EDGAR LOPES BANHESSE

UM METAMODELO DE PERFIS DE CAPACIDADE DE PROCESSO

CAMPINAS
2013

UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA ELÉTRICA E DE COMPUTAÇÃO
DEPARTAMENTO DE ENGENHARIA DE COMPUTAÇÃO E AUTOMAÇÃO INDUSTRIAL

EDGAR LOPES BANHESSE

UM METAMODELO DE PERFIS DE CAPACIDADE DE PROCESSO

Orientador: Prof. Dr. Mario Jino
Coorientador: Dr. Clenio Figueiredo Salviano

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas para obtenção do título de Mestre em Engenharia Elétrica, na área de Engenharia de Computação.

ESTE EXEMPLAR CORRESPONDE À VERSÃO FINAL DA DISSERTAÇÃO
DEFENDIDA PELO ALUNO EDGAR LOPES BANHESSE
E ORIENTADO PELO PROF. DR. MARIO JINO

Assinatura do Orientador

_____

CAMPINAS
2013

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca da Área de Engenharia e Arquitetura
Rose Meire da Silva - CRB 8/5974

B225m

Banhesse, Edgar Lopes, 1979-
Um metamodelo de perfis de capacidade de processo / Edgar Lopes Banhesse. – Campinas, SP : [s.n.], 2013.

Orientador: Mario Jino.
Coorientador: Clenio Figueiredo Salviano.
Dissertação (mestrado) – Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Melhoria de processo - Projetos. 2. Engenharia de software. 3. Software - Desenvolvimento. I. Jino, Mario,1943-. II. Salviano, Clenio Figueiredo. III. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. IV. Título.

Informações para Biblioteca Digital

Título em outro idioma: A process capability profiles metamodel
Palavras-chave em inglês:
Process improvement - Projects
Software engineering
Software - Development
Área de concentração: Engenharia de Computação
Titulação: Mestre em Engenharia Elétrica
Banca examinadora:
Mario Jino [Orientador]
Marcello Thiry Comicholi da Costa
Mário Lucio Côrtes
Data de defesa: 31-07-2013
Programa de Pós-Graduação: Engenharia Elétrica

COMISSÃO JULGADORA - TESE DE MESTRADO

Candidato: Edgar Lopes Banhesse

Data da Defesa: 31 de julho de 2013

Título da Tese: "Um Metamodelo de Perfis de Capacidade de Processo"

Prof. Dr. Mario Jino (Presidente): _____

Prof. Dr. Marcello Thiry Comicholi da Costa: _____

Prof. Dr. Mário Lucio Côrtes: _____

A Deus, meus pais Elidio e Cícera e a minha esposa Vanessa.

Agradecimentos

Agradeço a Deus, por ter me dado o dom da vida e por ter planos tão maravilhosos para mim. "Porque Dele e por Ele, e para Ele, são todas as coisas; glória, pois, a Ele eternamente. Amém." (Romanos 11:36 – Bíblia).

Aos meus pais Elidio e Cícera, por terem me educado e me ensinado os verdadeiros valores da vida.

A minha amada esposa Vanessa, por ter paciência em escutar minhas explicações sobre metamodelo, por suas orações e pelas palavras de apoio e encorajamento.

A meus irmãos Thiago e Maria, pelo apoio e pelas orações.

Ao meu orientador Mario Jino e meu coorientador Clenio Salviano, pelo apoio, orientações e ensinamentos valiosos que possibilitaram realizar esse trabalho.

Aos professores Marcello Thiry e Mario Côrtes, pelas ideias e sugestões que ajudaram a melhorar o trabalho.

Ao meu grande amigo, Francisco Primo, pelos sábios conselhos e pelas longas e proveitosas conversas que tivemos.

Aos professores da UNICAMP e aos colaboradores do CTI (Em especial à DMPS), que tiveram paciência comigo e me ajudaram durante essa caminhada.

Aos especialistas em modelos, que dedicaram parte do seu precioso tempo para avaliar os resultados do PRO2PI-MMCA.

A comunidade e a ferramenta de desenvolvimento CakePHP, que possibilitaram a construção do software PRO2PI-MMCA como prova de conceito.

