baseado nos modelos de processo de desenvolvimento de componentes: o UML *Components*, o Processo de DBC da Unicamp e os modelo referenciais de maturidade de capacidade de processo, FAA iCMM e o CMMI.

A segunda atividade realizada foi um estudo específico e profundo de algumas áreas de processo do CMMI, identificadas como relacionadas ao reuso, a análise das atividades e os produtos típicos de trabalho mais relevantes.

A terceira atividade realizada foi a especialização das áreas de processo através do incremento das mesmas com o conteúdo obtido através da revisão bibliográfica dos outros modelos utilizados no trabalho.

O modelo CMMI-SE/SW versão 1.1, foi escolhido como modelo de referência de capacidade de processo neste trabalho por sua qualidade, seu respaldo na comunidade de engenharia de software e por sua disseminação na indústria de software.

3.1. Tipo de Pesquisa

O método de pesquisa utilizado neste trabalho foi o de pesquisa aplicada, com objetivos de caráter exploratório, utilizando procedimentos operacionais e fundamentados em pesquisas bibliográficas e documentais.

De acordo com Jung (2004), a finalidade da pesquisa de objetivo exploratório é a descoberta de teorias e práticas que modificarão as existentes, a obtenção de alternativas ao conhecimento científico convalidado e, principalmente, inovações tecnológicas (produto ou processos).

Para Gil (1991), a pesquisa bibliográfica é desenvolvida a partir de material já elaborado, constituído principalmente de livros e artigos científicos. Enquanto a pesquisa bibliográfica se utiliza fundamentalmente das contribuições dos diversos autores sobre determinado assunto, a pesquisa documental vale-se de materiais que não receberam ainda um tratamento analítico, ou que ainda podem ser reelaborados de açodo com o objetivo da pesquisa.

No presente trabalho, foi escolhida a utilização em conjunto da pesquisa documental com a bibliográfica, pois os dois tipos de referências complementam uma a outra.

De acordo com Jung (2004), o procedimento de pesquisa operacional tem por princípio a investigação de forma sistemática e racional dos processos envolvidos na realização de uma atividade produtiva, com a finalidade de orientar a melhor opção para a tomada de decisões. (Jung, 2004).

Segundo Jung (2004), a pesquisa em campo tem por finalidade, em muitos casos, coletar dados que estejam necessariamente sob ação das variáveis presentes no local.

3.2. Procedimentos Metodológicos

O trabalho de pesquisa apresentado neste documento é parte do projeto de pesquisa COMPGOV (Componentes para o Governo), para o desenvolvimento de uma biblioteca pública de componentes de software com o seu processo de gerência para aplicação no domínio do governo eletrônico e um modelo de maturidade de capacitação de processo. Tal projeto é financiado pela FINEP – Financiadora de Estudos e Projetos (Dantas, 2005). O projeto iniciou-se em 01 de janeiro de 2005 com seu prazo de conclusão em 31 de dezembro de 2006, mas tendo sua finalização prorrogada devido a não conclusão de todas as metas do projeto.

O projeto COMPGOV é resultado de uma parceria entre o Centro de Estudos em Sistemas Avançados do Recife - C.E.S.A.R,, Universidade Federal da Paraíba - UFPB, Centro de Pesquisa Renato Archer - CenPRA, Universidade Estadual de Campinas - UNICAMP, CI&T e Centro de Informática da Universidade Ferderal de Pernambuco - CIN-UFPE. De acordo com documento do projeto, o COMPGOV é formado por 14 metas que foram atribuídas aos participantes do projeto:

- **M1 -** Definição do Processo de Reuso;
- **M2 -** Definição de um Processo Baseado em Componentes com Reuso;
- **M3 -** Especificação de uma Arquitetura;
- **M4 -** Definição de Provisionamento de Componentes e Conectores;
- **M5 -** Integração dos Componentes na Arquitetura;
- **M6 -** Especificação e Projeto de um Ambiente Integrado;
- **M7 -** Implementação de um Ambiente Integrado para Desenvolvimento e Testes;
- **M8 -** Concepção e Especificação do Modelo de Componentes;
- **M9 -** Concepção da Arquitetura do Serviço de Repositório;
- **M10 -** Especificação do Serviço de Repositório;
- **M12 -** Implementação do Serviço de Repositório;
- **M15 -** Definição do Modelo de Qualidade e Processo de Avaliação para Certificação dos Componentes;
- **M16 -** Desenvolver Sistema de Repositório Distribuído de Componentes;
- **M17 -** Desenvolver Componentes para o Domínio Aplicação de Governo Eletrônico.

Neste trabalho é apresentado o desenvolvimento de uma parte da meta 15. O escopo da meta 15 é definir o modelo de qualidade e processo de avaliação para certificação de componentes. Tal meta pode ser dividida nos seguinte sub-produtos (Dantas, 2005):

1. Modelo de Maturidade da Capacidade do Processo de Desenvolvimento Baseado em Componentes
2. Processo de Avaliação para Certificação de Componentes

Uma das atividades a serem realizadas para definição do Modelo de Qualidade, é a especialização de determinadas áreas relacionadas a reuso do CMMI que é objetivo deste trabalho. Nos capítulos posteriores serão apresentados quais foram os procedimentos metodológicos para atingir tal objetivo.

O trabalho tem como referência a meta M15 do projeto COMPGOV, que objetiva buscar a definição de um modelo de qualidade e maturidade de capacidade de processo, sendo executada seguindo uma estratégia embasada na abordagem PRO2PI para melhoria de processo de desenvolvimento de software baseado em competentes.

3.3. Definição dos Modelos

A estratégia é orientada pela definição de especializações dos modelos de capacitação de processos mais utilizados para engenharia de software em geral. Esta estratégia foca-se mais no uso de modelos consagrados do que na definição de um novo modelo. Esta estratégia é orientada por algumas decisões:

i) revisar o estado da arte e estado da prática da área, no caso, Desenvolvimento de Software Baseado em Componentes;

ii) identificar o modelo de capacidade para a melhoria de processo da engenharia de software em geral, que seja mais apropriado para as características da especialização e utilizá-lo como a base para a especialização;

iii) identificar ou definir um conjunto de novas áreas de processo para cobrir os principais aspectos específicos da Engenharia de Software Baseada em Componentes;

iv) identificar as áreas de processo do modelo identificado como base, mais afetadas pela Engenharia de Software Baseada em Componentes e prover alterações com relações ao uso das mesmas.

O item (iii) não será coberto neste trabalho devido ao considerável aumento de escopo e, por conseguinte, extrapolação dos prazos de conclusão do mesmo. Cabendo considerar a sua abordagem como trabalho futuro.

No contexto dos projetos COMPGOV, essa estratégia foi aplicada com as seguintes diretrizes chaves:

- após o estudo de processos, modelos e métodos abrangentes ao reuso na literatura atual, foi considerado o modelo CMMI-SE/SW versão 1.1, na sua representação estagiada, como um modelo de referência de capacidade de processo para engenharia de software em geral, pela sua qualidade e sua disseminação na indústria de software. Os outros modelos estudados como o ISO/IEC 15504-5 e o iCMM não foram escolhidos porque ainda não estão muito difundidos e o MR-MPS (SOFTEX, 2005) ser um modelo estagiado;
- identificar as áreas de processo do CMMI-SE/SW que mais se identificam com a Engenharia de Software Baseada em Componentes e prover alterações com relações ao uso das mesmas;
- considerar as práticas dos parceiros do projeto como referência inicial de práticas para Engenharia de Software Baseada em Componentes. Todos os três parceiros são reconhecidos como excelentes empresas, que já utilizam abordagens de CBD e melhoria de processo com CMMI-SE/SW e outros modelos;
- considerar o UML *Components* (Cheesman & Daniels, 2001) como a referência para um modelo de processo de desenvolvimento generalizado baseado em componentes, porque ele é simples, representativo e já é adotado por parceiros do projeto;
- utilizar o Processo para DBC com Reuso de Componentes desenvolvido pela UNICAMP também como modelo de referência pelo fato dele ser genérico e facilmente adaptável a outros processos de desenvolvimento existentes, tais como UML Components (Cheesman & Daniels) e Catalysis (D'Souza & Wills, 1999);
- Considerar o FAA iCMM, modelo de maturidade de capacidade de processos, como um modelo uma ferramenta para a adequação do perfil capacidade do CMMI a uma abordagem de desenvolvimento com componentes. Este modelo trata o reuso de forma mais abrangente, incluindo aspectos importantes como a Gerência de Configuração, a especificação das interfaces e a descontinuação do produto (MCT, 2005).

De acordo com o MCT (2005), as referências sobre componentes do CMMI, não são suficientes para o contexto do projeto COMPGOV. Não há, por exemplo, uma abordagem especial para Gerência de Configuração ou a referência a uma Política de Reuso. Cabe a meta M15 do COMPGOV a readequação do modelo através do incremento de suas áreas de processo com a utilização procedimentos complementares vistos nos modelos de processo de desenvolvimento e gerencial citados anteriormente.

Chessman & Daniels (2001) afirmam, que além do processo de desenvolvimento, que especifica e implementa o sistema a partir dos seus requisitos, para se desenvolver um sistema de software é necessário seguir ao mesmo tempo um processo gerencial, que esquematiza atividades, planeja liberações, aloca recursos e monitora progressos.

UML *Components* e o Processo de DBC da Unicamp são modelos de processo de desenvolvimento sendo considerados insatisfatórios, quando tomados isoladamente, segundo a ótica de Chessman & Daniels (2005). São, portanto, insuficientes para serem tomados como base para consolidação de um modelo de capacidade de processos voltado para reuso, pois carecem de um modelo de processo gerencial em suas estruturas como apresentado no CMMI.

Para DBC, o modelo mais significativo encontrado foi o OOSPICE (*Software Process Improvement and Capability dEtermination for Object Oriented / Component Based Software Development*) (Stallinger et al. 2002). A primeira idéia foi utilizá-lo o OOSPICE como o Modelo de Capacidade do Processo. A decisão de não usar o OOSPICE foi tomada devido a três razões principais:

- O OOSPICE não é um modelo público, e não foi obtido o acesso nem mesmo direito para usá-lo, o acesso foi concedido somente a artigos e apresentações disponíveis no site do OOSPICE. Foi solicitado o acesso ao modelo completo, mas ele não foi concedido e nem a permissão do seu uso;
- É necessário ter o controle de toda a tecnologia do projeto com o objetivo de facilitar sua utilização por outros parceiros do projeto.

3.4. Descrição do Desenvolvimento

As modificações das áreas de processo (PAs) do CMMI-SE/SW se deram através de estudo do modelo FAA iCMM, do processo simplificado de desenvolvimento de componentes UML *Components* (Cheesman & Daniels, 2001) e do processo proposto pela Universidade Estadual de Campinas (UNICAMP). A Figura 3.1 apresenta as etapas para a elaboração das PA's.

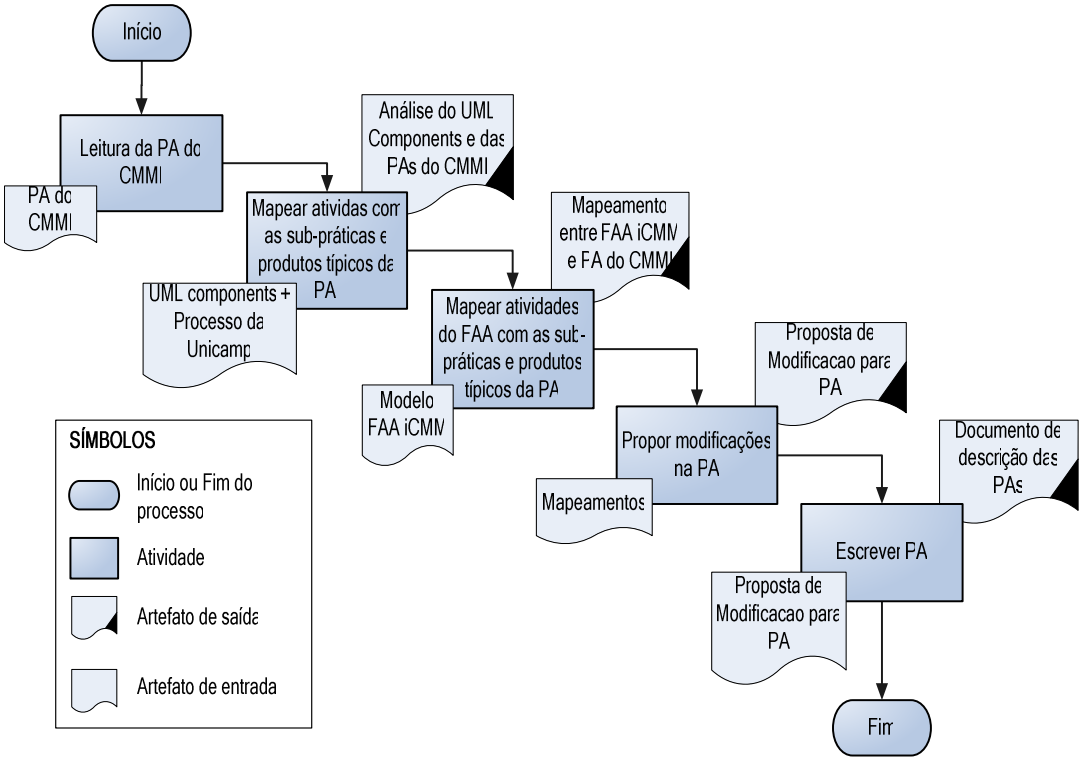

Figura 3.1 - Visão Geral da Metodologia. Fonte: (O Autor).

Primeiramente foi realizado um estudo de determinadas áreas de processo do CMMI-SE/SW, com o objetivo de entender melhor os aspectos considerados pela mesma dentro de um contexto voltado ao desenvolvimento de software baseado em componentes e suas principais deficiências.

Em seguida foi realizado um estudo dos processo de desenvolvimento baseado em componentes propostos pelo *UML Components* e pela Universidade Estadual de Campinas – UNICAMP. Foram consideradas neste estudo todas as atividades descritas nos modelos relacionadas ao contexto abordado pelas áreas de processo (PAs) do CMMI-SE/SW analisadas. Foram estudadas também as áreas de processo do FAA iCMM correspondentes.

Nos estudo do modelos UML components e Processo DBC da Unicamp foram identificados práticas cabíveis para a reestruturação do CMMI, sendo estas inseridas no modelo. O estudo do modelo de capacidade FAA iCMM foi registrados através de mapeamentos entre os modelos, que permitiu definir as alterações necessárias. Todos os elementos considerados relevantes identificados nos três modelos foram representados através de uma proposta gráfica, para facilitar a visualização e a compreensão das alterações propostas.

As alterações que foram propostas têm por objetivo principal explicitar as atividades relacionadas ao desenvolvimento baseado em componentes dentro do modelo CMMI-SE/SW. Isto implica em trazer o que estava implícito no modelo CMMI-SE/SW para algo mais facilmente observável aos utilizadores do mesmo, além de adicionar aspectos não considerados anteriormente pelo modelo CMMI-SE/SW.

Por fim, estas alterações propostas foram discutidas, revisadas, alteradas, quando necessário, e transferidas em forma de texto para o modelo proposto seguindo o padrão adotado pelas áreas de processo do CMMI-SE/SW, sendo encontradas nos apêndices, A, B e C.

4. RESULTADOS E DISCUSSÃO

Neste capítulo são descrita as Áreas de Processos do CMMI relacionadas a reuso, bem como os mapeamentos realizados através da aferição entres o CMMI e os modelos de processo de desenvolvimento (UML *Components* e Processo DBC da Unicamp) e de capacidade de processo, o FAA iCMM. Por fim, são apresentadas as Áreas de Processo do CMMI com suas respectivas práticas e eventuais alterações vindas do resultado do mapeamento. .

4.1. Identificação das Áreas de Processo do CMMI

Como dito anteriormente, a partir do estudo sobre componentes e o CMMI foram identificadas três áreas de processo que mais se relacionam com o desenvolvimento para reuso. As áreas de processo que foram classificadas como relevantes são: Desenvolvimento de Requisitos, Soluções Técnicas e Integração de Produto. A seguir será dada uma breve descrição de cada uma destas áreas de processo.

4.1.1. Desenvolvimento de Requisitos

A área de processo *Requirement Development* (RD) ou desenvolvimento de requisitos tem como objetivo produzir e analisar requisitos de clientes, produtos e componentes de produtos. Esta área de processo descreve três tipos de requisitos: requisitos de clientes, requisitos de produtos e requisitos de componentes de produtos. Juntos, estes requisitos tratam as necessidades dos *stakeholders* relevantes, incluindo aquelas pertinentes às diversas fases do ciclo de vida do produto (por exemplo, critérios de testes de aceitação) e atributos do produto (por exemplo, segurança, confiabilidade e possibilidade de manutenção). Os requisitos também tratam as restrições causadas pela seleção de soluções de *design* (por exemplo, integração de produtos comerciais de prateleira).

4.1.2. Solução Técnica

A área de processo *Technical Solution* (TS) ou Solução Técnica tem como objetivo criar o *design*, desenvolver e implementar soluções para os requisitos. Soluções, *designs* e implementações englobam produtos, componentes de produtos e processos relacionados com o ciclo de vida do produto, isoladamente ou combinados, conforme apropriado.

4.1.3. Integração de Produto

A área de processo *Product Integration* (PI), ou Integração do Produto, tem como objetivos: montar o produto a partir dos componentes do produto, garantindo que o produto, quando integrado, funcione apropriadamente; e realizar a entrega do produto.

4.2. Contribuição dos Modelos de Processo de Componentes

Os modelos de processo UML *Components* e Processo de DBC Unicamp tiveram considerável contribuição no incremento das PAs identificadas. O primeiro apresentando um nível técnico extremamente detalhado, contribuindo para o CMMI no quesito referente a especificação de interface e de componentes. O segundo modelo contribui com definições arquiteturais e com suas práticas de integração e adaptação de componentes.

4.3. Mapeamento de Áreas de Processo (PAs) com o Modelo FAA iCMM

Na descrição do mapeamento das PAs relacionadas aos dois modelos de maturidade de capacidade foram considerados para objetos de demonstração apenas as metas e práticas que se correlacionaram entre os modelos.

Como poderá ser visto nas tabelas referente de mapeamento das PAs do CMMI, as metas genéricas (SG) do modelo estarão referenciadas diretamente pela metas do FAA iCMM. As práticas específicas (SP) pelas práticas básicas (BP) do FAA iCMM. Os mapeamentos estão representados nas sessões subseqüentes.

4.3.1. PA Desenvolvimento de Requisitos

A Tabela 4.1 demonstra o mapeamento realizado entre a PA do CMMI desenvolvimento de Requisitos versus as PA do FAA iCMM Necessidades e Requerimentos.

Tabela 4.1 - Mapeamento da PA Desenvolvimento de Requisitos versus PA FAA iCMM.

CMMI PA Desenvolvimento de Requisitos	FAA iCMM 01 Necessidades FAA iCCM 02 Requisitos
SG 1 Elaborar Requesitos do Cliente	PA 01 – meta 1
SP 1.1 Elicitar Necessidades	BP 01.02 Elicitar Necessidades
SG 2 Desenvolver Requisitos de Produtos	PA 02 – meta 1
SP 2.1 Estabelecer os Requisitos do Produto e dos Componentes do Produto	BP 02.03 Identificar Requisitos Chaves
SP 2.2 Alocar Requisitos de Componentes do Produto	BP 03.04 Alocar Requisitos
SP 2.3 Identificar Requisitos de Interfaces	BP 02.05 Identificar Requisitos de Interface Externa
SG 3 Analisar e Validar Requisitos	PA 02 – meta 2
SP 3.1Estabelecer Conceitos Operacionais e Cenários	BP 01.03 Analisar Necessidades
SP 3.2 Estabelecer uma Definição da Funcionalidade Exigida	BP 02.01 Identificar Funcionalidade e Requisitos de Performance
SP 3.3 Analisar Requisitos	BP 02.01 Analisar Requisitos
SP 3.5Validar Requisitos com Métodos Abrangentes	BP 08.04 Avaliar Aumento de Produtos de Trabalhos
SG 3 Analisar e Validar Requisitos	PA 02 – meta 2

Fonte: (O Autor).

4.3.2. PA Solução Técnica

A Tabela 4.2 demonstra o mapeamento realizado entre a PA do CMMI Solução Técnica versus as PA do FAA iCMM Necessidades e Requerimentos.

Tabela 4.2 - Mapeamento da PA Solução Técnica versus PAs FAA iCMM

CMMI PA Solução Técnica	FAA PA 03 Design FAA PA 06 Implementação do Desing
SG 1 Selecionar Soluções de Componentes de Produtos	PA 03 – meta 1, 2
SP 1.1 Desenvolver Soluções Alternativas Detalhadas e Critério de Seleção	BP 03.02 Desenvolver Design Estrutural
SP 1.2 Evoluir os Conceitos Operacionais e Cenário	BP 01.03 Analisar Necessidades BP 03.05 Definir Interação entre Elementos do Projeto
SP 1.3-1 Selecionar Soluções de componentes do Produto	BP 03.02 Desenvolver Design Estrutural
SG 2 Desenvolver o Design	PA 03 – meta 1
SP 2.1 Criar o Design do Produto e dos Componentes do Produto	GP 2.2 Documentar o Processo GP 2.4 Fornecer Recursos Adequados GP 2.10 Estimar Objetivos e Conformidade do Processo GP 2.14 Tomar Ações Corretivas
SP 2.2 Estabelecer um Completo Pacote de Dados Técnicos.	BP 03.08 Estabelecer e Manter a Descrição do Design BP 06.03 Elaborar Documentação
SP 2.3 Estabelecer Descrição de Interface	BP 03.03 Desenvolver Especificação de Interface
SP 2.4 Executar Análise de Fabricação, Compra ou Reutilização	BP 03.07 Estabelercer e Utilizar uma Stratégia para o Não desenvolvimentos de Itens
SG 3 Implementar Design do Produto	PA 06 – meta 1

SP 3.1 Implementar Design	BP 06.02 Formular Produtos or Serviços de Componentes
SP 3.2 Desenvolver a Documentação de Suporte ao Produto	BP 06.03 Elaborar Documentação

Fonte: (O Autor).

4.3.3. PA Integração de Produto

A Tabela 4.3 demonstra o mapeamento realizado entre a PA do CMMI Integração de Produto versus as PA do FAA iCMM Necessidades e Requerimentos.

Tabela 4.3 - Mapeamento da PA Integração de Produto versus PAs FAA iCMM.

CMMI PA Integração de Produto	FAA iCMM PA 07 Integração
SG 1 Preparar para Integração do Produto	PA 07 – meta 1
SP 1.1 Estabelecer uma Estratégia de Integração	BP 07.01 Desenvolver Estratégia de Integração
SP 1.2Estabelecer o Ambiente da Integração sp 1.2	BP 07.01 Desenvolver Estratégia de Integração
SP 1.3 Estabelecer os Procedimentos e Critérios para Integração de Produtos	BP 07.01 Desenvolver Estratégia de integração
SG 2 Assegurar a Compatibilidade das Interfaces	PA 07 – meta 1
SP 2.1 Revisar as Descrições de Interfaces com Relação à Completude	BP 07.03 Revisar e Coordenar as Definições de Interface
SP 2.2- Montar os Componentes do Produto e Entregar o Produto.	BP 07.03 Revisar e Coordenar as Definições de Interface
SG 3 Assemble Product Components and Deliver the Product	PA 07 – meta 3
SP 3.1 Confirmar que os Componentes do Produto estão Prontos para a Integração	BP 07.02 Confirmar Presteza de Produtos e Elementos de Serviço
SP 3.2 Montar os Componentes de Produtos	BP 07.04 Montar Produto e Elementos de Serviço
SP 3.3 Avaliar os Componentes de Produtos Montados	BP 07.05 Confirmar Integração de Produto ou Operação do Serviço

SP 3.4 Empacotar e Entregar o Produto ou o Componente do Produto	BP 09.05 Transição do Produto ou Serviço

Fonte: (O Autor)

4.4. Proposta de Alteração Áreas de Processo Relacionadas ou Reuso

As propostas de reformulação das PAs do CMMI tomadas a partir do estudo dos modelos de processo de desenvolvimento baseados em componentes, UML *Componentes* e Processo de DBC da Unicamp e do modelo de maturidade FAA iCMM, estão representadas nas subseções deste tópico através de organograma.

4.4.1. PA Desenvolvimento de Requisitos

A representação gráfica da PA é mostrada na Figura 4.1, onde é apontado o desmembramento da PA em suas metas genéricas (SG), práticas específicas (SP) e esta por sua vez em produtos típicos ou sub-práticas.

A principal alteração da área de processo Desenvolvimento de Requisitos foi a adição no SG1, SP 1.1, da sub-prática 'Negociar requisitos com o cliente', para refletir a ação de reaproveitamento de componentes anteriormente adquiridos ou produzidos. Através desta os requisitos passam a ser re-avaliados com o cliente levando-se em consideração vantagens de se utilizar componentes já existentes no repositório. Na mesma prática especifica foi adicionada a sub-prática 'Identificar Requisitos não funcionais' para destacar a importância deste tipo de requisitos.

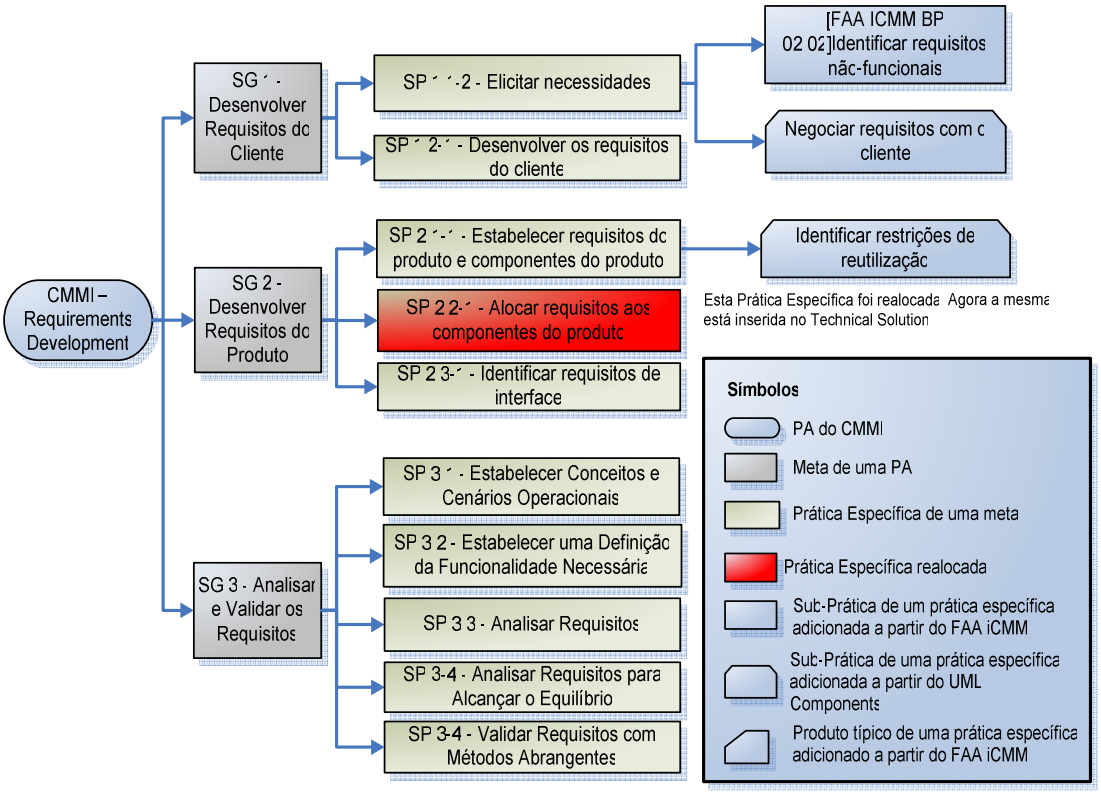

Figura 4.1 - Alteração proposta graficamente para a Área de Processo Desenvolvimento de Requisitos. Fonte: (Adaptado pelo Autor).

Os requisitos não-funcionais, apesar de não representarem funcionalidades diretamente, eles podem interferir na maneira como o sistema deve executá-las. Dessa forma, a organização estrutural do sistema, materializada na arquitetura de software, é diretamente relacionada a esses requisitos, sendo considerados como requisitos de qualidade, pois são reguladores da performance do sistema.

No SG2, SP 2.1, foi adicionada a sub-prática 'Identificar restrições de reutilização'. A prática específica 'Alocar requisitos aos componentes do produto' foi transferida para a área de processo 'Solução Técnica', por se encontrar em maior grau de conformidade com as demais práticas presentes na PA também reformulada

A Tabela 4.4 mostra a estrutura proposta para a área de processo Desenvolvimento de Requisitos (RD), refletindo as mudanças citadas.

Tabela 4.4- Estrutura proposta para a Área de Processo de Desenvolvimento de Requisitos.

PA Desenvolvimento de Requisitos
SG 1 Desenvolver Requisitos do Cliente

	SP 1.1 Elicitar Necessidades
	Sub-prática Identificar Requisitos Não Funcionais (adicionada)
	Sub-prática Negociar Requisitos com o Cliente (adicionada)
	SP 1.2 Desenvolver os Requisitos do Cliente
SG 2 Desenvolver Requisitos do Produto	
	SP 2.1 Estabelecer os Requisitos de Produtos e Componentes de Produtos
	Sub-prática Identificar Restrições de Reutilização (adicionada)
	~~SP 2.2 Alocar Requisitos aos Componentes do Produto~~ (Realocada para PA TS)
	SP 2.2 Identificar Requisitos de Interfaces
SG 3 Analisar e Validar os Requisitos	
	SP 3.1 Estabelecer Conceitos e Cenários Operacionais
	SP 3.2 Estabelecer uma Definição da Funcionalidade Necessária
	SP 3.3 Analisar Requisitos
	SP 3.4 Analisar Requisitos para Alcançar o Equilíbrio
	SP 3.5 Validar Requisitos com Métodos Abrangentes

Fonte: (O Autor).

Nesta e nas outras tabelas, os textos em itálico representam as principais alterações realizadas.

A área de processo completa e reformulada se encontra no apêndice C.

4.4.2. PA Solução Técnica

A representação gráfica da PA é mostrada na Figura 4.2 e Figura 4.3, onde é apontado o desmembramento da PA em metas genéricas (SG), práticas específicas (SP) e esta por sua vez em produtos típicos ou sub-práticas.

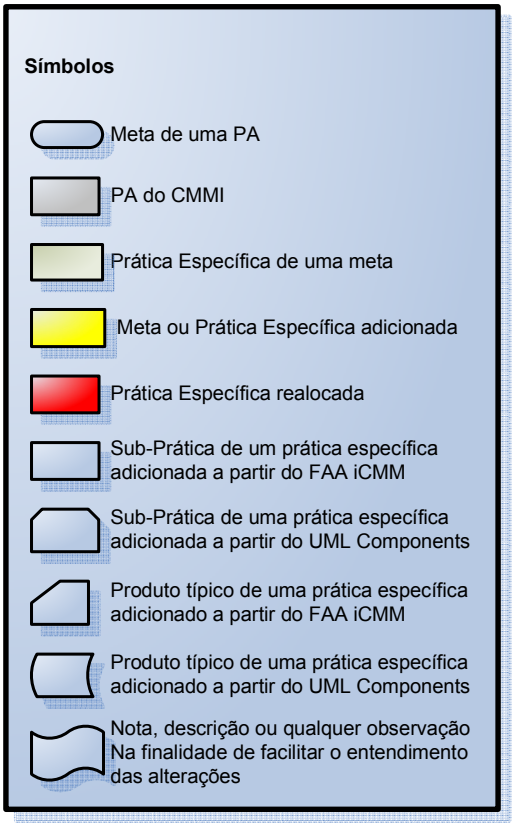

Figura 4.2 - Legenda dos símbolos do organograma da PA Solução Técnica. Fonte: (O Autor).

A principal proposta de modificação para esta área de processo foi a divisão do Objetivo específico 'Desenvolver *Design*' nas metas 'Definir Arquitetura', 'Desenvolver Especificações de Interface' e 'Desenvolver Especificações dos Componentes', com a conseqüente modificação na ordenação e numeração das SG. O objetivo principal dessas alterações foi ressaltar no modelo proposto aspectos fundamentais ao desenvolvimento de software baseado em componentes. Considerando mais detalhadamente os aspectos críticos relacionados ao projeto da arquitetura, especificação de interface e especificações dos próprios componentes.

Foram adicionadas várias sub-práticas aos objetivos específicos resultantes do desmembramento. Foi adicionada a prática específica 'Estabelecer Especificações da Arquitetura' com as sub-práticas 'Avaliar restrições arquiteturais', 'Identificar arquiteturas iniciais'.

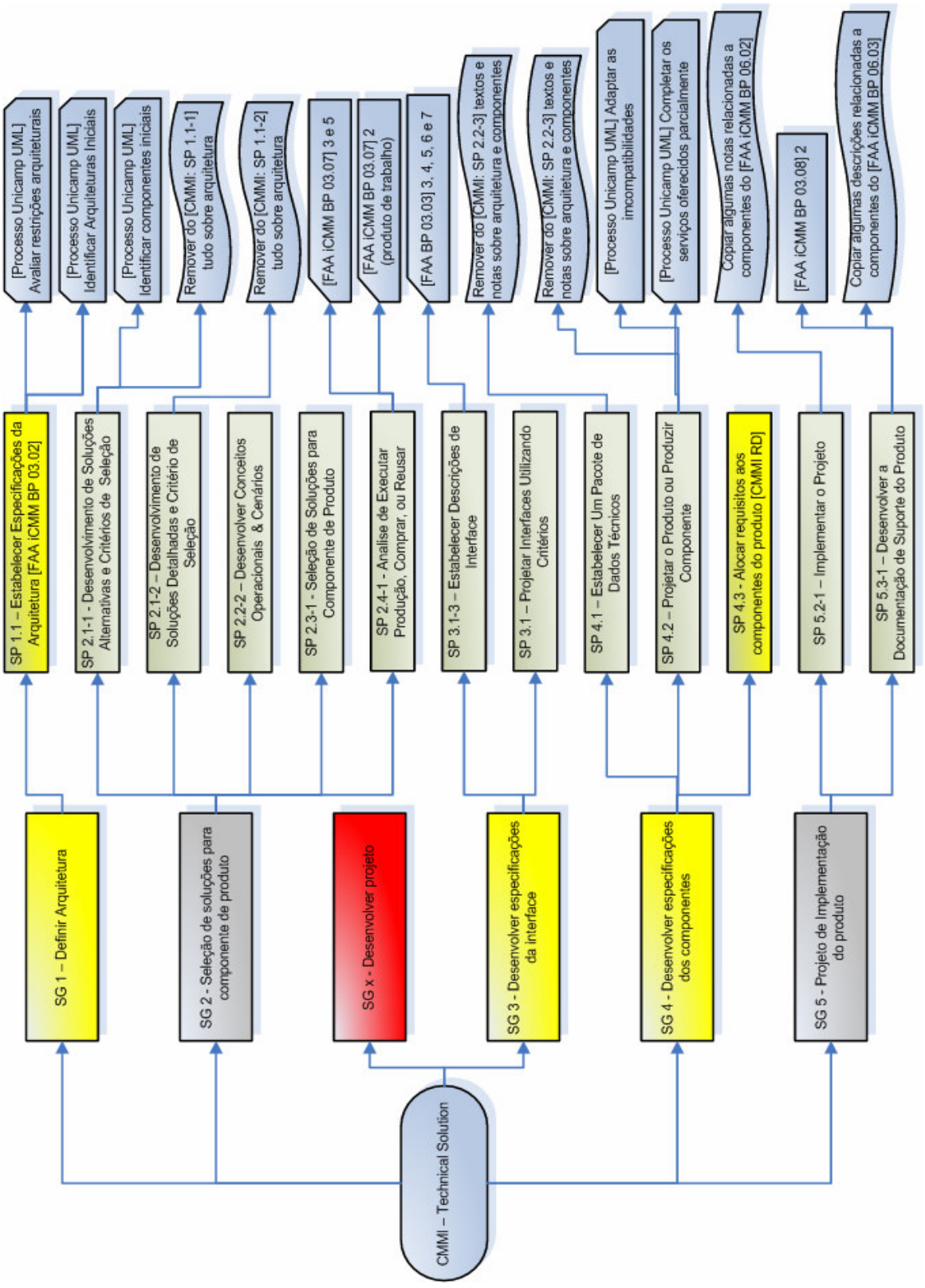

Figura 4.3 - Alteração para PA Desenvolvimento de Requisitos. Fonte: (O Autor).

A identificação de restrições arquiteturais levanta alguns aspectos a serem considerados numa fase posterior relacionada à seleção da arquitetura. Por exemplo, para a definição de um padrão arquitetural deve ser considerado um conjunto de restrições que identificam como componentes e conectores podem ser combinados

A Tabela 4.5 mostra a estrutura resultante para essa área de processo:

Tabela 4.5 - Resultado da Alteração sobre a Área de Processo Soluções Técnicas.

PA Solução Técnica		
SG 1 Definir Arquitetura (desmembrada da SG2 Desenvolver Design)		
	SP 1.1 Estabelecer Especificações da Arquitetura (adicionada)	
		Sub-Prática Avaliar Restrições Arquiteturais (adicionada)
		Sub-Prática Identificar Arquiteturas Inicias (adicionada)
SG 2 Selecionar Soluções de Componentes de Produtos (antiga SG1)		
	SP 2.1 Desenvolver Soluções Alternativas Detalhadas e Critérios de Seleção	
		Sub-Prática Identificar Componentes Iniciais (adicionada)
	SP 2.2 Evoluir Conceitos Operacionais e Cenários	
	SP 2.3 Selecionar Soluções de Componentes de Produtos	
	SP 2.4 Executar Análises de Fabricação, Compra ou Reuso (Realocada)	
		Produto Típico: Conjunto de COTS padronizados (adicionada)
		Produto Típico: Estratégia de não desenvolvimento de itens (adicionada)
		Produto Típico: Plano para evolução das versões dos produtos de prateleira
SG 3 Desenvolver Especificações de Interface (desmembrada da Meta Desenvolver Design)		
	SP 3.1 Criar o Design das Interfaces Utilizando os Critérios	
	SP 3.2 Estabelecer Descrições de Interface	
		Sub-Prática Requisitos de Interface (adicionada)
		Sub-Prática Especificação de Interface do Usuário (adicionada)
		Sub-Prática Especificação de Interface de Sub-Sistemas (adicionada)

	Sub-Prática Especificação de Interface de Componentes (adicionada)
SG 4 Desenvolver Especificações dos Componentes (desmembrada da Meta Desenvolver Design)	
	SP 4.1 Projetar o Produto ou Produzir o Componente
	Sub-Prática Completar os Serviços Oferecidos Parcialmente
	Sub-Prática Adaptar as Incompatibilidades
	~~*SP 4.2 Alocar Requisitos aos Componentes do Produto*~~ *(Realocada de RD-SP2.2)*
	SP 4.3 Estabelecer um Pacote de Dados Técnicos
SG 5 Implementar o Design do Produto	
	SP 5.1 Implementar o Design
	SP 5.2 Desenvolver a Documentação de Suporte do Produto
	Sub-Prática Gerar um Repositório de Dados do Projeto

Fonte: (O Autor).

A área de processo completa e reformulada se encontra no apêndice B.

4.4.3. PA Integração de Produtos.

A representação gráfica da PA é mostrada na Figura 4.4 , onde é apontado o desmembramento da PA em metas genéricas (SG), práticas específicas (SP) e esta por sua vez em produtos típicos ou sub-práticas.

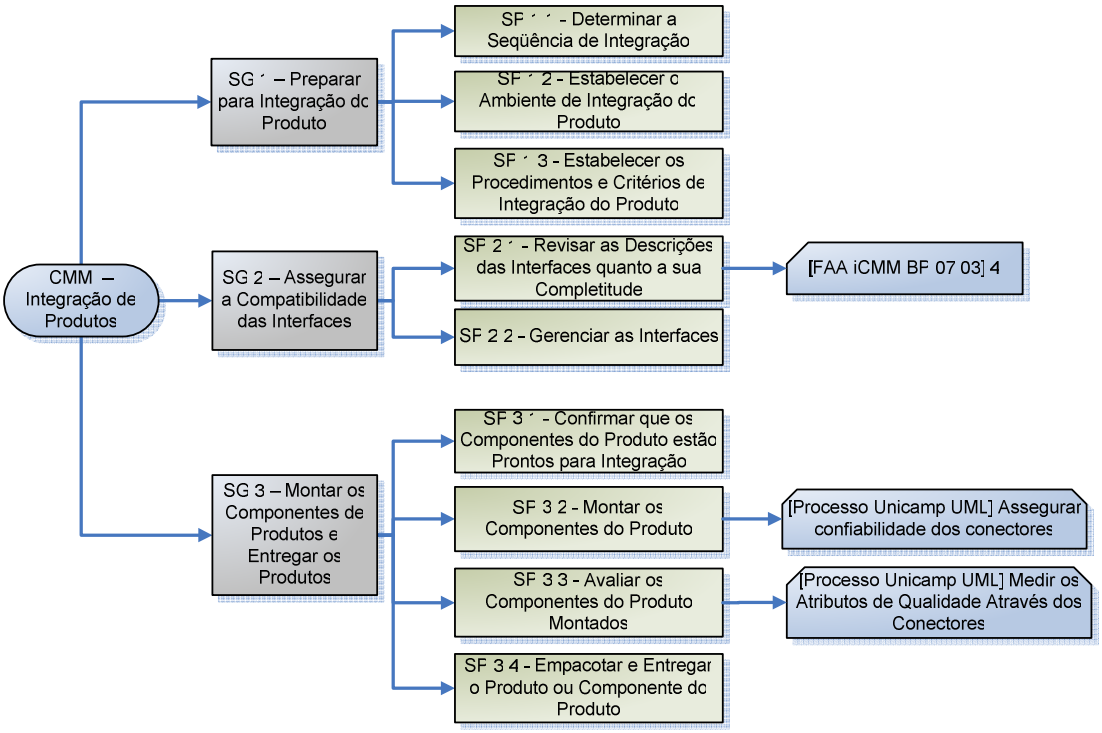

Figura 4.4 - Alteração proposta para a PA Integração de Produto. Fonte: (O Autor).

A principal alteração da área de processo Integração de Produtos foi a importância dada aos conectores. Sub-práticas e produtos típicos foram adicionados com o objetivo de dar uma maior importância aos conectores, pois estes são os únicos componentes do sistema que possuem o conhecimento de seus fluxos interativos. Dessa forma, eles são considerados os locais ideais para a implementação dos requisitos de qualidade do sistema, tais como a tolerância à falhas, distribuição e disponibilidade.

Sub-práticas: Foram adicionadas as sub-práticas ‘Assegurar confiabilidade dos conectores’ e ‘Medir os atributos de qualidade do componente através dos conectores’; Produtos Típicos: Foram adicionados os produtos típicos ‘Análise do reuso de interface de componentes’ e ‘Relatório de teste de instalação de integração’.

Como os conectores são responsáveis pela qualidade do sistema, a qualidade dos mesmos deve ser garantida assegurando que os conectores são confiáveis. O CMMI não trata apropriadamente dos conectores. Procurou-se ser dar mais ênfase aos conectores, pois eles são o elo de comunicação entre os componentes arquiteturais.

O produto típico, ‘Análise do reuso de interface de componentes’, foi adicionado para evidenciar que ao analisar as interfaces do componente, deve ser avaliada a reusabilidade do componente. De acordo com os estudos relacionados a reuso, a interface do componente está relacionada à sua reusabilidade.

A Tabela 4.6 mostra a estrutura resultante dessa área de processo.

Tabela 4.6 - Resultado da Alteração sobre a Área de Integração de Produtos.

PA Integração de Produtos	
SG 1 Preparar para a Integração do Produto	
	SP 1.1 Determinar a Seqüência de Integração
	SP 1.2 Estabelecer o Ambiente de Integração do Produto
	SP 1.3 Estabelecer os Procedimentos e Critérios de Integração do Produto
SG 2 Assegurar a Compatibilidade das Interfaces	
	SP 2.1 Revisar as Descrições das Interfaces quanto a sua Completitude
	Produto Típico Análise do reuso de interfaces de componentes (adicionada)
	SP 2.2 Gerenciar as Interfaces
SG 3 Montar os Componentes do Produto e Entregar o Produto	
	SP 3.1 Confirmar que os Componentes do Produto estão Prontos para Integração
	SP 3.2 Montar os Componentes do Produto
	Sub-Prática Assegurar Confiabilidade dos Conectores(adicionada)
	SP 3.3 Avaliar os Componentes do Produto Montados
	Sub-Prática Medir os Atributos de Qualidade Através dos Conectores (adicionada)
	SP 3.4 Empacotar e Entregar o Produto ou Componente do Produto

Fonte: (O Autor).

A área de processo completa e reformulada se encontra no apêndice C.