As minhas lindas meninas, Melissa e Larissa, que sempre me alegram e me surpreendem com suas brincadeiras, elas são as cachorrinhas mais adoráveis que alguém poderia ter.

"Frequentemente é necessário mais coragem para ousar fazer certo do que temer fazer errado."
(Abraham Lincoln)

"Deus não escolhe os capacitados, capacita os escolhidos.
Fazer ou não fazer algo só depende de nossa vontade e perseverança."
(Albert Einstein)

"Tenho constatado que toda perfeição tem limite;
mas não há limite para o teu mandamento."
(Salmos 119:96 – Bíblia – Nova Versão Internacional)

Resumo

A Melhoria de Processo de Software (MPS) em organizações intensivas em software está estabelecida na indústria como um meio bem sucedido para melhorar o desenvolvimento de software. Um ciclo de MPS tem sido orientado por um modelo de maturidade da capacidade, geralmente um modelo do CMMI ou um modelo baseado na Norma Internacional ISO/IEC 15504 (SPICE). Como consequência dessa melhora, existe um conjunto de fatores que levam a uma evolução da atual MPS para tratar novas oportunidades de melhoria. Entre esses fatores existem: os princípios fundamentais dos modelos e os múltiplos modelos.

O objetivo deste projeto de pesquisa é propor e mostrar que é viável utilizar um Metamodelo de Perfis de Capacidade de Processo para promover a integração de elementos de Múltiplos Modelos de Capacidade de Processo de Software. Este projeto faz parte de um esforço de pesquisa em direção a uma proposta de evolução da atual MPS. A especificação de um metamodelo foi revisada e foram especificadas as funcionalidades de uma ferramenta de software para representar e transformar arquiteturas de modelos de capacidade; esta ferramenta foi implementada como uma prova de conceito para validar o metamodelo.

A ferramenta foi utilizada para representar e transformar modelos de arquiteturas diferentes a fim de promover a integração de elementos de múltiplos modelos, seis Perfis de Capacidade de Processo foram gerados e avaliados por seis especialistas em modelos. A avaliação e análise dos perfis foram satisfatórias, pois mostrou que o metamodelo implementado foi capaz de representar e transformar elementos de Múltiplos Modelos de Capacidade de Processo.

Palavras-chave: Melhoria de Processo de Software. Engenharia Dirigida por Modelos. Metamodelo. Processo. Modelo de Capacidade de Processo.

Abstract

Software Process Improvement (SPI) in software intensive organizations has been established in industry as a successful way to improve software development. A SPI cycle has been driven by a capability maturity model, usually a CMMI model or a model based on the International Standard ISO/IEC 15504 (SPICE). As a result of this improvement, there is a group of factors that lead to an evolution in current SPI, addressing new opportunities of improvement. Among these factors are: fundamental principles of models and multiple models.

The objective of this research project is to propose and show that it is feasible to use a Process Capability Profiles Metamodel to promote integration of elements from Multiple Models of Software Process Capability. This project is part of research effort towards a proposed evolution of current SPI. A metamodel specification has been reviewed and the software tool features were specified to represent and transform capability models architectures; this tool has been implemented as a proof of concept to validate the metamodel.

The tool was used to represent and transform models of different architectures in order to promote the integration of elements from multiple models, six Process Capability Profiles have been generated and evaluated by six models experts. The profiles evaluation and analysis were satisfactory, whereas they have shown that the metamodel implemented was able to represent and transform elements from Multiple Models of Process Capability.

Keywords*: Software Process Improvement. Model Driven Engineering. Metamodel. Process. Process Capability Model.*