5. CONCLUSÕES

Neste trabalho foi descrita uma proposta de adequação do modelo de capacidade de processo, o CMMI-SE/SW v1.1 em sua versão estagiada, para o contexto específico da melhoria de processo de desenvolvimento de software baseado em componentes, sendo atacadas três áreas de processo que mais se relacionaram com o desenvolvimento para reuso, a saber: Desenvolvimento de Requisitos, Solução Técnica, Integração de Produto. Estas áreas de processos foram escolhidas por pertencerem às áreas de processo de engenharia do CMMI.

Foram utilizados os modelos de processo de desenvolvimento UML *Components,* o modelo de Processo de DBC da Unicamp e o modelo de maturidade de capacidade FAA iCMM como referências para a especialização das PAs do CMMI.

O modelo CMMI foi escolhido por ser um modelo de renome tanto na indústria mundial quanto na comunidade de qualidade e engenharia de software.

A principal alteração da área de processo Desenvolvimento de Requisitos foi a adição no SG1, SP 1.1, da sub-prática 'Negociar requisitos com o cliente', para refletir a ação de reaproveitamento de componentes anteriormente adquiridos ou produzidos. Na mesma prática especifica foi adicionada a sub-prática 'Identificar Requisitos não funcionais' para destacar a importância deste tipo de requisitos.

A proposta principal de modificação para Solução Técnica foi a divisão do Objetivo específico 'Desenvolver *Design*' nas metas 'Definir Arquitetura', 'Desenvolver Especificações de Interface' e 'Desenvolver Especificações dos Componentes', com a conseqüente modificação na ordenação e numeração das SG.

Na área de processo Integração de Produtos foi dada uma maior ênfase aos conectores. Sub-práticas e produtos típicos foram adicionados com o objetivo de dar uma maior importância aos mesmo, pois estes são os únicos componentes do sistema que possuem o conhecimento de seus fluxos interativos.

O modelo CMMI se caracteriza por ser um modelo de maturidade para processo de sistemas engenharia como um todo. Seus propositores procuraram tornar suas diretrizes extremamente genéricas para que assim contemplasse a maioria dos processos empregados nas engenharias, podendo servir como referência para maturidade dos mesmos.

A maior dificuldade identificada no trabalho foi a preocupação em se mensurar o que poderia realmente ser adicionado nas áreas de processo do CMMI, visto que dois dos modelos tomados como referência, para o incremento de das áreas de processo abordadas, tratavam-se do processo de desenvolvimento de componentes propriamente dito, indo da modelagem de componentes a um nível mais baixo, com utilização de pseudo-código. O nível de detalhamento desses modelos poderia tornar as áreas do CMMI consideravelmente complexas, dificultando o alcance pelas instituições da certificação do nível de maturidade que as áreas envolvidas pertence.

As áreas de processo reformuladas apresentadas neste trabalho são úteis para organizações desenvolvedoras de software que desejam basear o seu desenvolvimento em reuso de software, servindo como um guia para a implantação do desenvolvimento baseado em componentes.

A principal contribuição da proposta apresentada aqui para o campo de estudos da qualidade de software em geral e da melhoria de processo de software em particular, foi prover a especialização do modelo de capacidade de processo, o CMMI, para uma área, no caso desenvolvimento de software baseado em componentes, com potencial de impacto na qualidade e produtividade de software.

Em trabalhos futuro, poderá ser feita a adequação do modelo para a representação contínua, além das áreas de processo poderem ser implantadas em empresas que desenvolvam software reutilizando componentes. Após a implantação destas áreas de processo, avaliar as empresas de software de acordo com algum método de avaliação em conformidade com a CMMI-SE/SW.

6. REFERENCIAL BIBLIOGRÁFICO

(ABNT, 1994) Associação Brasileira de Normas Técnicas. **NBR ISO 8402/1994 - Gestão da qualidade e garantia da qualidade - Terminologia**. Rio de Janeiro: ABNT, 1994.

(Boehm, 1999) Boehm, B. **Managing Software Productivity and Reuse. Computer**, vol. 32, n. 9p. 111-113, set. 1999. Disponível em< http://ieeexplore.ieee.org/iel5/2/17107/00789755.pdf?isnumber=17107&arnumber=789755>. Acesso em 02/01/2007.

(Brito *et al*, 2005) Brito, P. H. da S.; Barbosa, M. A. M.; Guerra, P. A. de C.; Rubira, C. M. F. **Um Processo para o DBC com Reuso de Componentes.** Instituto de Computação / Universidade Estadual de Campinas – Campinas, SP, 2005.

(Cheesman & Daniels) Cheesman, J.; Daniels, J. **UML Components: A Simple Process for Specifying Component-Based Software** – Addison-Wesley, 2001. 176p.

(Crnkovic & Larsson, 2002) Crnkovic, I.; Larsson, M. **Challenges of component-based development.** Journal of Systems and Software Volume 61, Edição 3 , Abril de 2002, Páginas: 201-212.

(Dantas, 2005) Dantas, J. **COMPGOV – Biblioteca de Componentes pra E-gov.** In: EQPS – Encontro de Qualidade e Produtividade em Software – Programa Brasileiro de Qualidade e Produtividade em Software. 2005.

(D'Souza & Wills, 1999) D'Souza, D.; Wills, A. C. **Objects, Components, and Frameworks with UML The Catalysis Approach**. Addison-Wesley, 2nd edition, 1999.

(Garvin, 1984) Garvin, D. **What does "product quality" really means?** Sloan management review, 1984, p.25-43.

(Gerra, 2004) Gerra, P. A. de C. **Uma Abordagem Arquitetural para Tolerância a Falhas em Sistemas de Software Baseados em Componentes.** Tese de Doutorado, IC/Unicamp, Junho de 2004.

(Gil, 1991) Gil, A. C. **Como elaborar projetos de pesquisa.** 3ª Edição. São Paulo: Atlas, 1991.

(IEEE, 1990) Institute of Eletrical and Eletronics and Engineers. **IEEE Std 610.12-1990 - Standard glossary of software engineering terminology**. Piscataway: IEEE, 1990.

(Ibrahim, 2001) Ibrahim, Linda; BRADFORD, Bill; COLE, David; LABRUYERE, Larry; LEINNEWEBER, Heidi; PISZCZEK, Dave; REED, Natalie; RYMOND, Mike; SMITH, Dennis; VIRGA, Michael; WELLS, Curt (2001). **The federal aviation administration integrated capability maturity model version 2.0**: An integrated capability maturity model for enterprise-wide improvement. Published by the Federal Aviation Administration.

(ISO 8402, 1994) International Organization for Standardization. **ISO DIS 8402 - Quality Vocabulary.** Genebra, 1994.

(ISO/IEC 15504, 2003) The International Organization for Standardization and the International Electrotechnical Commission. **ISO/IEC 15504 – Information Technology - Process Assessment, International Standard (IS) with five parts**, 2003.

(ISO/IEC 15504-5, 2006) The International Organization for Standardization and the International Electrotechnical Commission. **ISO/IEC 15504 - Information Technology - Process Assessment – Part 5: An exemplar Process Assessment Model.** 2006.

(Jung, 2004) Jung, C. F. **Metodologia Para Pesquisa & Desenvolvimento: Aplicada a Novas Tecnologias, Produtos e Processos.** Rio de Janeiro. Axcel Books. 2004.

(Lego, 2006) **The LEGO Group.** Disponível em <http://www.lego.com/eng/default.aspx>. Acesso em 20/10/2006.

(Machado, 2003) Machado, C. A. F. **Definindo Processos do Ciclo de Vida de Software usando a Norma ISO/IEC 12207** – Lavras: UFLA/FAEPE 2003.

(MCT, 2004) Ministério de Ciência e Tecnologia (MCT). **CHAMADA PÚBLICA MCT/FINEP/Ação Transversal -Biblioteca de Componentes -05/2004.** Chamada Pública. FINEP / MCT. 2004.

(MCT, 2005) Ministério de Ciência e Tecnologia (MCT).. Sociedade para Promoção da Excelência do Software Brasileiro – SOFTEX e Departamento de Política Científica e Tecnológica - DPCT/UNICAMP – "**Estratégia Nacional para Componentes de Software**".

(Moreira, 2004) Moreira, R. T. **Gestão do Conhecimento em Qualidade de Software: Construção de um Portal da Qualidade de Software para o Brasil**. 2004. Monografia (Graduação em Ciência da Computação) – Universidade Federal de Lavras, Lavras. Disponível em http://www.comp.ufla.br/monografias/ano2004/ano2004.htm#segunda.

(Paulk *et al.*, 1995) Paulk, M.C.; Weber, C.V.; Curtis, B.; Chrissis, M.B.; *et al.* **The Capability Maturity Model: Guidelines for Improving the Software Process**. Estados Unidos: Addison-Wesley. 1995.

(Pessôa, 2004) Pessôa, M. S. P. **Introdução ao CMMI: Modelo Integrado de Maturidade da Capacidade do Processo.** – Lavras: UFLA/FAEPE 2004.

(PMI, 2004) *Project Management Institute.* **Um Guia do Conjunto de Conhecimentos em Gerenciamento de Projetos (Guia PMBOK®)** Terceira edição 2004 Project Management Institute, Four Campus Boulevard, Newtown Square, PA 19073-3299 EUA.

(Rossi, 2004) ROSSI, A. C. **Representação do Componente de software na FARCSOFT: Ferramenta de apoio à reutilização de componentes de software**. Dissertação de mestrado. Poli/USP, 2004.

(Rouiller, 2001) Rouiller, A. C. **Gerenciamento de Projetos de Software para Empresas de Pequeno Porte.** Tese de Doutorado – Recife, MG: UFPE 2001.

(Rouiller *et al*, 2004) Rouiller, A. C.; Vasconcelos, A. M. L. de; Maciel, T. M. de M. **Engenharia de Software** – Lavras: UFLA/FAEPE, 2004.

(Salviano, 2006) Salviano, C. F. **Melhoria e Avaliação de Processo de Software com o Modelo ISO/IEC 15504-5: 2006.** – Lavras: UFLA/FAEPE 2006.

(SEI, 2002) Software Engineering Institute. **Capability Maturity Model® Integration (CMMI^{SM}), Version 1.1. CMMI^{SM} for Software Engineering (CMMI-SW, V1.1), Staged Representation.** Pittsburgh, PA, EUA: Carnegie Mellon University 2002.

(SEI, 2005) *Software Engineering Institute*. **Frequently Asked Questions (FAQs).** . *Software Engineering Institute*. 2005. Disponível em < http://www.sei.cmu.edu/cmmi/faq/ps-faq.html#Q172>, acessado em 3/7/2006.

(Short, 1997) SHORT, K. **Component Based Development and Object Modeling**. Sterling Software. 1997. Disponível em <http://www.cool.sterling.com>. Acesso em 10/04/2006.

(Silva, 2006) SILVA, L. P. **Adaptação dos Processos do Grupo de Reuso a ISO/IEC 15504-5 para as Áreas de Processo do CMMI**. 2006. Monografia (Graduação em Ciência da Computação) – Universidade Federal de Lavras, Lavras. Disponível em < http://www.comp.ufla.br/monografias/ano2006/ano2006.htm >. Acesso em 03/10/2006.

(Softex, 2005) SOFTEX (2005) MPS.BR –Ministério de Ciência e Tecnologia – Melhoria de Processo de Software Brasileiro – Guia Geral (versão 1.0).

(Sommerville, 2003) Sommerville, I. **Engenharia de Software**. 6 ª ed. São Paulo, Addison Wesley, 2003.

(Souza, 2004) SOUZA, A. D. **Estudo e avaliação da área de processo de planejamento de projeto de acordo com o modelo CMMI-SW Nível 2 na empresa SWQuality situada em Lavras-MG**. 2004. Monografia (Graduação em Ciência da Computação) – Universidade Federal de Lavras, Lavras. Disponível em <http://www.comp.ufla.br/monografias/ano2004/ano2004.htm>. Acesso em 03/03/2006.

(Szyperski, 2002) SZYPERSKI, C. **Component Software – Beyond Object-Oriented Programming**. Second Edition. 2002

(Tsukumo, 1997) Tsukumo, A. N.; Rêgo, C. M.; Salviano, C. F.; Azevedo, G. F.; Meneghetti, L. K.; Costa, M. C. C.; Carvalho, M. B.; Colombo, R. M. T. **Qualidade de Software: Visões de Produto e Processo de Software**. In: II Escola Regional de

Informática da Sociedade Brasileira de Computação Regional de São Paulo - II ERI da SBC. Piracicaba, SP - Junho de 1997, pags: 173 – 189.

(Vasconcelos, 2003) Vasconcelos, A. M. L. de. **Introdução a Engenharia de Software e aos Princípios da Qualidade** – Lavras: UFLA/FAEPE 2003.

(Vidger, 1996) Vidger, M.R.; Gentleman, M.W.; Dean, J. **COTS Software Integration: State of the Art.** NRC-CNRC Report, National Research Council, Canada, Jan. 1996.

(Villas Boas, 2003) Villas Boas, A. L. C. **Gestão de Configuração para Teste de Software.** Dissertação de mestrado apresentada na Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas. Campinas, SP: [s.n.], 2003.

Apêndice A – PA Desenvolvimento de Requisitos

DESENVOLVIMENTO DE REQUISITOS

Nível de Maturidade 3

Objetivo

O objetivo do Desenvolvimento de Requisitos (Requirements Development) é produzir e analisar requisitos de clientes, produtos e componentes de produtos. [PA157]

Notas Introdutórias

Esta área de processo descreve três tipos de requisitos: requisitos de clientes, requisitos de produtos e requisitos de componentes de produtos. Juntos, estes requisitos tratam as necessidades dos stakeholders relevantes, incluindo aquelas pertinentes às diversas fases do ciclo de vida do produto (por exemplo, critérios de testes de aceitação) e atributos do produto (por exemplo, segurança, confiabilidade e possibilidade de manutenção). Os requisitos também tratam as restrições causadas pela seleção de soluções de design (por exemplo, integração de produtos comerciais de prateleira). [PA157.N101]

Os requisitos são a base do design. O desenvolvimento de requisitos inclui as seguintes atividades: [PA157.N102]

- Elicitação, análise, validação e comunicação das necessidades dos clientes, expectativas e restrições para obter requisitos de clientes que constituem um entendimento do que satisfará os stakeholders
- Coleta e coordenação das necessidades dos stakeholders
- Desenvolvimento dos requisitos do ciclo de vida do produto
- Estabelecimento dos requisitos de clientes
- Estabelecimento dos requisitos iniciais de produtos e de componentes de produtos consistentes com os requisitos de clientes

Esta área de processo trata todos os requisitos de clientes, e não somente os requisitos de produto, porque o cliente pode também fornecer requisitos específicos de design. [PA157.N103]

Os requisitos de clientes são melhor refinados em requisitos de produtos e de componentes de produtos. Além dos requisitos de clientes, requisitos de produtos e de componentes de produtos são derivados a partir das soluções de design selecionadas. [PA157.N104]

Os requisitos são identificados e refinados em todas as fases do ciclo de vida do produto. As decisões de design, ações corretivas subseqüentes e feedback durante cada fase do ciclo de vida do produto são analisados com relação ao impacto nos requisitos derivados e alocados. [PA157.N105]

A área de processo de Desenvolvimento de Requisitos inclui três metas específicas. A meta específica Desenvolver Requisitos de Clientes trata da definição de um conjunto de requisitos do cliente para uso no desenvolvimento dos requisitos de produtos. A meta específica Desenvolver Requisitos de Produtos trata da definição de um conjunto de requisitos de produtos ou de componentes de produtos para uso no design de produtos e componentes de produtos. A meta específica Analisar e Validar Requisitos trata da necessária análise dos requisitos de clientes, produtos e componentes de produtos para definir, derivar e entender os requisitos. As práticas específicas da terceira meta específica pretendem auxiliar as práticas específicas das duas primeiras metas específicas. Os processos associados com a área de processo de Desenvolvimento de Requisitos e aqueles associados com a área de processo de Soluções Técnicas podem interagir recursivamente uns com os outros. [PA157.N111]

As análises são utilizadas para entender, definir e selecionar os requisitos em todos os níveis a partir de alternativas. Estas análises incluem: [PA157.N106]

- Análise das necessidades e requisitos para cada fase do ciclo de vida do produto, incluindo as necessidades dos stakeholders relevantes, o ambiente operacional e fatores que refletem as expectativas e satisfação geral do cliente e dos usuários finais, como segurança, proteção e preço
- Desenvolvimento de um conceito operacional
- Definição da funcionalidade exigida

A definição da funcionalidade, também chamada de “análise funcional”, não é a mesma análise estruturada do desenvolvimento de software e não presume um design de software orientado a funcionalidades. No design de software orientado a objetos, ela se relaciona com a definição dos serviços. A definição de funções, seus agrupamentos lógicos e sua associação com requisitos é chamada de “arquitetura funcional”. [PA157.N107]

As análises ocorrem recursivamente em camadas sucessivas, cada vez mais detalhadas, da arquitetura de um produto até que detalhes suficientes estejam disponíveis para possibilitar o design detalhado, aquisição e teste do produto a ser executado. Como resultado da análise de requisitos e do conceito operacional (incluindo funcionalidade, suporte, manutenção e descarte), o conceito de manufatura ou produção produz requisitos mais derivados, incluindo as seguintes considerações: [PA157.N108]

- Restrições de diversos tipos
- Limitações tecnológicas
- Custo e direcionadores de custo

- Restrições de tempo e direcionadores de cronograma
- Riscos
- Consideração de questões citadas, mas não explicitamente declaradas pelo cliente ou usuário final
- Fatores introduzidos por considerações únicas de negócios do desenvolvedor, regulamentações e leis

Uma hierarquia de entidades lógicas (funções e subfunções, classes e subclasses de objetos) é estabelecida através de iterações com o conceito operacional em evolução. Os requisitos são refinados, derivados e alocados a estas entidades lógicas. Os requisitos e entidades lógicas são alocados a produtos, componentes de produtos, pessoas, processos associados ou serviços. [PA157.N109]

O envolvimento dos stakeholders relevantes no desenvolvimento e análise de requisitos lhes dá a visibilidade da evolução dos requisitos. Esta atividade continuamente assegura a eles que os requisitos estão sendo apropriadamente definidos. [PA157.N110]

Áreas de Processos Relacionadas

Veja a área de processo de Gerenciamento de Requisitos para obter maiores informações sobre o gerenciamento de requisitos de clientes e produtos, obtenção de acordos com o fornecedor dos requisitos, obtenção de compromissos com quem está implementando os requisitos e a manutenção da rastreabilidade. [PA157.R101]

Veja a área de processo de Soluções Técnicas para obter maiores informações sobre como as saídas dos processos de desenvolvimento de requisitos são utilizadas e como o desenvolvimento de soluções e designs alternativos é utilizado no refinamento e derivação de requisitos. [PA157.R102]

Veja a área de processo de Integração de Produtos para obter maiores informações sobre requisitos de interface e gerenciamento de interfaces. [PA157.R103]

Veja a área de processo de Verificação para obter maiores informações sobre como verificar se o produto resultante atende seus requisitos. [PA157.R104]

Veja a área de processo de Validação para obter maiores informações sobre como o produto construído será validado contra as necessidades do cliente. [PA157.R105]

Veja a área de processo de Gerenciamento de Riscos para obter maiores informações sobre a identificação e gerenciamento de riscos que estão relacionados aos requisitos. [PA157.R106]

Veja a área de processo de Gerenciamento de Configurações para obter maiores informações sobre como assegurar que os produtos de trabalho chave estão sendo controlados e gerenciados. [PA157.R107]

Metas Específicas e Genéricas

SG 1 **Desenvolver Requisitos do Cliente** [PA157.IG101]

As necessidades, expectativas, restrições e interfaces dos stakeholders são coletadas e traduzidas em requisitos do cliente.

SG 2 **Desenvolver Requisitos de Produtos** [PA157.IG103]

Os requisitos de clientes são refinados e elaborados para desenvolver requisitos de produtos e componentes de produtos.

SG 3 **Analisar e Validar os Requisitos** [PA157.IG102]

Os requisitos são analisados e validados e uma definição da funcionalidade exigida é desenvolvida.

GG 3 **Institucionalizar um Processo Definido** [CL104.GL101]

O processo é institucionalizado como um processo definido.

Tabela de Relacionamentos Práticas-Metas

SG 1 Desenvolver Requisitos do Cliente [PA157.IG101]
- SP 1.1 Elicitar Necessidades
- SP 1.2 Desenvolver os Requisitos do Cliente

SG 2 Desenvolver Requisitos do Produto [PA157.IG103]
- SP 2.1 Estabelecer os Requisitos de Produtos e Componentes de Produtos
- SP 2.2 Identificar Requisitos de Interfaces

SG 3 Analisar e Validar os Requisitos [PA157.IG102]
- SP 3.1 Estabelecer Conceitos e Cenários Operacionais
- SP 3.2 Estabelecer uma Definição da Funcionalidade Necessária
- SP 3.3 Analisar Requisitos
- SP 3.4 Analisar Requisitos para Alcançar o Equilíbrio
- SP 3.5 Validar Requisitos com Métodos Abrangentes

GG 3 Institucionalizar um Processo Definido [CL104.GL101]
- GP 2.1 (CO 1) Estabelecer uma Política Organizacional
- GP 3.1 (AB 1) Estabelecer um Processo Definido
- GP 2.2 (AB 2) Planejar o Processo
- GP 2.3 (AB 3) Fornecer Recursos
- GP 2.4 (AB 4) Atribuir Responsabilidades
- GP 2.5 (AB 5) Treinar as Pessoas
- GP 2.6 (DI 1) Gerenciar Configurações
- GP 2.7 (DI 2) Identificar e Envolver os Stakeholders Relevantes
- GP 2.8 (DI 3) Monitorar e Controlar o Processo
- GP 3.2 (DI 4) Coletar Informações de Melhorias
- GP 2.9 (VE 1) Avaliar Objetivamente a Aderência
- GP 2.10 (VE 2) Revisar o Status com o Nível Mais Alto de Gerência

SG 1 Desenvolver Requisitos de Clientes

As necessidades, expectativas, restrições e interfaces dos stakeholders são coletadas e traduzidas em requisitos de clientes. *[PA157.IG101]*

As necessidades dos stakeholders (por exemplo, clientes, usuários finais, fornecedores, construtores e testadores) são a base para determinar os requisitos de clientes. As necessidades dos stakeholders, expectativas, restrições, interfaces, conceitos operacionais e conceitos do produto são analisados, harmonizados, refinados e elaborados para serem traduzidos em um conjunto de requisitos de clientes. [PA157.IG101.N101]

Muitas vezes, as necessidades, expectativas, restrições e interfaces dos stakeholders são pobremente identificadas ou são conflitantes. Uma vez que as necessidades, expectativas, restrições e limitações dos stakeholders deverão estar claramente identificadas e entendidas, um processo iterativo é usado durante toda a vida do projeto para atingir este objetivo. Para facilitar a necessária interação, um substituto do usuário final ou cliente é freqüentemente envolvido para representar suas necessidades e ajudar a resolver conflitos. A área de relações com o cliente ou de marketing da organização, assim como membros da equipe de desenvolvimento de disciplinas como a de engenharia humana ou suporte podem ser utilizados como substitutos. Restrições ambientais, legais e outras deverão ser consideradas quando da criação e resolução do conjunto de requisitos do cliente. [PA157.IG101.N102]

SP 1.1 Elicitar Necessidades

Elicitar as necessidades, expectativas, restrições e interfaces dos stakeholders para todas as fases do ciclo de vida do produto. *[PA157.IG101.SP102]*

Elicitar vai além de coletar requisitos, envolvendo identificar de forma pró-ativa requisitos adicionais que não foram explicitamente fornecidos pelos clientes. Os requisitos adicionais deverão tratar as diversas atividades do ciclo de vida do produto e seu impacto no produto. [PA157.IG101.SP102.N102]

Exemplos de técnicas para elicitar necessidades incluem: [PA157.IG101.SP102.N103]

- Demonstrações tecnológicas
- Grupos de trabalho de controle de interfaces
- Grupos de trabalho de controle técnico
- Revisões intermediárias do projeto
- Questionários, entrevistas e cenários operacionais obtidos dos usuários finais
- Walkthroughs operacionais e análise de tarefas de usuários finais
- Protótipos e modelos
- Brainstorming
- Implementação de Funções de Qualidade (Quality Function Deployment)
- Pesquisas de mercado
- Testes beta
- Extração de fontes como documentos, padrões ou especificações
- Observação de produtos existentes, ambientes e padrões de fluxo de trabalho
- Use cases
- Análises de casos de negócios
- Engenharia reversa (para produtos legados)

Sub-práticas

1. Engajar os stakeholders relevantes utilizando métodos para elicitar as necessidades, expectativas, restrições e interfaces externas. [PA157.IG101.SP102.SubP101]
2. Identificar requisitos não-funcionais, também conhecidos como atributos de qualidade, que quantificam determinados aspectos do comportamento do sistema.
3. Negociar requisitos com o cliente, considerando os componentes já existentes em algum repositório.