Lista de figuras

Figura 2.1 – Classificação dos artigos baseada na utilização do Metamodelo. 15
Figura 2.2 – Arquitetura CMMI. 21
Figura 2.3 – Perfis do CMMI-DEV. 22
Figura 2.4 – Comparação dos níveis do CMMI-DEV. 23
Figura 2.5 – Dimensões do modelo ASPICE. 25
Figura 2.6 – Arquitetura SPICE. 26
Figura 2.7 – Processos do grupo de processos ACQ do modelo ASPICE. 26
Figura 2.8 – Processos do grupo de processos SPL do modelo ASPICE. 27
Figura 2.9 – Processos do grupo de processos ENG do modelo ASPICE. 27
Figura 2.10 – Processos do grupo de processos SUP do modelo ASPICE. 27
Figura 2.11 – Processos do grupo de processos MAN do modelo ASPICE. 28
Figura 2.12 – Processo do grupo de processos PIM do modelo ASPICE. 28
Figura 2.13 – Processo do grupo de processos REU do modelo ASPICE. 28
Figura 2.14 – Arquitetura MPS.BR. 29
Figura 2.15 – Níveis de maturidade, processos e atributos de processos do MR-MPS-SW. 30
Figura 3.1 – PRO2PI-MMC – Metamodelo de Perfis de Capacidade de Processo. 40
Figura 3.2 – Geraes – Metamodelo de Conceitos Unificados. 41
Figura 3.3 – Metamodelo de Arquitetura e Metamodelo de Regra Adicional de Transformação de Arquitetura. 44
Figura 3.4 – Relação de conformidade entre as arquiteturas CMMI e MPS.BR e o PRO2PI-MMC. 46
Figura 3.5 – Relação de conformidade entre a área de processo REQM e a arquitetura CMMI. 50
Figura 4.1 – Contexto e arquitetura do PRO2PI-MMCA. 59
Figura 4.2 – Diagrama de sequência do PRO2PI-MMCA. 60
Figura 4.3 – Modelo de Dados do PRO2PI-MMCA. 62
Figura 4.4 – Arquitetura do Geraes. 67
Figura 4.5 – Transformações possíveis entre os metaelementos do Geraes. 69
Figura 4.6 – Relacionamento entre os metaelementos da arquitetura do Geraes e os elementos das arquiteturas dos modelos de capacidade. 70

Figura 4.7 – Regras gerais de transformação da arquitetura CMMI para a arquitetura SPICE. ... 71
Figura 4.8 – Regras de transformação da arquitetura CMMI para a arquitetura SPICE. 72
Figura 4.9 – Regras de transformação da arquitetura CMMI para a arquitetura MPS.BR. 72
Figura 4.10 – Regras de transformação da arquitetura MPS.BR para a arquitetura CMMI. 73
Figura 4.11 – Regras de transformação da arquitetura MPS.BR para a arquitetura SPICE. 74
Figura 4.12 – Regras de transformação da arquitetura SPICE para a arquitetura CMMI. 75
Figura 4.13 – Regras de transformação da arquitetura SPICE para a arquitetura MPS.BR. 76
Figura 4.14 – Cadastro da informação da arquitetura CMMI. 77
Figura 4.15 – Informação das arquiteturas dos modelos cadastrados no PRO2PI-MMCA. 78
Figura 4.16 – Cadastro do elemento *Process Area Category* da arquitetura CMMI. 78
Figura 4.17 – Elementos da arquitetura CMMI. 79
Figura 4.18 – Cadastro de relacionamento entre os elementos *Process Area* e *Process Area Category* da arquitetura CMMI. 80
Figura 4.19 – Arquitetura CMMI. 81
Figura 4.20 – Arquitetura SPICE. 81
Figura 4.21 – Cadastro das informações do modelo CMMI-DEV versão 1.3. 82
Figura 4.22 – Consulta das informações do modelo CMMI-DEV versão 1.3. 83
Figura 4.23 – Tela de seleção de cadastro de componentes do modelo CMMI. 84
Figura 4.24 – Consulta do componente SG 1 do modelo CMMI-DEV versão 1.3. 85
Figura 4.25 – Consulta do relacionamento entre os componentes SP 1.1 e SG 1 do modelo CMMI-DEV versão 1.3. 86
Figura 4.26 – Consulta das regras de transformação entre as arquiteturas CMMI e SPICE. 87
Figura 5.1 – Esquema de Avaliação. 92
Figura 5.2 – Perfil de Capacidade de Processo de Engenharia. 94
Figura 5.3 – Formação Acadêmica dos Especialistas em Modelos de Capacidade de Processo. . 98
Figura 5.4 – Experiência dos Especialistas em Modelos de Capacidade de Processo. 99
Figura 5.5 – Avaliação da área de processo REQM do CMMI-DEV versão 1.3 na arquitetura SPICE. 100
Figura 5.6 – Classificação da justificativa da avaliação da área de processo REQM do CMMI-DEV versão 1.3 na arquitetura SPICE. 101