   Na finalidade de um maior re-aproveitamento dos ativos reutilizáveis, os requisitos devem ser negociados com o cliente. Desta forma, aspectos de reutilização são considerados desde a fase de elicitação dos requisitos.

SP 1.2 Desenvolver os Requisitos de Clientes

Transformar as necessidades, expectativas, restrições e interfaces dos stakeholders em requisitos de clientes. *[PA157.IG101.SP103]*

As várias entradas vindas do cliente devem ser consolidadas, as informações que faltarem devem ser obtidas e os conflitos devem ser resolvidos na documentação do conjunto reconhecido de requisitos de clientes. Os requisitos de clientes podem incluir necessidades, expectativas e restrições com relação a verificação e validação. [PA157.IG101.SP103.N101]

Produtos de Trabalho Típicos

1. Requisitos de clientes [PA157.IG101.SP103.W101]

2. Restrições de clientes na condução da verificação [PA157.IG101.SP103.W102]
3. Restrições de clientes na condução da validação [PA157.IG101.SP103.W103]

Sub-práticas

1. Traduzir as necessidades, expectativas, restrições e interfaces dos stakeholders em requisitos de clientes documentados. [PA157.IG101.SP103.SubP101]
2. Definir restrições para verificação e validação. [PA157.IG101.SP103.SubP102]

SG 2 Desenvolver Requisitos de Produtos

Os requisitos de clientes são refinados e elaborados para desenvolver requisitos de produtos e de componentes de produtos. *[PA157.IG103]*

Os requisitos de clientes são analisados em conjunto com o desenvolvimento do conceito operacional para derivar conjuntos de requisitos mais detalhados e precisos chamados de "requisitos de produtos e de componentes de produtos". Os requisitos de produtos e de componentes de produtos tratam as necessidades associadas com cada fase do ciclo de vida do produto. Requisitos derivados provêm das restrições, consideração das questões citadas mas não explicitamente declaradas na baseline de requisitos de clientes e fatores introduzidos pela arquitetura selecionada, o design e as considerações únicas de negócios dos desenvolvedores. Os requisitos são reexaminados com cada nível sucessivo mais baixo do conjunto de requisitos e arquitetura funcional e o conceito preferencial do produto é refinado. [PA157.IG103.N101]

Os requisitos são alocados a funções de produtos e componentes de produtos incluindo objetos, pessoas e processos. A rastreabilidade dos requisitos a funções, objetos, testes, questões e outras entidades é documentada. Os requisitos e funções alocados são a base para a síntese da solução técnica. Conforme os componentes internos são desenvolvidos, interfaces adicionais são definidas e requisitos de interfaces são estabelecidos. [PA157.IG103.N102]

Veja a prática específica Manter a Rastreabilidade Bidirecional dos Requisitos da área de processo de Gerenciamento de Requisitos para obter maiores informações sobre a manutenção da rastreabilidade bidirecional. *[PA157.IG103.N102.R101]*

SP 2.1 Estabelecer Requisitos de Produtos e de Componentes de Produtos

Estabelecer e manter os requisitos de produtos e componentes de produtos, que são baseados nos requisitos de clientes. *[PA157.IG103.SP101]*

Os requisitos de clientes podem ser expressos nos termos do cliente e podem ser descrições não técnicas. Os requisitos de produtos são a expressão destes requisitos em termos técnicos que podem ser utilizados para decisões de design. Um exemplo desta tradução é encontrado na primeira Casa de Implementação de Qualidade Funcional (The first House of Quality Functional Deployment), que mapeia os desejos do cliente em parâmetros técnicos. Por exemplo, "uma porta sólida e sonora" pode ser mapeada com o tamanho, peso, formato, umidade e freqüências de ressonância. [PA157.IG103.SP101.N101]

Requisitos de produtos e de componentes de produtos tratam a satisfação dos clientes, negócios e objetivos do projeto e atributos associados, como eficiência e custo. [PA157.IG103.SP101.N104]

Restrições de design incluem especificações sobre componentes de produtos que são derivados de decisões de design, e não de requisitos de nível mais elevado. [PA157.IG103.SP101.N102]

Para Engenharia de Software

Por exemplo, os componentes da aplicação que devem ter interface com componentes de bancos de dados "de prateleira" devem obedecer aos requisitos de interface impostos pelo banco de dados selecionado. Tais requisitos de componentes de produtos não são, geralmente, rastreáveis a níveis de requisitos mais elevados. *[PA157.IG103.SP101.N102.AMP101]*

Requisitos derivados também tratam o custo e o desempenho de outras fases do ciclo de vida (por exemplo, produção, operação e descarte) na medida compatível com os objetivos de negócios. [PA157.IG103.SP101.N103]

A modificação dos requisitos devido a mudanças de requisitos é coberta pela função de "manutenção" desta prática específica, enquanto a administração de mudanças de requisitos é coberta pela área de processo de Gerenciamento de Requisitos. [PA157.IG103.SP101.N105]

Veja a área de processo de Gerenciamento de Requisitos para obter maiores informações sobre gerenciamento de mudanças de requisitos. *[PA157.IG103.SP101.N105.R101]*

Produtos de Trabalho Típicos

1. Requisitos derivados [PA157.IG103.SP101.W101]
2. Requisitos de produtos [PA157.IG103.SP101.W102]
3. Requisitos de componentes de produtos [PA157.IG103.SP101.W103]

Sub-práticas

1. Desenvolver requisitos nos termos técnicos necessários para o design dos produtos e componentes de produtos. [PA157.IG103.SP101.SubP101]

   Desenvolver os requisitos de arquitetura trata das qualidades críticas do produto e do desempenho necessário para o design da arquitetura do produto. [PA157.IG103.SP101.SubP101.N101]

2. Derivar requisitos que resultam de decisões de design. [PA157.IG103.SP101.SubP102]

   Veja a área de processo de Soluções Técnicas para obter maiores informações sobre o desenvolvimento de soluções que geram requisitos derivados adicionais. *[PA157.IG103.SP101.SubP102.R101]*

   A seleção de uma tecnologia traz requisitos adicionais. Por exemplo, o uso de eletrônica exige requisitos tecnológicos específicos adicionais, tais como os limites da interferência eletromagnética. [PA157.IG103.SP101.SubP102.N101]

3. Estabelecer e manter os relacionamentos entre os requisitos para consideração durante o gerenciamento de mudanças e alocação de requisitos. [PA157.IG103.SP101.SubP103]

   Veja a área de processo de Gerenciamento de Requisitos para obter maiores informações sobre a manutenção da rastreabilidade dos requisitos. *[PA157.IG103.SP101.SubP103.R101]*

   Relacionamentos entre os requisitos podem auxiliar na avaliação do impacto das mudanças. [PA157.IG103.SP101.SubP103.N101]

4. Identificar restrições arquiteturais.

   A identificação de restrições arquiteturais levanta alguns aspectos a serem considerados numa fase posterior relacionada à seleção da arquitetura. Por exemplo, para a definição de um padrão arquitetural deve ser considerado um conjunto de restrições que identificam como componentes e conectores podem ser combinados.

SP 2.2 Identificar Requisitos de Interface

Identificar requisitos de interface. *[PA157.IG103.SP103]*

Interfaces entre funções (ou entre objetos) são identificadas. As interfaces funcionais podem direcionar o desenvolvimento de soluções alternativas descritas na área de processo de Soluções Técnicas. [PA157.IG103.SP103.N101]

Veja a área de processo de Integração de Produtos para obter maiores informações sobre o gerenciamento de interfaces e a integração de produtos e componentes de produtos. *[PA157.IG103.SP103.N101.R101]*

Requisitos de interfaces entre produtos e componentes de produtos identificados na arquitetura do produto são definidos. Eles são controlados como parte da integração do produto e dos componentes do produto e são parte integral da definição da arquitetura. [PA157.IG103.SP103.N102]

Produtos de Trabalho Típicos

1. Requisitos de interfaces [PA157.IG103.SP103.W101]

Sub-práticas

1. Identificar interfaces externas e internas ao produto (isto é, entre partes funcionais ou objetos). [PA157.IG103.SP103.SubP101]

Conforme o design progride, a arquitetura do produto será alterada pelos processos de soluções técnicas, criando novas interfaces entre os componentes de produtos e componentes externos ao produto. [PA157.IG103.SP103.SubP101.N101]

As interfaces com os processos do ciclo de vida relacionado ao produto deverão também ser identificadas. [PA157.IG103.SP103.SubP101.N102]

Exemplos destas interfaces incluem interfaces com equipamento de teste, sistemas de transporte, sistemas de suporte e recursos de fabricação. [PA157.IG103.SP103.SubP101.N103]

2. Desenvolver os requisitos para as interfaces identificadas. [PA157.IG103.SP103.SubP102]

Veja a área de processo de Soluções Técnicas para obter maiores informações sobre a geração de novas interfaces durante o processo de design. *[PA157.IG103.SP103.SubP102.R101]*

Os requisitos das interfaces são definidos em termos de características de origem, destino, estímulo e de dados para o software e características elétricas e mecânicas para o hardware. [PA157.IG103.SP103.SubP102.N102]

SG 3 Analisar e Validar os Requisitos

Os requisitos são analisados e validados e uma definição da funcionalidade necessária é desenvolvida. *[PA157.IG102]*

As práticas específicas da meta específica Analisar e Validar Requisitos suportam o desenvolvimento dos requisitos nas metas específicas Desenvolver Requisitos do Cliente e Desenvolver Requisitos de Produtos. As práticas específicas associadas com esta meta específica cobrem a análise e validação dos requisitos com relação ao ambiente pretendido para o usuário. [PA157.IG102.N104]

As análises são executadas para determinar o impacto que o ambiente operacional pretendido terá na capacidade de satisfazer as necessidades, expectativas, restrições e interfaces dos stakeholders. Considerações sobre a facilidade de construção, necessidades da missão, restrições de custo, tamanho potencial do mercado e estratégia de aquisição devem ser levadas em conta, dependendo do contexto do produto. Uma definição da funcionalidade exigida é também estabelecida. Todos os modos de utilização especificados para o produto são considerados e uma análise de tempo é gerada para o sequenciamento de funções criticamente dependentes do tempo. [PA157.IG102.N101]

Os objetivos das análises são determinar os requisitos candidatos para os conceitos de produtos que satisfarão as necessidades, expectativas e restrições dos stakeholders e, então, traduzir estes conceitos em requisitos. Em paralelo com esta atividade, os parâmetros que serão utilizados para avaliar a eficiência do produto são determinados, com base nas entradas do cliente e no conceito preliminar do produto. [PA157.IG102.N102]

Os requisitos são validados para aumentar a probabilidade do produto resultante funcionar como previsto no ambiente de uso. [PA157.IG102.N103]

SP 3.1 Estabelecer Conceitos e Cenários Operacionais

Estabelecer e manter conceitos e cenários operacionais associados. *[PA157.IG102.SP101]*

Veja a área de processo de Soluções Técnicas para obter maiores informações sobre o desenvolvimento detalhado de conceitos operacionais que dependem dos designs selecionados. [PA157.IG102.SP101.R101]

Um cenário é uma seqüência de eventos que podem ocorrer no uso do produto, que é utilizado para tornar explícitas algumas das necessidades dos stakeholders. Em contraste, um conceito operacional para um produto normalmente depende da solução de design e do cenário. Por exemplo, o conceito operacional para um produto de comunicação por satélite é bastante diferente de outro baseado em meridianos. Uma vez que as soluções alternativas normalmente ainda não foram definidas quando se está preparando os conceitos operacionais iniciais, as soluções conceituais são desenvolvidas para serem utilizadas quando da análise dos requisitos. Os conceitos operacionais são refinados conforme as decisões de soluções são tomadas e requisitos detalhados de nível mais baixo são desenvolvidos. [PA157.IG102.SP101.N101]

Da mesma forma que uma decisão de design para um produto pode se tornar um requisito para os componentes do produto, o conceito operacional pode se tornar o cenário (requisitos) para os componentes do produto. [PA157.IG102.SP101.N102]

Os cenários podem incluir seqüências operacionais, desde que essas seqüências sejam uma expressão dos requisitos do cliente e não conceitos operacionais. [PA157.IG102.SP101.N103]

Produtos de Trabalho Típicos

1. Conceito operacional [PA157.IG102.SP101.W101]
2. Conceitos de instalação do produto, operacionais, de manutenção e de suporte [PA157.IG102.SP101.W102]
3. Conceitos de descarte [PA157.IG102.SP101.W103]
4. Use cases [PA157.IG102.SP101.W104]
5. Cenários dependentes do tempo [PA157.IG102.SP101.W105]
6. Novos requisitos [PA157.IG102.SP101.W106]

Sub-práticas

1. Desenvolver conceitos operacionais e cenários que incluem a funcionalidade, desempenho, manutenção, suporte e descarte, conforme apropriado. [PA157.IG102.SP101.SubP101]

Identificar e desenvolver cenários, consistentes com o nível de detalhes nas necessidades, expectativas e restrições dos stakeholders, sob os quais o produto proposto deverá operar. [PA157.IG102.SP101.SubP101.N101]

2. Definir o ambiente no qual o produto irá operar, incluindo limites e restrições. [PA157.IG102.SP101.SubP102]
3. Revisar os conceitos operacionais e cenários para refinar e descobrir requisitos. [PA157.IG102.SP101.SubP103]

   O desenvolvimento de conceitos operacionais e cenários é um processo iterativo. As revisões deverão ocorrer periodicamente para assegurar que estes atendem os requisitos. A revisão pode tomar a forma de um walkthrough. [PA157.IG102.SP101.SubP103.N101]
4. Desenvolver um detalhado conceito operacional, conforme os produtos e componentes de produtos são selecionados, que define a interação do produto, usuário final e ambiente e que satisfaz as necessidades operacionais, de manutenção, de suporte e de descarte. [PA157.IG102.SP101.SubP104]

SP 3.2 Estabelecer uma Definição da Funcionalidade Exigida

Estabelecer e manter uma definição da funcionalidade exigida. *[PA157.IG102.SP102]*

A definição da funcionalidade, também chamada de "análise funcional", é a descrição do que o produto deve fazer. A descrição da funcionalidade pode incluir ações, seqüência, entradas, saídas e outras informações que transmitem a maneira pela qual o produto será utilizado. [PA157.IG102.SP102.N101]

A análise funcional não é a mesma coisa que a análise estruturada do desenvolvimento de software e não presume um design de software orientado a funcionalidades. No design de software orientado a objetos, ela se relaciona com a definição dos serviços. A definição das funções, seus agrupamentos lógicos e suas associações com os requisitos são referenciados como uma arquitetura funcional. Veja a definição de "arquitetura funcional" no Apêndice B, o glossário. [PA157.IG102.SP102.N102]

Produtos de Trabalho Típicos

1. Arquitetura funcional [PA157.IG102.SP102.W101]
2. Diagramas de atividades e use cases [PA157.IG102.SP102.W102]
3. Análise orientada a objetos com os serviços identificados [PA157.IG102.SP102.W103]

Sub-práticas

1. Analisar e quantificar as funcionalidades exigidas pelos usuários finais. [PA157.IG102.SP102.SubP101]
2. Analisar os requisitos para identificar particionamentos lógicos ou funcionais (por exemplo, subfunções). [PA157.IG102.SP102.SubP102]

3. Particionar os requisitos em grupos com base em critérios estabelecidos (por exemplo, funcionalidades semelhantes, desempenho ou acoplamento), para facilitar e concentrar a análise de requisitos. [PA157.IG102.SP102.SubP103]
4. Considerar a seqüência das funções críticas de tempo, tanto no momento inicial como durante o desenvolvimento dos componentes do produto. [PA157.IG102.SP102.SubP104]
5. Alocar os requisitos de clientes às partições funcionais, objetos, pessoas ou elementos de suporte para suportar a síntese de soluções. [PA157.IG102.SP102.SubP105]
6. Alocar requisitos funcionais e de desempenho a funções e subfunções. [PA157.IG102.SP102.SubP106]

SP 3.3 Analisar os Requisitos

Analisar os requisitos para assegurar que eles são necessários e suficientes. *[PA157.IG102.SP103]*

À luz do conceito operacional e dos cenários, os requisitos para um nível de hierarquia de produto são analisados para determinar se eles são necessários e suficientes para atender os objetivos dos níveis mais elevados da hierarquia do produto. Os requisitos analisados, então, fornecem a base para requisitos mais precisos e detalhados para os nível mais baixos da hierarquia do produto. [PA157.IG102.SP103.N102]

Conforme os requisitos são definidos, seus relacionamentos com requisitos e funcionalidades de mais alto nível devem ser entendidos. Uma outra ação é a determinação dos requisitos chave que serão utilizados para rastrear o progresso técnico. Por exemplo, o peso de um produto ou tamanho de um produto de software podem ser monitorados através do desenvolvimento baseado no seu risco. [PA157.IG102.SP103.N101]

Produtos de Trabalho Típicos

1. Relatórios de defeitos em requisitos [PA157.IG102.SP103.W101]
2. Mudanças propostas nos requisitos para resolver defeitos [PA157.IG102.SP103.W102]
3. Requisitos chave [PA157.IG102.SP103.W103]
4. Medidas de desempenho técnico [PA157.IG102.SP103.W104]

Sub-práticas

1. Analisar as necessidades, expectativas e interfaces externas dos stakeholders para remover conflitos e organizá-las em assuntos relacionados. [PA157.IG102.SP103.SubP101]
2. Analisar requisitos para determinar se eles satisfazem os objetivos de requisitos de nível mais alto. [PA157.IG102.SP103.SubP102]
3. Analisar requisitos para assegurar que eles são completos, possíveis de serem executados, percebidos e verificados. [PA157.IG102.SP103.SubP103]

Embora o design determine a possibilidade de se construir uma solução específica, esta sub-prática trata da descoberta de quais requisitos afetam esta possibilidade. [PA157.IG102.SP103.SubP103.N101]

4. Identificar requisitos chave que têm uma forte influência no custo, cronograma, funcionalidade, risco ou desempenho. [PA157.IG102.SP103.SubP104]

5. Identificar medidas de desempenho técnico que serão rastreadas durante o esforço de desenvolvimento. [PA157.IG102.SP103.SubP105]

   Veja a área de processo de Medições e Análises para obter maiores informações sobre o uso de medições. [PA157.IG102.SP103.SubP105.R101]

6. Analisar os conceitos operacionais e cenários para refinar as necessidades, restrições e interfaces dos clientes e descobrir novos requisitos. [PA157.IG102.SP103.SubP106]

   Esta análise pode resultar em conceitos operacionais e cenários mais detalhados, bem como suportar a derivação de novos requisitos. [PA157.IG102.SP103.SubP106.N101]

SP 3.4 Analisar os Requisitos para Alcançar o Equilíbrio

Analisar os requisitos para equilibrar as necessidades e as restrições dos stakeholders. *[PA157.IG102.SP104]*

As necessidades dos stakeholders e as restrições podem tratar de custos, cronograma, desempenho, funcionalidade, componentes reusáveis, possibilidade de manutenção e riscos. [PA157.IG102.SP104.N102]

Produtos de Trabalho Típicos

1. Análise de riscos relacionados a requisitos [PA157.IG102.SP104.W101]

Sub-práticas

1. Usar modelos comprovados, simulações e protótipos para analisar o equilíbrio entre as necessidades dos stakeholders e as restrições. [PA157.IG102.SP104.SubP103]

   Os resultados das análises podem ser usados para reduzir o custo do produto e o risco do desenvolvimento do produto. [PA157.IG102.SP104.SubP103.N101]

2. Executar uma análise dos riscos sobre os requisitos e a arquitetura funcional. [PA157.IG102.SP104.SubP101]

   Veja a área de processo de Gerenciamento de Riscos para obter maiores informações sobre como executar uma análise de riscos sobre os requisitos de clientes e do produto e a arquitetura funcional. [PA157.IG102.SP104.SubP101.R101]

3. Examinar os conceitos do ciclo de vida do produto com relação ao impacto dos requisitos nos riscos. [PA157.IG102.SP104.SubP102]

SP 3.5 Validar os Requisitos com Métodos Abrangentes

Validar os requisitos para assegurar que o produto resultante funcionará como previsto no ambiente do usuário utilizando diversas técnicas, conforme necessário. *[PA157.IG102.SP106]*

A validação dos requisitos é executada antecipadamente ao esforço de desenvolvimento para se adquirir a confiança de que os requisitos são capazes de guiar um desenvolvimento que resultará numa validação final bem sucedida. Esta atividade deverá estar integrada com as atividades de gerenciamento de riscos. Organizações maduras normalmente farão a validação dos requisitos de forma mais sofisticada e irão ampliar a base de validação para incluir outras necessidades e expectativas dos stakeholders. Estas organizações geralmente farão análises, simulações ou protótipos para assegurar que os requisitos satisfarão as necessidades e expectativas dos stakeholders. [PA157.IG102.SP106.N102]

Produtos de Trabalho Típicos

1. Registro dos métodos e resultados da análise [PA157.IG102.SP106.W101]

Sub-práticas

1. Analisar os requisitos para determinar o risco do produto resultante não funcionar apropriadamente no seu ambiente de uso previsto.
2. Explorar a adequação e a completitude dos requisitos através do desenvolvimento de representações do produto (por exemplo, protótipos, simulações, modelos, cenários e storyboards) e da obtenção de feedback sobre eles dos stakeholders relevantes. []
3. Analisar o design, conforme ele amadurece, no contexto do ambiente de validação dos requisitos para identificar questões de validação e expor necessidades e requisitos de clientes não declarados.

Apêndice B - PA Soluções Técnicas

SOLUÇÕES TÉCNICAS

Nível de Maturidade 3

Objetivo

O objetivo das Soluções Técnicas (Technical Solution) é criar o design, desenvolver e implementar soluções para os requisitos. Soluções, designs e implementações englobam produtos, componentes de produtos e processos relacionados com o ciclo de vida do produto, isoladamente ou combinados, conforme apropriado. [PA160]

Notas Introdutórias

A área de processo de Soluções Técnicas é aplicável em qualquer nível da arquitetura do produto e a todos os produtos, componentes de produtos, processos relacionados com o ciclo de vida do produto e serviços. A área de processo se concentra no seguinte: [PA160.N101]

- Avaliar e selecionar soluções (algumas vezes referenciadas como “abordagens de design”, “conceitos de design” ou “designs preliminares”) que potencialmente satisfazem um conjunto apropriado de requisitos alocados
- Desenvolver designs detalhados para as soluções selecionadas (detalhados no contexto de conter todas as informações necessárias para a construção, codificação ou outro tipo de implementação do design como um produto ou componente de produto).
- Implementar os designs como um produto ou componente de produto

Geralmente, estas atividades suportam interativamente umas às outras. Algum nível de design, às vezes moderadamente detalhado, pode ser necessário para selecionar soluções. Protótipos de componentes de produtos podem ser utilizados como meios de obter um conhecimento suficiente para desenvolver um pacote de dados técnicos ou um conjunto completo de requisitos. [PA160.N102]

As práticas específicas de Soluções Técnicas se aplicam não somente ao produto e componentes do produto, mas também aos serviços e processos relacionados com o ciclo de vida do produto. Os processos relacionados com o ciclo de vida do produto são desenvolvidos conjuntamente com o produto ou componente do produto. Tal desenvolvimento pode incluir selecionar e adaptar processos existentes (incluindo processos padrão) para o uso, bem como desenvolver novos processos. [PA160.N103]

Os processos associados com a área de processo de Soluções Técnicas recebem os requisitos do produto e dos componentes de produtos dos processos de gerenciamento de requisitos. Os processos de gerenciamento de requisitos colocam os requisitos, que se originam dos processos de desenvolvimento de requisitos, sob o gerenciamento de configurações apropriado e mantêm sua rastreabilidade aos requisitos anteriores. [PA160.N104]

Para uma organização de manutenção ou sustentação, os requisitos que precisam de ações de manutenção ou redesign podem ser direcionados pelas necessidades dos usuários ou defeitos latentes nos componentes do produto. Novos requisitos podem surgir de mudanças no ambiente operacional. Tais requisitos podem ser descobertos durante a verificação de produto(s), onde o desempenho real possa ser comparado contra o desempenho especificado e uma degradação inaceitável possa ser identificada. Os processos associados com a área de processo de Soluções Técnicas deverão ser utilizados para executar os esforços de manutenção ou sustentação do design. [PA160.N105]

Áreas de Processos Relacionadas

Veja a área de processo de Desenvolvimento de Requisitos para obter maiores informações sobre alocação de requisitos, estabelecimento de um conceito operacional e definição de requisitos de interface. *[PA160.R101]*

Veja a área de processo de Verificação para obter maiores informações sobre a condução de revisões por pares e verificação se o produto e os componentes do produto atendem os requisitos. *[PA160.R102]*

Veja a área de processo de Análise de Decisões e Resoluções para obter maiores informações sobre avaliação formal. *[PA160.R103]*

Veja a área de processo de Gerenciamento de Requisitos para obter maiores informações sobre o gerenciamento de requisitos. As práticas específicas da área de processo de Gerenciamento de Requisitos são executadas de forma interativa com as da área de processo de Soluções Técnicas. *[PA160.R104]*

Veja a área de processo de Inovação e Desenvolvimento Organizacional para obter maiores informações sobre como melhorar a tecnologia da organização. *[PA160.R105]*

Metas Específicas e Genéricas

SG 1 **Definir Arquitetura**

Avaliar as alternativas considerando os critérios estabelecidos para selecionar a arquitetura.

SG 2 **Selecionar Soluções de Componentes de Produtos** [PA160.IG101]

Soluções de produtos ou componentes de produtos são selecionadas a partir de soluções alternativas.

SG 3 **Desenvolver as Especificações da Interface**

A especificação da interface é desenvolvida.

SG 4 **Desenvolver as Especificações dos Componentes**

A especificação dos componentes é desenvolvida.

SG 5 **Implementar o Design do Produto** [PA160.IG103]

Os componentes de produto e documentação de suporte associada são implementados a partir de seus designs.

GG 3 **Institucionalizar um Processo Definido** [CL104.GL101]

O processo é institucionalizado como um processo definido.

Tabela de Relacionamentos Práticas-Metas

SG 1 Definir Arquitetura
- SP 1.1 Estabelecer Especificações da Arquitetura

SG 2 Selecionar Soluções de Componentes de Produtos [PA160.IG101]
- SP 1.1 Desenvolver Soluções Alternativas Detalhadas e Critérios de Seleção
- SP 1.2 Evoluir Conceitos Operacionais e Cenários
- SP 1.3 Selecionar Soluções de Componentes de Produtos
- SP 1.4 Executar Análises de Construção, Compra ou Reuso

SG 3 Desenvolver o Design
- SP 3.1 Criar o Design das Interfaces Utilizando os Critérios

SG 4 Desenvolver Especificações dos Componentes
- SP 4.1 Estabelecer um Pacote de Dados Técnicos
- SP 4.2 Projetar o Produto ou Produzir o Componente
- SP 4.2 Alocar Requisitos aos Componentes do Produto

SG 5 Implementar o Design do Produto [PA160.IG103]
- SP 5.1 Implementar o Design

SP 5.2 Desenvolver a Documentação de Suporte do Produto

GG 3 Institucionalizar um Processo Definido [CL104.GL101]

GP 2.1	(CO 1)	Estabelecer uma Política Organizacional
GP 3.1	(AB 1)	Estabelecer um Processo Definido
GP 2.2	(AB 2)	Planejar o Processo
GP 2.3	(AB 3)	Fornecer Recursos
GP 2.4	(AB 4)	Atribuir Responsabilidades
GP 2.5	(AB 5)	Treinar as Pessoas
GP 2.6	(DI 1)	Gerenciar Configurações
GP 2.7	(DI 2)	Identificar e Envolver os Stakeholders Relevantes
GP 2.8	(DI 3)	Monitorar e Controlar o Processo
GP 3.2	(DI 4)	Coletar Informações de Melhorias
GP 2.9	(VE 1)	Avaliar Objetivamente a Aderência
GP 2.10	(VE 2)	Revisar o Status com o Nível Mais Alto de Gerência

Práticas Específicas por Meta

SG 1 Definir Arquitetura

Avaliar as alternativas considerando os critérios estabelecidos para selecionar a arquitetura.

O primeiro nível de decomposição do projeto define a arquitetura e o nível mais baixo do projeto estabelece a especificação do projeto por meio da construção do ou aquisição do produto ou elemento de serviço.

SP 1.1 Estabelecer Especificações da Arquitetura

Estabelecimento das especificações da arquitetura

Veja a área de processo de Desenvolvimento de Requisitos para obter maiores informações sobre o desenvolvimento de requisitos de arquitetura. *[PA160.IG102.SP101.N101.R101]*

A definição da arquitetura é feita a partir de um conjunto de requisitos de arquitetura, desenvolvidos durante os processos de desenvolvimento de requisitos. Estes requisitos expressam as qualidades e pontos de desempenho que são críticos para o sucesso do produto. A arquitetura define os elementos estruturais e mecanismos de coordenação que satisfazem diretamente os requisitos ou suportam o atendimento dos requisitos, conforme os detalhes do design do produto são estabelecidos. As arquiteturas podem incluir padrões e regras de design que governam o desenvolvimento de componentes de produtos e suas interfaces, bem como um direcionamento para auxiliar os desenvolvedores do produto. As práticas específicas da meta específica Selecionar Soluções de Componentes de Produtos contêm maiores informações sobre como utilizar a arquitetura do produto como base para as soluções alternativas. [PA160.IG102.SP101.N102]

Os arquitetos postulam e desenvolvem um modelo do produto, tomando decisões sobre a alocação de requisitos a componentes do produto, incluindo hardware e software. Diversas arquiteturas, suportando soluções alternativas, podem ser desenvolvidas e analisadas para definir as vantagens e desvantagens no contexto dos requisitos de arquitetura. [PA160.IG102.SP101.N103]

Os conceitos operacionais e cenários são utilizados para gerar use cases e cenários de qualidade que são utilizados para refinar a arquitetura. Eles também são utilizados como meios para avaliar a adequação da arquitetura com relação ao seu objetivo pretendido, durante as avaliações de arquitetura que são conduzidas periodicamente durante o design do produto. A prática específica Evoluir Conceitos Operacionais e Cenários traz maiores informações sobre como elaborar conceitos operacionais e cenários utilizados na avaliação da arquitetura. [PA160.IG102.SP101.N104]

Veja a prática específica Estabelecer Conceitos Operacionais e Cenários da área de processo de Desenvolvimento de Requisitos para obter maiores informações sobre como desenvolver conceitos operacionais e cenários utilizados na avaliação da arquitetura. [PA160.IG102.SP101.N104.R101]

Para Engenharia de Software

Além das tarefas identificadas acima, a definição da arquitetura de software pode incluir: [PA160.IG102.SP101.N104.AMP101]

- *Estabelecer as relações estruturais de partições e regras com relação a interfaces entre elementos dentro de partições e entre partições*
- *Identificar as principais interfaces internas e todas as interfaces externas do software*
- *Identificar componentes de produtos de software*
- *Definir mecanismos de coordenação de software*
- *Estabelecer capacidades e serviços de infra-estrutura*
- *Desenvolver templates de componentes de produtos ou classes e frameworks*
- *Estabelecer regras de design e autoridade para a tomada de decisões*
- *Definir um processo/modelo a ser seguido*
- *Definir a implantação física do software no hardware*
- *Identificar principais abordagens e fontes de reuso*

Produtos de Trabalho Típicos

1. Arquitetura do produto [PA160.IG102.SP101.W101]

Sub-práticas

1. Estabelecer e manter critérios para selecionar a arquitetura.

   Os critérios para seleção da arquitetura incluem muitos fatores. Como exemplo de itens a serem considerados durante o estabelecimento de critérios de seleção temos: o custo do ciclo de vida de um produto ou serviço (custo inicial, operação, manutenção, remoção); performance técnica ou complexidade de serviço ou produto.

Veja a área de processo de Desenvolvimento de Requisitos para obter maiores informações sobre como identificar restrições arquiteturais.

2. Avaliar restrições arquiteturais

   Avaliar as restrições quanto a aspectos arquiteturais, ou atributos de qualidade.

3. Identificar arquiteturas iniciais

   Projetar a arquitetura inicial a partir de uma lista de arquiteturas compatíveis, das quais podem ser selecionadas as arquiteturas candidatas e posteriormente a mais adequada.

SG 2 Selecionar Soluções de Componentes de Produto

Soluções de produtos ou componentes de produtos são selecionadas a partir de soluções alternativas. *[PA160.IG101]*

As soluções alternativas e seus méritos relativos são considerados antes de selecionar uma solução. Requisitos chave, questões de design e restrições são estabelecidos para o uso na análise de soluções alternativas. Recursos de arquitetura que fornecem uma fundação para a melhoria e evolução do produto são considerados. O uso de componentes de produto "de prateleira" é considerado com relação ao custo, cronograma, desempenho e riscos. As alternativas de prateleira podem ser utilizadas com ou sem modificações. Algumas vezes, tais itens podem exigir modificações em aspectos como interfaces ou uma customização de alguns de seus recursos para melhor atender os requisitos de produtos. [PA160.IG101.N101]

Um indicador de um bom processo de design é que o design foi escolhido após ter sido comparado e avaliado contra soluções alternativas. Decisões sobre arquitetura, desenvolvimento customizados versus compra de produto de prateleira e modularização de componentes de produtos são as opções típicas de design que são tratadas. [PA160.IG101.N102]

Algumas vezes, a pesquisa por soluções examina instâncias alternativas dos mesmos requisitos, sem precisar de alocações a um nível mais baixo de componentes do produto. Este é o caso na base da arquitetura do produto. Existem também casos onde uma ou mais soluções são estabelecidas (por exemplo, uma solução específica é direcionada ou componentes de produtos disponíveis, como os de prateleira, são examinados para serem utilizados). [PA160.IG101.N103]

No caso geral, as soluções são definidas como um conjunto. Isto é, quando se está definindo a próxima camada de componentes de produtos, a solução para cada componente de produto do conjunto é estabelecida. As soluções alternativas são, não somente diferentes maneiras de se tratar os mesmos requisitos, mas elas também refletem uma diferente alocação de requisitos entre os componentes de produto que compõem o conjunto da solução. O objetivo é otimizar o conjunto como um todo e não as peças individuais. Existirá uma interação significativa com os processos associados com a área de processo de Desenvolvimento de Requisitos para suporte às alocações provisionais de componentes de produtos, até que um conjunto de soluções seja selecionado e as alocações finais estabelecidas. [PA160.IG101.N104]

Processos relacionados com o ciclo de vida do produto estão entre as soluções de componentes de produtos que são selecionadas entre as soluções alternativas. Exemplos destes processos relacionados com o ciclo de vida do produto estão os processos de construção e de suporte. [PA160.IG101.N105]

SP 2.1 Desenvolver Soluções Alternativas Detalhadas e Critérios de Seleção

Desenvolver soluções alternativas detalhadas e critérios de seleção. *[PA160.IG101.SP102]*

Veja a área de processo de Análises de Decisões e Resoluções para obter maiores informações sobre como estabelecer critérios utilizados na tomada de decisões. *[PA160.IG101.SP102.R101]*

As soluções alternativas detalhadas são um conceito essencial da área de processo de Soluções Técnicas. Elas fornecem informações mais precisas e abrangentes sobre a solução que alternativas não detalhadas. Por exemplo, a caracterização do desempenho baseada no conteúdo do design, em lugar de uma simples estimativa possibilita uma efetiva avaliação e entendimento dos impactos do ambiente e do conceito operacional. Soluções alternativas precisam ser identificadas e analisadas para possibilitar a seleção de uma solução balanceada durante toda a vida do produto em termos de custo, cronograma e desempenho técnico. Estas soluções se baseiam nas arquiteturas de produto propostas que tratam das qualidades críticas do produto. As práticas específicas associadas com a meta específica Desenvolver o Design fornecem maiores informações sobre o desenvolvimento de arquiteturas potenciais de produtos que podem ser incorporadas em soluções alternativas para o produto. [PA160.IG101.SP102.N104]

Soluções alternativas estendem a faixa aceitável de custo, cronograma e desempenho. Os requisitos de componentes de produtos são recebidos e utilizados com as questões de design, restrições e critérios para desenvolver as soluções alternativas. Os critérios de seleção normalmente tratarão de custos (por exemplo, tempo, pessoas, dinheiro), benefícios (por exemplo, desempenho, capacidade, eficiência) e riscos (por exemplo, técnicos, de custo, de cronograma). Considerações sobre as soluções alternativas detalhadas e os critérios de seleção incluem: [PA160.IG101.SP102.N102]

- Custo (desenvolvimento, compras, suporte, ciclo de vida do produto)
- Desempenho técnico
- Complexidade dos componentes do produto e dos processos relacionados com o ciclo de vida do produto
- Robustez das condições de operação e uso do produto, modos de operação, ambientes e variações nos processos relacionados com o ciclo de vida do produto
- Expansão e crescimento do produto
- Limitações tecnológicas

- Sensibilidade aos métodos e materiais de construção
- Riscos
- Evolução dos requisitos e tecnologia
- Descarte
- Capacitação e limitações dos usuários finais e operadores

As considerações listadas acima são um conjunto básico; as organizações deverão desenvolver critérios de avaliação para restringir a lista de alternativas que são consistentes com seus objetivos de negócios. Custo do ciclo de vida do produto, embora seja um parâmetro que se deseja minimizar, pode estar fora do controle das organizações de desenvolvimento. Um cliente pode não desejar pagar por recursos que custam mais no curto prazo, mas que fazem o custo ao longo da vida do produto decrescer. Em tais casos, os clientes deverão, pelo menos, ser avisados de potenciais reduções dos custos do ciclo de vida. Os critérios utilizados na seleção de soluções finais deverão fornecer uma abordagem balanceada de custos, benefícios e riscos. [PA160.IG101.SP102.N103]

Produtos de Trabalho Típicos

1. Critérios de avaliação de soluções alternativas [PA160.IG101.SP102.W103]
2. Avaliações de novas tecnologias [PA160.IG101.SP102.W104]
3. Soluções alternativas [PA160.IG101.SP102.W101]
4. Critérios de seleção para a seleção final [PA160.IG101.SP102.W102]

Sub-práticas

1. Identificar critérios de avaliação para selecionar um conjunto de soluções alternativas para serem consideradas. [PA160.IG101.SP102.SubP101]
2. Identificar tecnologias atualmente em uso e novas tecnologias de produto para uma vantagem competitiva. [PA160.IG101.SP102.SubP102]

   Veja a área de processo de Inovação e Desenvolvimento Organizacional para obter maiores informações sobre como melhorar a tecnologia da organização. [PA160.IG101.SP102.SubP102.R101]

   O projeto deverá identificar as tecnologias aplicadas aos atuais produtos e processos e monitorar o progresso das tecnologias atualmente utilizadas, ao longo da vida do projeto. O projeto deverá identificar, selecionar, avaliar e investir em novas tecnologias para conseguir vantagens competitivas. As soluções alternativas poderiam incluir tecnologias recém-desenvolvidas, mas também poderiam incluir a aplicação de tecnologias maduras em aplicações diferentes ou manter os métodos atuais. [PA160.IG101.SP102.SubP102.N101]
3. Gerar soluções alternativas. [PA160.IG101.SP102.SubP103]
4. Obter uma completa alocação de requisitos para cada alternativa. [PA160.IG101.SP102.SubP104]
5. Desenvolver os critérios para seleção da melhor solução alternativa. [PA160.IG101.SP102.SubP105]

Critérios deverão ser incluídos para tratar questões de design para a vida do produto, como provisões para inserir mais facilmente novas tecnologias ou a capacidade de melhor explorar produtos comerciais. Exemplos incluem critérios relacionados a conceitos de design e arquitetura abertos para as alternativas que estão sendo avaliadas. [PA160.IG101.SP102.SubP105.N101]

6. Desenvolver cenários ao longo do tempo para a operação do produto e interação do usuário para cada solução alternativa. [PA160.IG101.SP102.SubP106]

7. Identificar componentes iniciais.

SP 2.2 Evoluir os Conceitos Operacionais e Cenários

Evoluir o conceito operacional, cenários e ambientes para descrever as condições, modos de operação e estados operacionais para cada componente do produto. *[PA160.IG101.SP103]*

Veja a prática específica Estabelecer Conceitos Operacionais e Cenários da área de processo de Desenvolvimento de Requisitos para obter maiores informações sobre influências e implicações no nível do produto das operações de componentes de produtos. *[PA160.IG101.SP103.R101]*

Para Engenharia de Sistemas

Integrar os conceitos operacionais e cenários produzidos por diversos indivíduos e grupos para cada nível de decomposição física do produto. *[PA160.IG101.SP103.AMP101]*

Deve-se evoluir os conceitos operacionais e cenários para facilitar a seleção de soluções de componentes de produtos que, quando implementadas, satisfarão o uso pretendido do produto. Conceitos operacionais e cenários documentam a interação dos componentes de produtos com o ambiente, usuários e outros componentes de produtos, independente da disciplina de engenharia. Eles deverão ser documentados para operação, implementação do produto, entrega, suporte (incluindo manutenção e sustentação), treinamento e descarte e para todos os modos e estados. [PA160.IG101.SP103.N101]

Os ambientes (operacional, de suporte, de treinamento, etc.) também precisam evoluir. O ambiente de um dado componente de produto será influenciado por outros componentes de produtos, assim como pelo ambiente externo. [PA160.IG101.SP103.N102]

Produtos de Trabalho Típicos

1. Conceitos operacionais de componentes de produtos, cenários e ambientes para todos os processos relacionados com o ciclo de vida do produto (por exemplo, operação, suporte, treinamento, construção, implementação, campo, entrega e descarte) [PA160.IG101.SP103.W101]

2. Análises de tempo para as interações de componentes de produtos [PA160.IG101.SP103.W102]

3. Use cases [PA160.IG101.SP103.W104]

Sub-práticas

1. Evoluir os conceitos operacionais e cenários a um grau apropriado de detalhes para o componente de produto. [PA160.IG101.SP103.SubP101]

2. Evoluir os ambientes operacionais para os componentes de produtos. [PA160.IG101.SP103.SubP102]

   Os ambientes podem incluir elementos térmicos, stress, eletromagnéticos e outros que precisem ser documentados. [PA160.IG101.SP103.SubP102.N101]

SP 2.3 Selecionar Soluções de Componentes de Produtos

Selecionar as soluções de componentes de produtos que melhor satisfazem os critérios estabelecidos. *[PA160.IG101.SP104]*

Veja as práticas específicas Alocar Requisitos de Componentes de Produtos e Identificar Requisitos de Interfaces, da área de processo de Desenvolvimento de Requisitos para obter maiores informações sobre como estabelecer os requisitos alocados para os componentes de produtos e os requisitos de interface entre os componentes do produto. *[PA160.IG101.SP104.R101]*