Figura 5.7 – Avaliação da área de processo REQM do CMMI-DEV versão 1.3 na arquitetura MPS.BR. 103
Figura 5.8 – Classificação das justificativas da avaliação da área de processo REQM do CMMI-DEV versão 1.3 na arquitetura MPS.BR. 104
Figura 5.9 – Avaliação do processo DRE do MR-MPS-SW versão agosto/2012 na arquitetura CMMI. 106
Figura 5.10 – Classificação das justificativas da avaliação do processo DRE do MR-MPS-SW versão agosto/2012 na arquitetura CMMI. 107
Figura 5.11 – Avaliação do processo DRE do MR-MPS-SW versão agosto/2012 na arquitetura SPICE. 109
Figura 5.12 – Classificação da justificativa da avaliação do processo DRE do MR-MPS-SW versão agosto/2012 na arquitetura SPICE. 110
Figura 5.13 – Avaliação do processo ENG.1 do ASPICE versão 2.5 na arquitetura CMMI. 112
Figura 5.14 – Classificação da justificativa da avaliação do processo ENG.1 do ASPICE versão 2.5 na arquitetura CMMI. 113
Figura 5.15 – Avaliação do processo ENG.1 do ASPICE versão 2.5 na arquitetura MPS.BR. .. 115
Figura 5.16 – Classificação da justificativa da avaliação do processo ENG.1 do ASPICE versão 2.5 na arquitetura MPS.BR. 116

Lista de quadros

Quadro 4.1 – Dicionário de dados do banco de dados do PRO2PI-MMCA. 63
Quadro 4.2 – Finalidade das tabelas do PRO2PI-MMCA. 66
Quadro 4.3 – REQM – *Requirements Management* do modelo CMMI-DEV versão 1.3 na arquitetura SPICE. 88
Quadro 5.1 – Área de processo "REQM – *Requirements Management*" do modelo CMMI-DEV versão 1.3 expressa na arquitetura MPS.BR. 95
Quadro 5.2 – Questionário utilizado para avaliar a área de processo REQM do CMMI-DEV versão 1.3 na arquitetura MPS.BR. 97
Quadro 5.3 – Comentários gerais da área de processo REQM do CMMI-DEV versão 1.3 na arquitetura SPICE. 102
Quadro 5.4 – Comentários gerais da área de processo REQM do CMMI-DEV versão 1.3 na arquitetura MPS.BR. 105
Quadro 5.5 – Comentários gerais do processo DRE do MR-MPS-SW versão agosto/2012 na arquitetura CMMI. 108
Quadro 5.6 – Comentários gerais do processo DRE do MR-MPS-SW versão agosto/2012 na arquitetura SPICE. 111
Quadro 5.7 – Comentários gerais do processo ENG.1 do ASPICE versão 2.5 na arquitetura CMMI. 114
Quadro 5.8 – Comentários gerais do processo ENG.1 do ASPICE versão 2.5 na arquitetura MPS.BR. 117

Lista de tabelas

Tabela 2.1 – Escopo da Revisão Sistemática da Literatura. 12
Tabela 2.2 – Protocolo da Revisão Sistemática da Literatura. 13
Tabela 2.3 – Classificação dos artigos da revisão sistemática. 15
Tabela 2.4 – Exemplos de transformação de modelos. 19
Tabela 3.1 – Modelagem da arquitetura CMMI via PRO2PI-MMC. 47
Tabela 3.2 – Modelagem da arquitetura MPS.BR via PRO2PI-MMC. 49
Tabela 3.3 – Modelagem da área de processo REQM via arquitetura CMMI. 51
Tabela 3.4 – Relacionamento entre as arquiteturas CMMI e MPS.BR via PRO2PI-MMC. 52
Tabela 3.5 – Modelagem das regras adicionais entre as arquiteturas CMMI e MPS.BR via PRO2PI-MMC. 53
Tabela 3.6 – Transformação de parte da área de processo REQM do CMMI-DEV versão 1.3 para a arquitetura MPS.BR. 54
Tabela 5.1 – Áreas de Processo/Processos e Níveis de Capacidade cadastrados no PRO2PI-MMCA. 93