Veja a área de processo de Análises de Decisões e Resoluções para obter maiores informações sobre avaliações formais. *[PA160.IG101.SP104.R102]*

Selecionar componentes de produtos que melhor satisfazem os critérios estabelece as alocações de requisitos aos componentes de produtos. Requisitos de mais baixo nível são gerados a partir da alternativa selecionada e utilizados para desenvolver o design dos componentes do produto. Os requisitos de interface entre os componentes do produto são descritos, inicialmente apenas funcionalmente. Descrições de interfaces físicas são incluídas na documentação para a interface com itens e atividades externos ao produto. [PA160.IG101.SP104.N101]

A descrição das soluções e a justificativa para a seleção são documentadas. A documentação evolui ao longo do desenvolvimento, conforme as soluções e designs detalhados são desenvolvidos e estes designs implementados. Manter um registro das justificativas é crítico para a tomada das decisões seguintes. Tais registros impedem que os stakeholders seguintes da cadeia de decisões tenham um retrabalho e fornecem entendimentos sobre como aplicar a tecnologia, conforme ela se torna disponível, em circunstâncias aplicáveis. [PA160.IG101.SP104.N102]

Produtos de Trabalho Típicos

1. Decisões e justificativa da seleção de componentes de produtos [PA160.IG101.SP104.W101]

2. Relacionamentos documentados entre requisitos e componentes de produtos [PA160.IG101.SP104.W102]

3. Soluções, avaliações e justificativas documentadas [PA160.IG101.SP104.W103]

Sub-práticas

1. Avaliar cada solução/conjunto de soluções alternativos contra os critérios de seleção estabelecidos no contexto dos conceitos operacionais, modos de operação e estados operacionais. [PA160.IG101.SP104.SubP101]

2. Com base na avaliação das alternativas, analisar a adequação dos critérios de seleção e atualizar estes critérios, conforme necessário. [PA160.IG101.SP104.SubP102]

3. Identificar e resolver questões com as soluções alternativas e requisitos. [PA160.IG101.SP104.SubP103]

4. Selecionar o melhor conjunto de soluções alternativas que satisfazem os critérios de seleção estabelecidos. [PA160.IG101.SP104.SubP104]

5. Estabelecer os requisitos associados com o conjunto selecionado de alternativas como sendo o conjunto de requisitos alocados àqueles componentes de produtos. [PA160.IG101.SP104.SubP105]

6. Identificar as soluções de componentes de produtos que serão reusadas ou adquiridas. [PA160.IG101.SP104.SubP107]

   Veja a área de processo de Gerenciamento de Acordos com Fornecedores para obter maiores informações sobre como adquirir produtos e componentes de produtos. [PA160.IG101.SP104.SubP107.R101]

7. Estabelecer e manter a documentação das soluções, avaliações e justificativas. [PA160.IG101.SP104.SubP106]

SP 2.4 Executar Análises de Fabricação, Compra ou Reuso

Avaliar se os componentes do produto deverão ser desenvolvidos, comprados ou reutilizados, com base nos critérios estabelecidos. *[PA160.IG102.SP106]*

A determinação de quais produtos ou componentes de produtos serão adquiridos é freqüentemente chamada de "análise de fabricação ou compra" (*make-or-buy analysis*). Esta é baseada em uma análise das necessidades do projeto. Esta análise de fabricação ou compra começa bem cedo no projeto durante a primeira iteração de design, continua durante o processo de design e se completa com a decisão de desenvolver, adquirir ou reutilizar o produto. [PA160.IG102.SP106.N103]

Veja a área de processo de Desenvolvimento de Requisitos para obter maiores informações sobre como determinar os requisitos de produtos e componentes de produtos. [PA160.IG102.SP106.N103.R101]

Veja a área de processo de Gerenciamento de Requisitos para obter maiores informações sobre como gerenciar requisitos. [PA160.IG102.SP106.N103.R102]

Fatores que afetam a decisão de fabricação ou compra incluem: [PA160.IG102.SP106.N104]

- Funções que os produtos ou serviços fornecerão e como estas funções se encaixam no projeto
- Recursos e habilidades disponíveis para o projeto
- Custos de aquisição versus desenvolvimento interno
- Datas críticas de entrega e integração
- Alianças estratégicas de negócios, incluindo requisitos de negócios de alto nível
- Pesquisa de mercado dos produtos disponíveis, inclusive os de prateleira
- Funcionalidade e qualidade dos produtos disponíveis
- Conhecimento e capacitação dos fornecedores potenciais
- Impacto nas competências principais
- Licenças, garantias, responsabilidades e limitações associadas com os produtos que estão sendo adquiridos
- Disponibilidade do produto
- Questões de propriedade
- Redução de riscos

Muitos destes fatores são tratados pelo projeto. [PA160.IG102.SP106.N105]

A decisão de fabricar ou comprar pode ser conduzida utilizando uma abordagem de avaliação formal. [PA160.IG102.SP106.N106]

Veja a área de processo de Análises de Decisões e Resoluções para obter maiores informações sobre como definir critérios e alternativas e como executar avaliações formais. *[PA160.IG102.SP106.N106.R101]*

Da mesma forma que a tecnologia evolui, as justificativas para a escolha entre desenvolver ou comprar um componente de produto também evoluem. Embora esforços complexos de desenvolvimento possam favorecer a compra de um componente de produto de prateleira, avanços na produtividade e nas ferramentas podem oferecer uma justificativa oposta. Produtos de prateleira podem ter documentações incompletas ou imprecisas e podem não ter suporte no futuro. [PA160.IG102.SP106.N101]

Uma vez que tenha sido tomada a decisão de se adquirir um componente de produto de prateleira, os requisitos são utilizados para se estabelecer um acordo com o fornecedor. Existem momentos em que “de prateleira” se refere a um item já existente que pode não estar prontamente disponível no mercado. Por exemplo, alguns itens de aviação e motores não são realmente “de prateleira”, mas podem ser prontamente adquiridos. Em alguns casos, o uso de tais itens não desenvolvidos se faz em situações onde as especificidades de desempenho e outras características esperadas do produto precisam estar dentro de limites definidos. Nestes casos, os requisitos e critérios de aceitação podem precisar ser incluídos no acordo com o fornecedor e ser gerenciados. Em outros casos, o produto de prateleira é literalmente de prateleira (software de processamento de texto, por exemplo) e não há um acordo com o fornecedor que precise ser gerenciado. [PA160.IG102.SP106.N102]

Veja a área de processo de Gerenciamento de Acordos com Fornecedores para obter maiores informações sobre como tratar a aquisição de componentes de produtos que serão comprados. *[PA160.IG102.SP106.N102.R101]*

Produtos de Trabalho Típicos

1. Critérios para design e reuso de componentes de produtos [PA160.IG102.SP106.W101]
2. Análises de fabricar ou comprar (Make-or-buy analyses) [PA160.IG102.SP106.W102]
3. Instruções para a escolha de componentes de produtos de prateleira [PA160.IG102.SP106.W103]
4. Conjunto de produtos de prateleira padronizados de acordo com os critérios definidos para design e reuso de componentes de produtos.
5. Estratégia de não desenvolvimento de itens.

   A avaliação de itens a serem reutilizados inclui a definição de uma estratégia para gerência de registros relacionados ao uso de produtos e itens não desenvolvidos e elementos de serviço.
6. Plano para evolução das versões dos produtos de prateleira e potenciais impactos no produto ou serviço em desenvolvimento.

Sub-práticas

1. Desenvolver critérios para o reuso de designs de componentes de produtos. [PA160.IG102.SP106.SubP102]
2. Analisar os designs para determinar se os componentes do produto deverão ser desenvolvidos, reutilizados ou comprados. [PA160.IG102.SP106.SubP103]
3. Quando itens comprados ou não desenvolvidos (de prateleira, de prateleira governamentais e reuso) forem selecionados, planejar a sua manutenção. [PA160.IG102.SP106.SubP101]

   Para Engenharia de Software

   Considerar como será tratada a compatibilidade de futuros releases de um sistema operacional ou de um gerenciador de banco de dados. *[PA160.IG102.SP106.SubP101.AMP101]*

SG 3 Desenvolver Especificações da Interface

Desenvolver especificação de interface para escolher para selecionar o produto ou elementos de serviço.

Especificações contendo uma boa definição do conjunto de requisitos são estabelecidas através das interfaces entre produtos e elementos de serviço. Detalhes de interfaces externas não especificados em alto nível do produto e requisitos de serviço são completados. Especificações do formato e conteúdo da interface devera ser baseado no padrão industrial ou no estabelecimento interno e padrão de manutenção.

Uma especificação de uma interface é um documento ou conjunto de informações que descrevem completamente uma interface em termos de todas as restrições e requisitos de projeto localizados na interface.

Interfaces devem ser bem definidas adequadamente para estabelecer padrões do conteúdo da informação. Por exemplo, uma boa definição de uma interface dos elementos de produto pode incluir a origem da mensagem, destino, estímulos de eventos de comunicação, características dos elementos de dados (intervalo, tipo de dados, precisão, certeza, etc) e propriedades elétricas e mecânicas. Interfaces externas são usualmente estabelecidas antes de desenvolver o projeto do produto ou do serviço, como parte dos requisitos.

Isto ocorre também por chamadas de restrições de interface de um sistema existente ou através de uma aplicação de alto-nível da área de processo de projeto para o produto ou serviço num diferente escopo ou contexto.

SP 3.1 Criar Design de Interfaces Utilizando Critérios

Criar design de interfaces abrangentes de componentes de produtos, em termos dos critérios estabelecidos e mantidos. *[PA160.IG102.SP105]*

Os designs de interface incluem: [PA160.IG102.SP105.N101]

- Origem
- Destino
- Estímulos e características de dados para software
- Características elétricas, mecânicas e funcionais para hardware

Os critérios para as interfaces, muitas vezes, refletem uma lista abrangente de parâmetros críticos que devem ser definidos ou, pelo menos, investigados, para assegurar sua aplicabilidade. Estes parâmetros são, muitas vezes, específicos de um dado tipo de produto (por exemplo, software, mecânico, elétrico) e são freqüentemente associados com características de segurança, proteção, durabilidade e missão crítica. [PA160.IG102.SP105.N102]

Produtos de Trabalho Típicos

1. Especificações de design de interfaces [PA160.IG102.SP105.W101]
2. Documentos de controle de interfaces [PA160.IG102.SP105.W102]
3. Critérios de especificação de interfaces [PA160.IG102.SP105.W103]
4. Justificativa para o design de interface selecionado [PA160.IG102.SP105.W104]

Sub-práticas

1. Definir critérios de interfaces. [PA160.IG102.SP105.SubP101]

   Estes critérios podem ser uma parte dos ativos de processos da organização. [PA160.IG102.SP105.SubP101.N101]

Veja a área de processo de Definição do Processo Organizacional para obter maiores informações sobre o estabelecimento e manutenção dos ativos de processos organizacionais. *[PA160.IG102.SP105.SubP101.N101.R101]*

2. Aplicar os critérios às alternativas de design das interfaces. [PA160.IG102.SP105.SubP102]

   Veja a área de processo de Análises de Decisões e Resoluções para obter maiores informações sobre identificação de critérios e seleção de alternativas com base naqueles critérios. *[PA160.IG102.SP105.SubP102.R101]*

3. Documentar os designs de interfaces selecionados e a justificativa para a seleção. [PA160.IG102.SP105.SubP103]

SG 4 Desenvolver Especificações dos Componentes

Estabelecer especificação do projeto para cada elemento do produto ou serviço.

Os designs de produtos e componentes de produtos devem fornecer o conteúdo apropriado não somente para a implementação, mas também para outras fases do ciclo de vida do produto, como modificações, recompra, manutenção, sustentação e instalação. A documentação do design fornece uma referência para suportar o entendimento mútuo do design pelos stakeholders relevantes e apoiar futuras mudanças no design, tanto durante o desenvolvimento como nas fases subseqüentes do ciclo de vida do produto. Uma descrição completa do design é documentada em um pacote de dados técnicos, que inclui uma gama completa de recursos e parâmetros incluindo formato, encaixe, funções, interfaces, características do processo de construção e outros parâmetros. Padrões organizacionais estabelecidos ou de design de projetos (por exemplo, checklists, templates, frameworks de objetos) formam a base para atingir um alto nível de definição e completitude na documentação do design. [PA160.IG102.N101]

O estágio final da especificação é quando a especificação detalhada das operações e restrições é realizada. Dada uma interface esta define os principais estados potenciais dos objetos de componentes no modelo de informação de interface, e então especificando pré e pós-condições e capturando regras de negócio como restrições. As pré e pós-condições e outras restrições são referencia para os tipos no modelo de informação de interface e os tipos dos parâmetros. Em adição a estes detalhes de especificação de interface, este estágio também prova que as especificações das restrições são especificas para uma particular especificação de um componente e independem de cada interface. Esta especificação das restrições dos componentes determina como os tipos de definições em uma interface individual irão corresponder um ao outro no contexto do componente.

A arquitetura não poderá alterar este estágio. Esta tarefa de especificação detalhada deverá ser garantida, uma vez que a arquitetura for estabelecida e todas as operações de interface tenham sido identificadas. A ação de escrever regras precisas para cada operação pode ajudar a descobrir parâmetros ausentes, ou informações ausentes, mas a ênfase está no ponto de detalhe para um modelo estável.

SP 4.1 Estabelecer um Pacote de Dados Técnicos

Estabelecer e manter um pacote de dados técnicos. *[PA160.IG102.SP103]*

Um pacote de dados técnicos oferece ao desenvolvedor uma descrição abrangente do produto ou do componente de produto, conforme ele é desenvolvido. Tal pacote também fornece flexibilidade na aquisição de suprimentos em várias ocasiões, como em contratação baseada em desempenho ou de construir para receber. [PA160.IG102.SP103.N102]

O design é registrado em um pacote de dados técnicos que é criado durante o design preliminar para documentar a definição da arquitetura. Este pacote de dados técnicos é mantido durante toda a vida do produto para registrar os detalhes essenciais do design do produto. O pacote de dados técnicos fornece a descrição de um produto ou componente de produto (incluindo processos relacionados ao ciclo de vida do produto, se não forem tratados como componentes de produtos separados) que suporta uma estratégia de aquisição ou as fases de implementação, produção, engenharia e suporte logístico do ciclo de vida do produto. A descrição inclui a definição da configuração de design necessária e dos procedimentos para assegurar a adequação do desempenho do produto ou componente de produto. Isto inclui todos os dados técnicos disponíveis, como diagramas, listas associadas, especificações, descrições de design, bancos de dados de design, padrões, requisitos de desempenho, provisões de garantia da qualidade e detalhes de empacotamento. O pacote de dados técnicos inclui uma descrição da solução alternativa que foi escolhida para implementação. [PA160.IG102.SP103.N106]

Um pacote de dados técnicos deverá incluir o seguinte, se tais informações forem apropriadas para o tipo de produto ou componente de produto (por exemplo, requisitos de material e de manufatura podem não ser úteis para componentes de produtos associados com serviços ou processos de software): [PA160.IG102.SP103.N103]

- Descrição da arquitetura do produto
- Requisitos alocados
- Descrições de componentes de produtos

- Descrições de processos relacionados com o ciclo de vida do produto, se não estiverem descritos como componentes de produtos separados
- Características dos produtos chave
- Características físicas exigidas e restrições
- Requisitos de interface
- Requisitos de materiais (listas e características de materiais)
- Requisitos de fabricação e manufatura (para o equipamento original de fabricação e para o suporte de campo)
- Os critérios de verificação utilizados para assegurar que os requisitos foram atendidos
- Condições de uso (ambientes) e cenários operacionais/de uso, modos e estados de operações, suporte, treinamento, manufatura, descarte e verificações durante toda a vida do produto
- Justificativas para decisões e características (requisitos, alocações de requisitos e opções de design)

Uma vez que descrições de design podem envolver um volume muito grande de dados e ser cruciais para o desenvolvimento bem sucedido do componente de produto, é recomendável estabelecer critérios para organizar e selecionar o conteúdo dos dados. É particularmente útil utilizar a arquitetura do produto como um meio de organizar esses dados e abstrair visões que sejam claras e relevantes para uma questão ou característica de interesse. Estas visões incluem: [PA160.IG102.SP103.N104]

- Clientes
- Requisitos
- O ambiente
- Funcional
- Lógica
- Segurança
- Dados
- Estados/modos
- Construção
- Gerenciamento

Estas visões estão documentadas em um pacote de dados técnicos. [PA160.IG102.SP103.N105]

Produtos de Trabalho Típicos

1. Pacote de dados técnicos [PA160.IG102.SP103.W101]

Sub-práticas

1. Determinar a quantidade de níveis de design e o nível apropriado de documentação para cada nível de design. [PA160.IG102.SP103.SubP101]

Determinar o número de níveis de componentes de produtos (por exemplo, subsistema, item de configuração de hardware, placa de circuito, item de configuração de software do computador [CSCI], componente de produto de software do computador, unidade de software de computador) que exigem documentação e rastreabilidade de requisitos, é importante para gerenciar os custos de documentação e para suportar os planos de integração e verificação [PA160.IG102.SP103.SubP101.N101]

2. Basear as descrições detalhadas de design nos requisitos alocados de componentes de produto, arquitetura e níveis mais altos de design. [PA160.IG102.SP103.SubP102]
3. Documentar o design no pacote de dados técnicos. [PA160.IG102.SP103.SubP103]
4. Documentar a justificativa para as decisões chave (isto é, que têm um efeito significativo no custo, cronograma ou desempenho técnico) tomadas ou definidas. [PA160.IG102.SP103.SubP104]
5. Revisar o pacote de dados técnicos, conforme necessário. [PA160.IG102.SP103.SubP105]

SP 4.2 Criar o Design do Produto ou Componente do Produto

Desenvolver um design detalhado para o produto ou componente do produto. *[PA160.IG102.SP101]*

O design do produto consiste de duas grandes fases que podem se sobrepor na execução: design preliminar e detalhado. O design preliminar estabelece as capacidades do produto e a arquitetura do produto, incluindo partições de produto, identificações de componentes de produtos, estados e modos do sistema, principais interfaces entre os componentes e interfaces com produtos externos. O design detalhado define completamente a estrutura e capacidades dos componentes do produto. [PA160.IG102.SP101.N101]

Durante o design detalhado, os detalhes da arquitetura do produto são finalizados, os componentes do produto são completamente definidos e as interfaces são totalmente caracterizadas. Os designs dos componentes dos produtos podem ser otimizados para certas qualidades ou características de desempenho. Os designers podem avaliar o uso de produtos legados ou de prateleira para os componentes de produtos. Conforme o design amadurece, os requisitos atribuídos a componentes de produtos de nível mais baixo são rastreados para assegurar que estes requisitos estão sendo satisfeitos. [PA160.IG102.SP101.N105]

Veja a área de processo de Gerenciamento de Requisitos para obter maiores informações sobre como rastrear requisitos para componentes de produtos. *[PA160.IG102.SP101.N105.R101]*

Para Engenharia de Software

O design detalhado está concentrado no desenvolvimento do componente de produto de software. A estrutura interna dos componentes de produtos é definida, o modelo de dados é gerado, os algoritmos são desenvolvidos e heurísticas são estabelecidas para fornecer capacidades de componentes de produtos que satisfazem os requisitos alocados. *[PA160.IG102.SP101.N105.AMP101]*

Um completo conjunto de detalhes da especificação do projeto para cada elemento inclui todas as restrições do projeto, restrições de interface, e alocação dos requisitos que são necessários e suficientes a fim de produzir ou adquirir um elemento que é capaz de reunir condições de verificações e validações. Nos níveis intermediários do projeto, a especificação do projeto de um elemento é usada também como base pelo projeto de um elemento ou pela outsourcing daquele elemento. Na camada mais baixa de projeto, a especificação do projeto do elemento (componente) provê uma condição essencial para a construção ou codificação do elemento.

Produtos de Trabalho Típicos

1. Designs dos componentes de produtos [PA160.IG102.SP101.W102]
2. Documentação e especificação dos componentes.

   Uma completa especificação do componente descreve um componente e suas interfaces em termos de todas as restrições e requisitos de design (incluindo funcionais, alocações, performance e restrições) colocados sobre o componente ou interface e o método de qualificação de cada requisito.

Sub-práticas

1. Estabelecer e manter critérios contra os quais o design pode ser avaliado. [PA160.IG102.SP101.SubP101]

   Exemplos de atributos, além do desempenho esperado, para os quais podem ser estabelecidos critérios de design incluem: [PA160.IG102.SP101.SubP101.N101]

   - Modular
   - Claro
   - Simples
   - Possível de ser mantido
   - Verificável
   - Portável
   - Confiável
   - Preciso
   - Seguro
   - Possível de ser expandido
   - Usável

2. Identificar, desenvolver ou adquirir os métodos de design apropriados para o produto. [PA160.IG102.SP101.SubP102]

Métodos de design eficientes podem incorporar uma ampla gama de atividades, ferramentas e técnicas descritivas. Se um dado método é eficiente ou não depende da situação. Duas empresas podem ter métodos de design muito eficientes para os produtos nos quais elas são especializadas, mas estes métodos podem não ser eficientes em empreendimentos de cooperação. Métodos altamente sofisticados não são necessariamente eficientes nas mãos de designers que não tenham sido treinados no uso destes métodos. [PA160.IG102.SP101.SubP102.N101]

Se um método é eficiente ou não também depende da quantidade de auxílio que este fornece ao designer e da eficiência de custo deste auxílio. Por exemplo, um recurso de prototipação de multicamadas pode não ser apropriado para um simples componente de produto, mas poderia ser a coisa certa a ser feita para um desenvolvimento de um produto complexo, sem precedentes e caro. Técnicas rápidas de prototipação, entretanto, podem ser altamente eficientes para muitos componentes de produtos. Métodos que utilizam ferramentas para assegurar que um design tratará todos os atributos necessários para implementar o design do componente do produto podem ser muito eficientes. Por exemplo, uma ferramenta de design que "conhece" as capacidades dos processos de fabricação pode permitir que a variação no processo de construção seja considerada na tolerância do design. [PA160.IG102.SP101.SubP102.N102]

Exemplos de técnicas e métodos que facilitam o design eficiente incluem: [PA160.IG102.SP101.SubP102.N103]

- Protótipos
- Modelos estruturais
- Design orientado a objetos
- Análise essencial de sistemas
- Modelos de Entidades Relacionamentos
- Reuso do design
- Design patterns (padrões de design)

3. Assegurar que o design adere aos padrões e critérios de design aplicáveis. [PA160.IG102.SP101.SubP103]

Exemplos de padrões de design incluem (alguns ou todos estes padrões podem ser critérios de design, particularmente em circunstâncias onde os padrões não tenham sido estabelecidos): [PA160.IG102.SP101.SubP103.N101]

- Padrões de interface com o operador
- Padrões de segurança
- Restrições de produção
- Tolerâncias do design
- Padrões de partes (por exemplo, particionamento e desperdício da produção)

4. Assegurar que o design adere aos requisitos alocados. [PA160.IG102.SP101.SubP104]

Os componentes de produtos de prateleira identificados devem ser levados em conta. Por exemplo, colocar componentes de produtos existentes na arquitetura do produto poderia modificar os requisitos e a alocação dos requisitos. [PA160.IG102.SP101.SubP104.N101]

5. Documentar o design. [PA160.IG102.SP101.SubP105]
6. Adaptar as incompatibilidades.

   Adaptar as incompatibilidades entre os componentes: de parâmetro; de operação e operações incompletas.

7. Completar os serviços oferecidos parcialmente.

   Implementar alguns serviços que não são completamente oferecidos pelos componentes.

8. Identificar as restrições do projeto para cada nível do projeto.

   Identificar as restrições do projeto como requisitos para cada nível do projeto. As restrições do projeto podem incluir restrições relacionadas a estrutura, elementos e interfaces dos componentes.

9. Rever os requisitos dos componentes para assegurar que os componentes são necessários e suficientemente especificados com respeito à satisfação dos requisitos de alto nível.

   Avaliar os requisitos com o objetivo de verificar se os requisitos refletem as necessidades do domínio do projeto.

SP 4.3 Alocar Requisitos de Componentes de Produtos

Alocar os requisitos para cada componente de produto. *[PA157.IG103.SP102]*

Os requisitos para os componentes de produtos da solução definida incluem a alocação do desempenho do produto; restrições de design e encaixe, formato e funções para atender os requisitos e facilitar a produção. Em casos onde um nível mais elevado de requisitos especifica um desempenho que será de responsabilidade de dois ou mais componentes de produtos, o desempenho deve ser particionado para alocação única a cada componente de produto como um requisito derivado. [PA157.IG103.SP102.N101]

Produtos de Trabalho Típicos

1. Planilhas de alocação de requisitos [PA157.IG103.SP102.W101]
2. Alocações provisionais de requisitos [PA157.IG103.SP102.W102]
3. Restrições de design [PA157.IG103.SP102.W103]
4. Requisitos derivados [PA157.IG103.SP102.W104]
5. Relacionamentos entre os requisitos derivados [PA157.IG103.SP102.W105]

Sub-práticas

1. Alocar requisitos a funções. [PA157.IG103.SP102.SubP101]
2. Alocar requisitos a componentes de produtos. [PA157.IG103.SP102.SubP102]

3. Alocar restrições de design a componentes de produtos. [PA157.IG103.SP102.SubP103]

4. Documentar relacionamentos entre os requisitos alocados. [PA157.IG103.SP102.SubP104]

   Relacionamentos incluem as dependências nas quais uma mudança em um requisito pode afetar outros requisitos. [PA157.IG103.SP102.SubP104.N101]

SG 5 Implementar o Design do Produto

Componentes de produtos e a documentação de suporte associada são implementados a partir dos seus designs. *[PA160.IG103]*

Componentes de produtos são implementados a partir dos designs estabelecidos pelas práticas específicas da meta específica Desenvolver o Design. A implementação normalmente inclui testes de unidade dos componentes do produto, antes de enviá-los para a integração do produto e desenvolvimento da documentação do usuário final. [PA160.IG103.N101]

SP 5.1 Implementar o Design

Implementar os designs dos componentes de produtos. *[PA160.IG103.SP101]*

Para Engenharia de Software

Código de software é um típico componente de produto de software. *[PA160.IG103.SP101.AMP101]*

Uma vez que o design tenha sido completado, ele é implementado como um componente de produto. As características desta implementação dependem do tipo de componente de produto. [PA160.IG103.SP101.N101]

A implementação do design no nível mais alto da hierarquia do produto envolve a especificação de cada um dos componentes de produtos no próximo nível da hierarquia do produto. Esta atividade inclui a alocação, refinamento e verificação de cada componente de produto. Ela também envolve a coordenação entre os diversos esforços de desenvolvimento de componentes de produtos. [PA160.IG103.SP101.N103]

Veja a área de processo de Desenvolvimento de Requisitos para obter maiores informações sobre a alocação e refinamento de requisitos. *[PA160.IG103.SP101.N103.R101]*

Veja a área de processo de Integração de Produtos para obter maiores informações sobre o gerenciamento de interfaces e a integração de produtos e componentes de produtos. *[PA160.IG103.SP101.N103.R102]*

Exemplos de características desta implementação são: [PA160.IG103.SP101.N102]

- O software é codificado.
- Os dados são documentados.
- Os serviços são documentados.
- As partes elétricas e mecânicas são fabricadas.
- Processos de manufatura específicos do produto são colocados em operação.
- Os processos são documentados.
- Os recursos são construídos.
- Os materiais são produzidos (por exemplo, um material específico de um produto poderia ser um tipo de petróleo, óleo, lubrificante ou uma nova liga).

Produtos de Trabalho Típicos

1. Design implementado [PA160.IG103.SP101.W101]

Sub-práticas

1. Usar métodos eficientes para implementar os componentes dos produtos. [PA160.IG103.SP101.SubP101]

Para Engenharia de Software

Exemplos de métodos de codificação de software incluem: [PA160.IG103.SP101.SubP101.AMP101]

- *Programação estruturada*
- *Programação orientada a objetos*
- *Geração automática de códigos*
- *Reuso de código de software*
- *Uso de padrões de design (design patterns) aplicáveis*

Para Engenharia de Sistemas

Exemplos de métodos de fabricação apropriados incluem: [PA160.IG103.SP101.SubP101.AMP102]

- *Prensa*
- *Modelagem*
- *Formas*
- *Combinação*
- *Corte*
- *Ferramental*
- *Solda*
- *Extrusão*

2. Aderir aos padrões e critérios aplicáveis. [PA160.IG103.SP101.SubP102]

Para Engenharia de Software

Exemplos de padrões para codificação de software incluem: *[PA160.IG103.SP101.SubP102.AMP101]*

- *Padrões de linguagens*
- *Convenções de nomes de variáveis*
- *Estruturas de linguagem aceitáveis*
- *Estrutura e hierarquia de componentes de produtos de software*
- *Formato do código e comentários*

Para Engenharia de Software

Exemplos de critérios de codificação de software incluem: *[PA160.IG103.SP101.SubP102.AMP102]*

- *Modularização*
- *Clareza*
- *Simplicidade*
- *Estrutura (por exemplo, sem GOTOs, uma entrada, uma saída)*
- *Possibilidade de manutenção*

Para Engenharia de Sistemas

Exemplos de padrões incluem: *[PA160.IG103.SP101.SubP102.AMP103]*

- *Listas de Partes Padrão (Standard Parts Lists)*
- *Requisitos de diagramação padrão*
- *Padrões da International Organization for Standardization (ISO) T3303 para as partes manufaturadas*

Para Engenharia de Sistemas

Exemplos de critérios incluem: *[PA160.IG103.SP101.SubP102.AMP104]*

- *Possibilidade de manutenção*
- *Confiabilidade*
- *Segurança*

3. Conduzir revisões por pares dos componentes de produtos selecionados. [PA160.IG103.SP101.SubP103]

   Veja a área de processo de Verificação para obter maiores informações sobre como conduzir revisões por pares. *[PA160.IG103.SP101.SubP103.R101]*

4. Executar testes de unidade do componente de produto, conforme apropriado. [PA160.IG103.SP101.SubP104]

   Note que o teste de unidade não está limitado ao software. Testes de unidade envolvem o teste de unidades individuais de hardware ou software ou grupos de itens relacionados antes da sua integração. [PA160.IG103.SP101.SubP104.N101]

   Veja a área de processo de Verificação para obter maiores informações sobre métodos e procedimentos de verificação e sobre como verificar os produtos de trabalho contra seus requisitos especificados. *[PA160.IG103.SP101.SubP104.N101.R101]*

Para Engenharia de Software

Exemplos de métodos de testes de unidade incluem: *[PA160.IG103.SP101.SubP104.N101.AMP101]*

- *Teste de avaliação de cobertura*
- *Teste de cobertura de ramos*
- *Teste de cobertura de predicados*
- *Teste de cobertura de caminhos*
- *Teste de valores de fronteira*
- *Teste de valores especiais*

5. Revisar o componente de produto, conforme necessário. [PA160.IG103.SP101.SubP105]

Um exemplo de quando o componente do produto precisa ser revisado é quando aparecem problemas durante a implementação que não poderiam ser previstos durante o design. [PA160.IG103.SP101.SubP105.N101]

SP 5.2 Desenvolver Documentação de Suporte ao Produto

Desenvolver e manter a documentação do usuário final. *[PA160.IG103.SP102]*

Esta prática específica desenvolve e mantém a documentação que será utilizada para instalar, operar e manter o produto. [PA160.IG103.SP102.N101]

Produtos de Trabalho Típicos

1. Materiais de treinamento do usuário final [PA160.IG103.SP102.W101]
2. Manual do usuário [PA160.IG103.SP102.W102]
3. Manual de operação [PA160.IG103.SP102.W103]
4. Manual de manutenção [PA160.IG103.SP102.W104]
5. Help Online [PA160.IG103.SP102.W105]

Sub-práticas

1. Revisar os requisitos, design, produto e resultados de testes para assegurar que as questões que afetam a documentação de instalação, operação e manutenção foram identificadas e resolvidas. [PA160.IG103.SP102.SubP101]
2. Usar métodos eficientes para desenvolver a documentação de instalação, operação e manutenção. [PA160.IG103.SP102.SubP102]
3. Aderir aos padrões aplicáveis de documentação. [PA160.IG103.SP102.SubP103]

Exemplos de padrões de documentação incluem: [PA160.IG103.SP102.SubP103.N101]

- Compatibilidade com os processadores de texto definidos
- Fontes aceitáveis
- Numeração de páginas, seções e parágrafos
- Consistência com um estilo definido de manual
- Uso de abreviações
- Marcações de classificações de segurança
- Requisitos de internacionalização

4. Desenvolver versões preliminares da documentação de instalação, operação e manutenção nas fases iniciais do ciclo de vida do projeto, para revisão com os stakeholders relevantes. [PA160.IG103.SP102.SubP104]

5. Conduzir revisões por pares da documentação de instalação, operação e manutenção. [PA160.IG103.SP102.SubP105]

   Veja a área de processo de Verificação para obter maiores informações sobre a condução de revisões por pares. [PA160.IG103.SP102.SubP105.R101]

6. Revisar a documentação de instalação, operação e manutenção, conforme necessário. [PA160.IG103.SP102.SubP106]

Exemplos de quando a documentação pode precisar ser revisada incluem quando os seguintes eventos ocorrem: [PA160.IG103.SP102.SubP106.N101]

- Mudanças de requisitos
- São feitas mudanças no design
- São feitas mudanças no produto
- Erros são identificados na documentação
- Consertos provisórios são identificados

7. Gerar um repositório de dados do projeto, incluindo documentação dos componentes envolvidos e a forma de utilização dos mesmos, que provê eficiente recuperação e proteção.

   Veja a área de processo de Gerência de Configuração para obter maiores informações sobre como gerenciar dados do projeto.

Apêndice C – Integração de Produtos

INTEGRAÇÃO DE PRODUTOS

Nível de Maturidade 3

Objetivo

O objetivo da Integração de Produtos (Product Integration) é montar o produto a partir dos componentes de produtos, assegurar que o produto, uma vez integrado, funciona apropriadamente e entregar o produto. [PA147]

Notas Introdutórias

Esta área de processo trata da integração de componentes de produtos em componentes de produtos mais complexos ou em produtos completos. O termo "integração" é utilizado neste sentido em toda esta área de processo e não deve ser confundido com a integração de pessoas ou atividades que possa estar descrita em outra parte do modelo. [PA147.N101]

O escopo desta área de processo é alcançar a completa integração do produto, através da progressiva montagem de componentes de produtos, em uma única etapa ou em etapas incrementais, de acordo com uma seqüência e um procedimento de integração definidos. [PA147.N102]

Um aspecto crítico da integração do produto é o gerenciamento das interfaces internas e externas dos produtos e dos componentes de produtos para assegurar a compatibilidade entre as interfaces. Deverá ser dada atenção ao gerenciamento das interfaces em todo o projeto. [PA147.N103]

A integração de produtos é mais que apenas uma única montagem dos componentes do produto na conclusão do design e fabricação. A integração de produtos pode ser conduzida de forma incremental, utilizando um processo iterativo de montagem de componentes de produtos, sua avaliação e, em seguida, montagem de mais componentes de produtos. Este processo pode começar com análises e simulações (por exemplo, seqüências, protótipos rápidos, protótipo virtuais e protótipos físicos) e progredir de forma estável para uma funcionalidade incremental mais realista até que o produto final seja conseguido. Em cada construção (*build*) sucessiva, protótipos (virtuais, rápidos ou físicos) são construídos, avaliados, aprovados e reconstruídos com base no conhecimento adquirido no processo de avaliação. O grau de prototipagem virtual versus física exigido depende da funcionalidade das ferramentas de design, da complexidade do produto e dos seus riscos associados. Há uma alta probabilidade de que o produto, integrado desta maneira, passará pela verificação e validação do produto. Para alguns produtos, a última fase de integração ocorrerá quando o produto for implantado no seu ambiente operacional pretendido. [PA147.N104]

Áreas de Processos Relacionadas

Veja a área de processo de Desenvolvimento de Requisitos para obter maiores informações sobre a identificação de requisitos de interfaces. [PA147.R101]

Veja a área de processo de Soluções Técnicas para obter maiores informações sobre como definir as interfaces e o ambiente de integração (quando o ambiente de integração precisar ser desenvolvido). [PA147.R102]

Veja a área de processo de Verificação para obter maiores informações sobre como verificar as interfaces, o ambiente de integração e os componentes de produtos progressivamente montados. [PA147.R103]

Veja a área de processo de Validação para obter maiores informações sobre como executar a validação dos componentes de produtos e do produto integrado. [PA147.R104]

Veja a área de processo de Gerenciamento de Riscos para obter maiores informações sobre a identificação de riscos e o uso de protótipos na mitigação de riscos para a compatibilidade das interfaces e para a integração de componentes de produtos.. [PA147.R105]

Veja a área de processo de Análises de Decisões e Resoluções para obter maiores informações sobre como utilizar um processo formal de avaliação para selecionar a seqüência e procedimentos apropriados de integração e para decidir se o ambiente de integração deverá ser adquirido ou desenvolvido. [PA147.R106]

Veja a área de processo de Gerenciamento de Configurações para obter maiores informações sobre o gerenciamento de mudanças nas definições de interfaces e sobre a distribuição de informações. *[PA147.R107]*

Veja a área de processo de Gerenciamento de Acordos com Fornecedores para obter maiores informações sobre como adquirir componentes de produtos ou partes do ambiente de integração. *[PA147.R108]*

Metas Específicas e Genéricas

SG 1 **Preparar para a Integração do Produto** [PA147.IG101]

A preparação para a integração do produto é conduzida.

SG 2 **Assegurar a Compatibilidade das Interfaces** [PA147.IG102]

As interfaces, internas e externas, dos componentes de produtos estão compatíveis.

SG 3 **Montar os Componentes de Produtos e Entregar o Produto** [PA147.IG103]

Os componentes de produtos verificados são montados e o produto integrado, verificado e validado é entregue.

GG 3 **Institucionalizar um Processo Definido** [CL104.GL101]

O processo é institucionalizado como um processo definido.

Tabela de Relacionamentos Práticas-Metas

SG 1 Preparar para a Integração do Produto [PA147.IG101]
- SP 1.1 Determinar a Seqüência de Integração
- SP 1.2 Estabelecer o Ambiente de Integração do Produto
- SP 1.3 Estabelecer os Procedimentos e Critérios de Integração do Produto

SG 2 Assegurar a Compatibilidade das Interfaces [PA147.IG102]
- SP 2.1 Revisar as Descrições das Interfaces quanto a sua Completitude
- SP 2.2 Gerenciar as Interfaces

SG 3 Montar os Componentes do Produto e Entregar o Produto [PA147.IG103]
- SP 3.1 Confirmar que os Componentes do Produto estão Prontos para Integração
- SP 3.2 Montar os Componentes do Produto
- SP 3.3 Avaliar os Componentes do Produto Montados
- SP 3.4 Empacotar e Entregar o Produto ou Componente do Produto

GG 3 Institucionalizar um Processo Definido [CL104.GL101]
- GP 2.1 (CO 1) Estabelecer uma Política Organizacional
- GP 3.1 (AB 1) Estabelecer um Processo Definido
- GP 2.2 (AB 2) Planejar o Processo

GP 2.3	(AB 3)	Fornecer Recursos
GP 2.4	(AB 4)	Atribuir Responsabilidades
GP 2.5	(AB 5)	Treinar as Pessoas
GP 2.6	(DI 1)	Gerenciar Configurações
GP 2.7	(DI 2)	Identificar e Envolver os Stakeholders Relevantes
GP 2.8	(DI 3)	Monitorar e Controlar o Processo
GP 3.2	(DI 4)	Coletar Informações de Melhorias
GP 2.9	(VE 1)	Avaliar Objetivamente a Aderência
GP 2.10	(VE 2)	Revisar o Status com o Nível Mais Alto de Gerência

Práticas Específicas por Meta

SG 1 Preparar para a Integração do Produto

A preparação para a integração do produto é conduzida. *[PA147.IG101]*

Preparar para a integração dos componentes do produto envolve estabelecer e manter uma seqüência de integração, o ambiente para se efetuar a integração e os procedimentos de integração. As práticas específicas da meta específica Preparar para a Integração do Produto se baseiam uma na outra da seguinte forma. A primeira prática específica determina a seqüência para a integração do produto e dos componentes do produto. A segunda determina o ambiente que será utilizado para se executar a integração do produto e dos componentes do produto. A terceira desenvolve procedimentos e critérios para a integração do produto e dos componentes do produto. A preparação para a integração começa bem cedo no projeto e a seqüência de integração é desenvolvida paralelamente com as práticas da área de processo de Soluções Técnicas. [PA147.IG101.N101]

SP 1.1 Determinar a Seqüência de Integração

Determinar a seqüência de integração de componentes do produto. *[PA147.IG101.SP101]*

Os componentes do produto que são integrados podem incluir aqueles que são parte do produto a ser entregue, juntamente com equipamento de teste, software de teste e outros itens de integração como circuitos elétricos. Uma vez que você tenha analisado as alternativas de seqüências de integração da montagem e do teste, selecione a melhor seqüência de integração. [PA147.IG101.SP101.N101]

A seqüência de integração do produto pode servir de base para uma montagem incremental e para a avaliação dos componentes do produto que oferecem uma base livre de problemas para a incorporação de outros componentes de produtos, conforme eles se tornem disponíveis, ou para a prototipação de componentes de produtos de alto risco. [PA147.IG101.SP101.N103]

A seqüência de integração deverá estar em harmonia com a seleção de soluções e o design do produto e dos componentes do produto na área de processo de Soluções Técnicas. [PA147.IG101.SP101.N104]

Veja a área de processo de Análises de Decisões e Resoluções para obter maiores informações sobre o uso de um processo formal de avaliação para selecionar a seqüência apropriada de integração de produtos. *[PA147.IG101.SP101.N104.R101]*

Veja a área de processo de Gerenciamento de Riscos para obter maiores informações sobre como identificar e tratar riscos associados com a seqüência de integração. *[PA147.IG101.SP101.N104.R102]*

Veja a área de processo de Gerenciamento de Acordos com Fornecedores para obter maiores informações sobre como fazer a transição de componentes de produtos adquiridos e sobre a necessidade de tratar estes componentes de produtos na seqüência de integração de produtos. *[PA147.IG101.SP101.N104.R103]*

Produtos de Trabalho Típicos

1. Seqüência de integração de produtos [PA147.IG101.SP101.W101]
2. Justificativa para selecionar ou rejeitar seqüências de integração [PA147.IG101.SP101.W102]

Sub-práticas

1. Identificar os componentes de produtos a serem integrados. [PA147.IG101.SP101.SubP101]
2. Identificar as verificações da integração do produto a serem executadas, utilizando a definição das interfaces entre os componentes de produtos. [PA147.IG101.SP101.SubP102]
3. Identificar seqüências alternativas de integração de componentes de produtos. [PA147.IG101.SP101.SubP103]

   Isto pode incluir definir as ferramentas e equipamentos de testes específicos para suportar a integração do produto. [PA147.IG101.SP101.SubP103.N101]
4. Selecionar a melhor seqüência de integração. [PA147.IG101.SP101.SubP105]
5. Periodicamente rever a seqüência de integração do produto e revisá-la, conforme necessário. [PA147.IG101.SP101.SubP106]

   Avaliar a seqüência de integração do produto para assegurar que as variações nos cronogramas de produção e entrega não tiveram um impacto adverso na seqüência ou comprometeram os fatores sobre os quais as decisões anteriores foram tomadas. [PA147.IG101.SP101.SubP106.N101]
6. Registrar a justificativa para as decisões tomadas e postecipadas. [PA147.IG101.SP101.SubP107]

SP 1.2 Estabelecer o Ambiente para a Integração do Produto

Estabelecer e manter o ambiente necessário para suportar a integração dos componentes do produto. *[PA147.IG101.SP102]*

Veja a área de processo de Soluções Técnicas para obter maiores informações sobre decisões de comprar ou fabricar (make-or-buy decisions). *[PA147.IG101.SP102.R101]*

O ambiente para a integração do produto pode ser adquirido ou desenvolvido. Para estabelecer um ambiente, requisitos para a compra ou desenvolvimento de equipamentos, software ou outros recursos precisarão ser desenvolvidos. Estes requisitos são reunidos quando se implementa os processos associados com a área de processo de Desenvolvimento de Requisitos. O ambiente de integração de produtos pode incluir a reutilização de recursos organizacionais já existentes. A decisão de adquirir ou desenvolver o ambiente de integração do produto é tratada nos processos associados com a área de processo de Soluções Técnicas. [PA147.IG101.SP102.N101]

O ambiente necessário em cada etapa do processo de integração do produto pode incluir equipamento de testes, simuladores (tomando o lugar de componentes de produtos não disponíveis), peças de equipamento real e dispositivos de gravação. [PA147.IG101.SP102.N102]

Produtos de Trabalho Típicos

1. Ambiente verificado para integração de produtos [PA147.IG101.SP102.W101]
2. Documentação de suporte para o ambiente de integração do produto [PA147.IG101.SP102.W102]