Lista de abreviaturas e siglas

ASPICE	*Automotive Software Process Improvement and Capability dEtermination*
CMMI	*Capability Maturity Model Integration*
CMMI-ACQ	*Capability Maturity Model Integration for Acquisition*
CMMI-DEV	*Capability Maturity Model Integration for Development*
CMMI-SVC	*Capability Maturity Model Integration for Services*
DSL	*Domain Specific Language* – Linguagem Específica de Domínio
eSCM	*eSourcing Capability Model*
ISO/IEC	*International Organization for Standardization/ International Electrotechnical Commission*
MDA	*Model Driven Architecture* – Arquitetura Dirigida por Modelos
MDE	*Model Driven Engineering* – Engenharia Dirigida por Modelos
MoProSoft	Modelo de Processos para a Indústria de Software
MPS	Melhoria de Processo de Software
MPS.BR	Melhoria de Processo do Software Brasileiro
MR-MPS	Modelo de Referência de Melhoria de Processo de Software
MR-MPS-SV	Modelo de Referência de Melhoria de Processo do Software para Serviços
MR-MPS-SW	Modelo de Referência de Melhoria de Processo do Software para Software
OMG	*Object Management Group*
PHP	*Hypertext Preprocessor*
PICOC	*Population, Intervention, Comparison, Outcomes and Context*
PRO2PI	*Process Modeling Profile to drive Process Improvement*
PRO2PI-MMC	*Process Modeling Profile to drive Process Improvement MetaModel for Process Capability Profile*
PRO2PI-MMCA	*Process Modeling Profile to drive Process Improvement MetaModel for Process Capability Profile Application*
SPI	*Software Process Improvement*
SPICE	*Software Process Improvement and Capability dEtermination*
SW-CMM	*Software Capability Maturity Model*
UML	*Unified Modeling Language* – Linguagem de Modelagem Unificada
URL	*Uniform Resource Locator*

Trabalhos publicados pelo autor

Banhesse, Edgar L., Clenio F. Salviano, and Mario Jino. 2012a. “Integrando Elementos de Múltiplos Modelos com um Metamodelo de Perfis de Capacidade de Processo.” XI Simpósio Brasileiro de Qualidade de Software – SBQS, Junho 11-15, pp. 128-142.

Banhesse, Edgar L., Clenio F. Salviano, and Mario Jino. 2012b. “Towards a Metamodel for integrating Multiple Models for Process Improvement.” Euromicro Conference on Software Engineering and Advanced Applications, September 5-8, pp. 315-318.

Sumário

1 Introdução 1
1.1 Motivação e contribuição da pesquisa 3
1.2 Objetivo 6
1.3 Etapas do trabalho 6
1.4 Metodologia da pesquisa 7
1.5 Público alvo 9
1.6 Estrutura da dissertação 9
2 Fundamentação Teórica 11
2.1 Revisão Sistemática da Literatura 11
2.2 Engenharia Dirigida por Modelos 17
2.3 Transformação de Modelos 18
2.4 Modelos de Capacidade de Processo 19
2.5 Melhoria de Processo de Software 31
2.6 Abordagens para múltiplos modelos 32
2.7 Metodologia PRO2PI 34
3 PRO2PI-MMC – Metamodelo de Perfis de Capacidade de Processo 37
3.1 Estratégia de Desenvolvimento do Trabalho 37
3.2 Metamodelo de Perfis de Capacidade de Processo 39
3.3 Exemplo de Modelagem com o PRO2PI-MMC 45
4 Implementação do PRO2PI-MMCA: uma Prova de Conceito 55
4.1 Aplicação prática do PRO2PI-MMCA 55
4.2 Funcionalidades do PRO2PI-MMCA 56
4.3 Arquitetura do PRO2PI-MMCA 59
4.4 Modelo de dados do PRO2PI-MMCA 62
4.5 Regras de transformação das arquiteturas dos modelos 67
4.6 Exemplo de utilização do PRO2PI-MMCA 77
5 Avaliação dos resultados do PRO2PI-MMCA e análise da avaliação 91
5.1 Critérios de seleção dos especialistas para avaliação por Painel de Especialistas 91
5.2 Perfis de Capacidade de Processo avaliados 92
5.3 Questionário de avaliação 96