Sub-práticas

1. Identificar os requisitos para o ambiente de integração do produto. [PA147.IG101.SP102.SubP101]
2. Identificar critérios e procedimentos de verificação para o ambiente de integração do produto. [PA147.IG101.SP102.SubP102]
3. Decidir entre construir ou comprar o ambiente necessário para a integração do produto. [PA147.IG101.SP102.SubP103]

   Veja a área de processo de Gerenciamento de Acordos com Fornecedores para obter maiores informações sobre como adquirir partes do ambiente de integração. [PA147.IG101.SP102.SubP103.R101]
4. Desenvolver um ambiente de integração, se um ambiente adequado não puder ser adquirido. [PA147.IG101.SP102.SubP104]

   Para projetos complexos e sem precedentes, o ambiente de integração do produto pode ser um grande desenvolvimento. Em tal caso, ele envolveria um planejamento do projeto, desenvolvimento de requisitos, soluções técnicas, verificação, validação e gerenciamento de riscos. [PA147.IG101.SP102.SubP104.N101]
5. Manter o ambiente de integração do produto durante todo o projeto. [PA147.IG101.SP102.SubP105]
6. Descartar as partes do ambiente que não são mais úteis. [PA147.IG101.SP102.SubP106]

SP 1.3 Estabelecer Procedimentos e Critérios de Integração do Produto

Estabelecer e manter procedimentos e critérios para integração dos componentes do produto. *[PA147.IG101.SP103]*

Os procedimentos para a integração dos componentes do produto podem incluir coisas como o número de iterações incrementais a serem executadas e detalhes dos testes esperados e outras avaliações a serem efetuadas em cada etapa. [PA147.IG101.SP103.N102]

Os critérios podem indicar se um componente do produto está pronto para a integração ou sua aceitabilidade. [PA147.IG101.SP103.N103]

Procedimentos e critérios para integração de produtos tratam do seguinte: [PA147.IG101.SP103.N105]

- Nível de testes dos componentes da versão (*build*)
- Verificação de interfaces
- Limites de desvio de desempenho
- Requisitos derivados para a montagem e suas interfaces externas
- Substituições permitidas de componentes
- Parâmetros do ambiente de testes
- Limites do custo do teste
- Equilíbrio de qualidade/custo para operações de integração
- Probabilidade de funcionamento correto
- Taxa de entrega e suas variações
- Intervalo de tempo do pedido até a entrega
- Disponibilidade de pessoal
- Disponibilidade dos recursos/linhas/ambientes de integração

Critérios podem ser definidos sobre como os componentes do produto serão verificados e as funções que se espera que eles tenham. Critérios podem ser definidos para como os componentes montados do produto e o produto integrado final será validado e entregue. [PA147.IG101.SP103.N106]

Critérios também podem restringir o grau de simulação permitida para que um componente de produto passe em um teste ou podem restringir o ambiente a ser utilizado para o teste de integração. [PA147.IG101.SP103.N104]

Produtos de Trabalho Típicos

1. Procedimentos de integração de produtos [PA147.IG101.SP103.W101]
2. Critérios de integração de produtos [PA147.IG101.SP103.W102]

Sub-práticas

1. Estabelecer e manter os procedimentos de integração do produto para os componentes do produto. [PA147.IG101.SP103.SubP101]
2. Estabelecer e manter critérios para a integração e avaliação dos componentes do produto. [PA147.IG101.SP103.SubP102]
3. Estabelecer e manter critérios para a validação e entrega do produto integrado. [PA147.IG101.SP103.SubP103]

SG 2 Assegurar Compatibilidade das Interfaces

As interfaces, internas e externas, dos componentes do produto são compatíveis. *[PA147.IG102]*

Muitos problemas de integração de produtos surgem de aspectos desconhecidos ou fora de controle de interfaces internas e externas. Um gerenciamento eficiente dos requisitos de interfaces de componentes de produtos, especificações e designs ajuda a assegurar que as interfaces implementadas estarão completas e compatíveis. [PA147.IG102.N101]

SP 2.1 Revisar as Descrições de Interfaces quanto a sua Completitude

Revisar as descrições de interfaces quanto a sua cobertura e completitude. *[PA147.IG102.SP101]*

As interfaces deverão incluir, além das interfaces dos componentes do produto, todas as interfaces com o ambiente de integração do produto. [PA147.IG102.SP101.N101]

Produtos de Trabalho Típicos

1. Categorias de interfaces [PA147.IG102.SP101.W101]
2. Lista de interfaces por categoria [PA147.IG102.SP101.W102]
3. Mapeamento das interfaces com os componentes do produto e com o ambiente de integração do produto [PA147.IG102.SP101.W103]
4. Análise do reuso de interfaces de componentes

   Documento que especifica uma análise com relação à interface dos componentes, na finalidade de verificar a reusabilidade das mesmas.

Sub-práticas

1. Revisar os dados de interfaces quanto à completitude e assegurar completa cobertura de todas as interfaces. [PA147.IG102.SP101.SubP101]

   Considerar todos os componentes do produto e preparar uma tabela de relacionamentos. As interfaces são normalmente classificadas em três classes principais: ambientais, físicas e funcionais. As categorias comuns para estas classes incluem as seguintes: mecânicas, fluídas, sonoras, elétricas, climáticas, eletromagnéticas, térmicas, mensagens e homem-máquina ou interface humana. [PA147.IG102.SP101.SubP101.N101]

   Para Engenharia de Software

   Na categoria mensagem para software, as interfaces incluem: [PA147.IG102.SP101.SubP101.N101.AMP101]

   - *Origem*
   - *Destino*
   - *Estímulo*
   - *Protocolos e características de dados*

Para Engenharia de Sistemas

Para componentes mecânicos ou eletrônicos, os dados de interfaces deverão incluir: *[PA147.IG102.SP101.SubP101.N101.AMP102]*

- *Interfaces mecânicas (por exemplo, peso e tamanho, centro de gravidade, distanciamento das partes na operação, espaço exigido para manutenção, ligações fixas, ligações móveis, choques e vibrações recebidas da estrutura de suporte)*
- *Interfaces de ruído (por exemplo, ruído transmitido pela estrutura, ruído transmitido através do ar, acústica)*
- *Interfaces climáticas (por exemplo, temperatura, umidade, pressão, salinidade)*
- *Interfaces térmicas (por exemplo, dissipação de calor, transmissão de calor para a estrutura de suporte, características do ar condicionado)*
- *Interfaces fluidas (por exemplo, água fresca entrando ou saindo, água do mar entrando ou saindo para um produto naval / costeiro, ar condicionado, ar comprimido, nitrogênio, combustível, óleo lubrificante, saída de gás)*
- *Interfaces elétricas (por exemplo, consumo de suprimento de energia pela rede com valores transientes e de pico; controle não sensitivo de sinais para suprimento de energia, comunicações, etc; sinais sensitivos [links analógicos]; sinal de perturbação [microondas, etc]; sinal do solo em conformidade com o padrão TEMPEST)*
- *Interfaces eletromagnéticas (por exemplo, campo magnético, links de rádio e radar, guias de ondas de links de banda ótica, fibras óticas e coaxiais)*
- *Interfaces homem-máquina (por exemplo, síntese de áudio ou voz, reconhecimento de áudio ou voz, display [dial analógico, tela de TV ou display de cristal líquido, luzes indicativas emitindo diodos], controles manuais [pedais, joystick, bola, chaves, botões, touch screen])*

2. Assegurar que os componentes do produto e interfaces estão marcados para assegurar uma fácil e correta conexão com o componente de produto. [PA147.IG102.SP101.SubP102]

3. Periodicamente revisar a adequação das descrições de interfaces. [PA147.IG102.SP101.SubP103]

   Uma vez estabelecidas, as descrições de interfaces devem ser periodicamente revisadas para assegurar que não há desvio entre as descrições existentes e os produtos que estão sendo desenvolvidos, processados, produzidos ou comprados. [PA147.IG102.SP101.SubP103.N101]

SP 2.2 Gerenciar Interfaces

Gerenciar as definições de interfaces internas e externas, os designs e as mudanças em produtos e componentes de produtos. *[PA147.IG102.SP102]*

Os requisitos de interface dirigem o desenvolvimento das interfaces necessárias para integrar componentes de produtos. O gerenciamento das interfaces do produto e dos componentes do produto começa bem cedo no desenvolvimento do produto. As definições e designs para as interfaces afetam não somente os componentes do produto e sistemas externos, mas também afetam os ambientes de verificação e validação. [PA147.IG102.SP102.N104]

Veja a área de processo de Desenvolvimento de Requisitos para obter maiores informações sobre requisitos de interfaces. *[PA147.IG102.SP102.N104.R101]*

Veja a área de processo de Soluções Técnicas para obter maiores informações sobre o design de interfaces entre componentes de produtos. *[PA147.IG102.SP102.N104.R102]*

Veja a área de processo de Gerenciamento de Requisitos para obter maiores informações sobre como gerenciar as mudanças nos requisitos de interfaces. *[PA147.IG102.SP102.N104.R103]*

Veja a área de processo de Gerenciamento de Configurações para obter maiores informações sobre como distribuir mudanças nas descrições (especificações) de interfaces, de maneira que todos possam saber o atual estado das interfaces. *[PA147.IG102.SP102.N104.R104]*

O gerenciamento das interfaces inclui a manutenção da consistência das interfaces durante toda a vida do produto e a resolução de conflitos, não conformidades e questões relativas a mudanças. [PA147.IG102.SP102.N101]

As interfaces deverão incluir, além das interfaces de componentes do produto, todas as interfaces com o ambiente, bem como outros ambientes de verificação, validação, operação e suporte. [PA147.IG102.SP102.N102]

As mudanças na interface são documentadas, mantidas e estão prontamente acessíveis. [PA147.IG102.SP102.N103]

Produtos de Trabalho Típicos

1. Tabela de relacionamentos entre os componentes do produto e o ambiente externo (por exemplo, suprimento mestre de energia, produto de conexão, sistema de bus do computador) [PA147.IG102.SP102.W101]
2. Tabela de relacionamentos entre os diferentes componentes do produto [PA147.IG102.SP102.W102]
3. Lista de interfaces aprovadas definidas para cada par de componentes do produto, quando aplicável [PA147.IG102.SP102.W103]
4. Relatórios das reuniões do grupo de trabalho de controle de interfaces [PA147.IG102.SP102.W104]
5. Itens de ação para atualizar interfaces [PA147.IG102.SP102.W105]
6. Interface de programas de aplicação (Application program interface - API) [PA147.IG102.SP102.W106]
7. Descrição ou acordo de interfaces atualizado [PA147.IG102.SP102.W107]

Sub-práticas

1. Assegurar a compatibilidade das interfaces durante toda a vida do produto. [PA147.IG102.SP102.SubP101]
2. Resolver conflitos, não conformidades e questões de mudanças. [PA147.IG102.SP102.SubP102]
3. Manter um repositório de dados de interfaces acessível a todos os participantes do projeto. [PA147.IG102.SP102.SubP103]

   Um repositório único para os dados de interface oferece um mecanismo para assegurar que todos sabem onde estão os dados atuais de interfaces e podem acessá-los para uso. [PA147.IG102.SP102.SubP103.N101]

SG 3 Montar os Componentes do Produto e Entregar o Produto

Os componentes do produto verificados são montados e o produto integrado, verificado e validado é entregue. *[PA147.IG103]*

A integração dos componentes do produto se dá de acordo com a seqüência de integração do produto e os procedimentos disponíveis. Antes da integração, cada componente do produto deverá ser confirmado como estando em conformidade com seus requisitos de interface. Os componentes do produto são montados em componentes do produto maiores e mais complexos. Estes componentes do produto montados são verificados quanto a sua correta interoperação. Este processo continua até que a integração do produto esteja completa. Se, durante este processo, forem identificados problemas, o problema deverá ser documentado e um processo de ação corretiva iniciado. [PA147.IG103.N101]

Assegurar que a montagem dos componentes do produto em componentes do produto maiores e mais complexos é conduzida de acordo com a seqüência de integração do produto e os procedimentos disponíveis. A imediata recepção dos componentes do produto necessários e o envolvimento das pessoas certas contribui para a integração bem sucedida dos componentes do produto que compõem o produto. [PA147.IG103.N102]

SP 3.1 Confirmar que os Componentes do Produto estão Prontos para a Integração

Confirmar, antes da montagem, que cada componente do produto exigido para montar o produto foi apropriadamente identificado, funciona de acordo com sua descrição e que as interfaces do componente do produto estão em conformidade com as descrições de interfaces. *[PA147.IG103.SP101]*

Veja a área de processo de Verificação para obter maiores informações sobre como verificar os componentes de produtos. *[PA147.IG103.SP101.R101]*

Veja a área de processo de Soluções Técnicas para obter maiores informações sobre testes de unidade de componentes de produtos. *[PA147.IG103.SP101.R102]*

O objetivo desta prática específica é assegurar que o componente do produto apropriadamente identificado, que atende a sua descrição, pode realmente ser montado de acordo com a seqüência de integração do produto e os procedimentos disponíveis. Os componentes do produto são verificados quanto à quantidade, danos óbvios e consistência entre o componente do produto e as descrições de interfaces. [PA147.IG103.SP101.N101]

Os que conduzem a integração do produto são, em última análise, responsáveis por verificar se tudo está certo com os componentes do produto antes da montagem. [PA147.IG103.SP101.N102]

Produtos de Trabalho Típicos

1. Documentos de aceitação para os componentes do produto recebidos [PA147.IG103.SP101.W101]
2. Recibos de entrega [PA147.IG103.SP101.W102]
3. Listas de pacotes verificados [PA147.IG103.SP101.W103]
4. Relatórios de exceção [PA147.IG103.SP101.W104]
5. Dispensas [PA147.IG103.SP101.W105]

Sub-práticas

1. Rastrear o status de todos os componentes do produto, a partir do momento em que ficam disponíveis para integração. [PA147.IG103.SP101.SubP101]
2. Assegurar que os componentes do produto são entregues no ambiente de integração do produto em concordância com a seqüência de integração do produto e procedimentos disponíveis. [PA147.IG103.SP101.SubP102]
3. Confirmar o recebimento de cada componente do produto apropriadamente identificado. [PA147.IG103.SP101.SubP103]
4. Assegurar que cada componente do produto recebido atende sua descrição. [PA147.IG103.SP101.SubP104]
5. Verificar o status da configuração contra a configuração esperada. [PA147.IG103.SP101.SubP105]
6. Executar uma pré-verificação (por exemplo, por inspeção visual e utilizando medidas básicas) de todas as interfaces físicas, antes de conectar os componentes do produto. [PA147.IG103.SP101.SubP106]

SP 3.2 Montar os Componentes do Produto

Montar os componentes do produto de acordo com a seqüência de integração e procedimentos disponíveis. *[PA147.IG103.SP102]*

Veja a área de processo de Verificação para obter maiores informações sobre como verificar os componentes de produtos montados. *[PA147.IG103.SP102.R101]*

Veja a área de processo de Validação para obter maiores informações sobre como validar os componentes de produtos montados. *[PA147.IG103.SP102.R102]*

(Para os usuários da representação contínua, esta é uma prática específica da capacitação de nível 1. Os processos de integração de produtos nos níveis de capacitação 1 e 2 podem não incluir procedimentos e critérios, que são criados na prática específica Estabelecer Procedimentos e Critérios para Integração de Produtos na capacitação nível 3. Quando não há procedimentos ou critérios estabelecidos, utilize a seqüência estabelecida na prática específica Determinar a Seqüência de Integração para cumprir o desempenho da capacitação de nível 1). [PA147.IG103.SP102.N102]

As atividades de montagem desta prática específica e as atividades de avaliação da próxima prática específica são conduzidas de forma iterativa, desde os componentes iniciais do produto, passando por uma série de montagens de componentes do produto intermediários até o produto como um todo. [PA147.IG103.SP102.N101]

Produtos de Trabalho Típicos

1. Produto ou componentes do produto montados [PA147.IG103.SP102.W101]

Sub-práticas

1. Assegurar que o ambiente de integração do produto está pronto. [PA147.IG103.SP102.SubP101]
2. Assegurar que a seqüência de montagem está sendo executada de forma apropriada. [PA147.IG103.SP102.SubP102]

   Registrar todas as informações apropriadas (por exemplo, status da configuração, números de série dos componentes do produto, tipos e datas de calibração dos instrumentos de medição). [PA147.IG103.SP102.SubP102.N101]
3. Revisar a seqüência de integração do produto e procedimentos disponíveis, conforme apropriado. [PA147.IG103.SP102.SubP104]
4. Assegurar confiabilidade dos conectores.

   Os conectores são os únicos componentes do sistema que possuem o conhecimento de seus fluxos interativos. Dessa forma, eles são considerados os locais ideais para a implementação dos requisitos de qualidade do sistema, tais como a tolerância a falhas, distribuição e disponibilidade.

SP 3.3 Avaliar os Componentes do Produto Montados

Avaliar os componentes do produto montados com relação à compatibilidade da interface. *[PA147.IG103.SP103]*

Esta avaliação envolve o exame e o teste dos componentes do produto montados com relação ao desempenho, adequação e prontidão utilizando os procedimentos e ambiente disponíveis. Isto é feito, conforme apropriado, em diferentes etapas da montagem de componentes do produto, conforme identificado na seqüência de integração do produto e procedimentos disponíveis. A seqüência de integração do produto e procedimentos disponíveis podem definir uma seqüência de integração e avaliação mais refinada do que a que pode ser prevista apenas examinando a arquitetura do produto. Por exemplo, se uma montagem de componentes do produto é composta de quatro componentes do produto menos complexos, a seqüência de integração não necessariamente fará uma integração e avaliação simultânea das quatro unidades como se fossem uma. Em vez disso, as quatro unidades menos complexas podem ser integradas progressivamente, uma de cada vez, com uma avaliação após cada operação de montagem antes de chegar ao componente do produto mais complexo que atende a especificação da arquitetura do produto. Alternativamente, a seqüência de integração e procedimentos disponíveis poderiam determinar que somente uma avaliação final seria a melhor forma de fazer o serviço. [PA147.IG103.SP103.N101]

Produtos de Trabalho Típicos

1. Relatórios de exceção [PA147.IG103.SP103.W102]
2. Relatórios de avaliação de interfaces [PA147.IG103.SP103.W103]
3. Relatórios de resumo de integração do produto [PA147.IG103.SP103.W104]

Sub-práticas

1. Conduzir a avaliação dos componentes do produto montados, seguindo a seqüência de integração do produto e procedimentos disponíveis. [PA147.IG103.SP103.SubP101]
2. Registrar os resultados da avaliação. [PA147.IG103.SP103.SubP102]

   Exemplos de resultados incluem: [PA147.IG103.SP103.SubP102.N101]

   - Qualquer adaptação necessária nos procedimentos de integração
   - Qualquer mudança na configuração do produto (peças de reposição, novo release)
   - Avaliação de desvios de procedimentos
3. Medir os atributos de qualidade através dos conectores

   Por representar o elo de comunicação entre componentes arquiteturais, os conectores são os únicos componentes do sistema que possuem o conhecimento dos seus fluxos interativos.

SP 3.4 Empacotar e Entregar o Produto ou Componente do Produto

Empacotar o produto ou componente do produto montado e entregá-lo ao cliente apropriado. *[PA147.IG103.SP104]*

Veja a área de processo de Verificação para obter maiores informações sobre como verificar o produto ou uma montagem de componentes de produtos antes de empacotá-los. *[PA147.IG103.SP104.R101]*

Veja a área de processo de Validação para obter maiores informações sobre como validar o produto ou uma montagem de componentes de produtos antes de empacotá-los. *[PA147.IG103.SP104.R102]*

Os requisitos de empacotamento para alguns produtos podem ser tratados em suas especificações e critérios de verificação. Isto é especialmente importante quando os itens são armazenados e transportados pelo cliente. Em tais casos, pode haver uma série de condições ambientais e de stress definidas para o pacote. Em outras circunstâncias, fatores como os seguintes podem se tornar importantes: [PA147.IG103.SP104.N101]

- Economia e facilidade de transporte (por exemplo, uso de containers)
- Comunicação (por exemplo, utilizando filmes plásticos)
- Facilidade e segurança no desempacotamento (por exemplo, bordas afiadas, a força da cola utilizada, cuidados com crianças, o cuidado ambiental no material de empacotamento, peso)

O ajuste necessário para encaixar os componentes do produto na fábrica poderia ser diferente do necessário para encaixar os componentes do produto quando instalados no local operacional. Neste caso, o registro de log do produto para o cliente deverá ser utilizado para registrar tais parâmetros específicos. [PA147.IG103.SP104.N102]

Produtos de Trabalho Típicos

1. Produto ou componentes do produto empacotados [PA147.IG103.SP104.W101]
2. Documentação de entrega [PA147.IG103.SP104.W102]

Sub-práticas

1. Revisar os requisitos, design, produto, resultados de verificação e documentação para assegurar que as questões que afetam o empacotamento e a entrega do produto foram identificadas e resolvidas. [PA147.IG103.SP104.SubP101]
2. Usar métodos eficientes para empacotar e entregar o produto montado. [PA147.IG103.SP104.SubP102]

Para Engenharia de Software

Exemplos de métodos de empacotamento e entrega de software incluem: *[PA147.IG103.SP104.SubP102.AMP101]*

- *Fita magnética*
- *Disquetes*
- *Documentos impressos*
- *Compact disks (CDs)*
- *Outros tipos de distribuição eletrônica como a Internet*

3. Satisfazer os requisitos e padrões aplicáveis para o empacotamento e entrega do produto. [PA147.IG103.SP104.SubP103]

Para Engenharia de Software

Exemplos de requisitos e padrões para empacotamento e entrega do software incluem: *[PA147.IG103.SP104.SubP103.AMP101]*

- *Tipo de mídia de armazenamento e entrega*
- *Custódia das cópias mestre e backup do software*
- *Documentação exigida*
- *Copyrights*
- *Licenças provisionais*
- *Proteção do software*

Para Engenharia de Sistemas

Exemplos de requisitos e padrões incluem aqueles referentes a segurança, o ambiente, a proteção e a portabilidade. *[PA147.IG103.SP104.SubP103.AMP102]*

4. Preparar o local operacional para a instalação do produto. [PA147.IG103.SP104.SubP104]

   Preparar o local operacional pode ser responsabilidade do cliente ou usuários finais. [PA147.IG103.SP104.SubP104.N101]

5. Entregar o produto e documentação relacionada e confirmar o recebimento.

6. Instalar o produto no local operacional e confirmar a correta operação.

   Instalar o produto pode ser responsabilidade do cliente ou usuários finais. Em algumas circunstâncias, muito pouco pode precisar ser feito para confirmar a correta operação. Em outras circunstâncias, a verificação final do produto integrado ocorre no local operacional.