5.4 Resultado e análise do Painel de Especialistas 97
5.5 Ameaças à validade da avaliação 118
5.6 Discussão 118
6 Conclusões 121
6.1 Análise das etapas do trabalho 121
6.2 Resultados adicionais do trabalho 123
6.3 Trabalhos futuros 123
6.4 Conclusão 125
Apêndice A 135
Perfil de Capacidade de Processo de Múltiplos Modelos 135
Apêndice B 151
Questionário e resultados avaliados 151
Avaliação dos Perfis de Capacidade de Processo do PRO2PI-MMCA 151

"A vida é uma peça de teatro que não permite ensaios...
Por isso, cante, ria, dance, chore e viva intensamente cada momento de sua vida,
antes que a cortina se feche e a peça termine sem aplausos..."
(Charles Chaplin)

1 Introdução

A Melhoria de Processo de Software (MPS) em organizações intensivas em software está estabelecida na indústria como um meio bem sucedido para a melhoria do desenvolvimento de software (Unterkalmsteiner et al. 2012). A MPS tem sido orientada por um determinado modelo de capacidade (ou de maturidade da capacidade), geralmente um modelo do CMMI – *Capability Maturity Model Integration* (SEI 2010) ou um modelo baseado na Norma Internacional ISO/IEC 15504, também conhecido como SPICE – *Software Process Improvement and Capability dEtermination* (ISO/IEC 2006). No Brasil o modelo do MPS.BR – Melhoria de Processo do Software Brasileiro (SOFTEX 2012a) também tem sido utilizado.

Como consequência dessa melhora existe um conjunto de fatores que levam a uma evolução da atual MPS para tratar novas oportunidades de melhoria. Entre esses fatores existem: a diversidade de estratégia, o relacionamento entre modelo e processo, a generalização do software, a disseminação de processo, os princípios fundamentais dos modelos, a visão da ISO/IEC 15504 e os múltiplos modelos (Salviano 2009).

Em relação aos princípios fundamentais, Card (2004) identifica que "Apesar de ser um fator no estado da prática do setor de software, a área de melhoria de processo ainda não é um tópico popular de pesquisas mais rigorosas, principalmente nas universidades. Como a área tem sido impulsionada pela prática, os diferentes modelos e abordagens são considerados concorrentes, embora compartilhem os mesmos princípios básicos. A insuficiência de pesquisas nessa área tem limitado a identificação desses princípios básicos, que é papel da ciência".

Um grupo de especialistas em Melhoria de Processo de Software de diversas partes do mundo se reuniu em setembro de 2009 na conferência EuroSPI - *European System & Software Process Improvement and Innovation* para a realização de um workshop na Universidade de Alcalá na Espanha. Quinze especialistas apresentaram seus conhecimentos adquiridos ao longo de muitos anos de experiência com melhoria de processo. Com base nesses conhecimentos, em ja-

neiro de 2010 foi publicado um manifesto (Pries-Heje and Johansen 2010) que apresenta três princípios e dez valores para a Melhoria de Processo de Software, o que reforçou a necessidade já identificada por Card em 2004.

Salviano e Figueiredo (2008) propõem um conjunto de conceitos básicos unificados, que são princípios fundamentais para perfis e modelos de capacidade de processo, como base para a criação de um metamodelo para melhoria de processo com múltiplos modelos. Este conjunto de conceitos básicos unificados foi denominado Geraes. O Metamodelo de Perfis de Capacidade de Processo faz parte de uma proposta para evolução da atual MPS em direção a uma melhoria de processo dirigida por modelagem de processo. Metamodelo (Salviano 2011) nesse trabalho é considerado como um modelo de uma linguagem de modelos (Favre 2005); no caso, linguagem de modelos de capacidade de processo. Desta forma, o metamodelo visa promover uma utilização mais ampla de múltiplos modelos para a melhoria de processos de software.

O Geraes é um componente metodológico da metodologia PRO2PI – *Process Modeling Profile to drive Process Improvement* (Salviano 2011) e que se baseia no paradigma da MDE – *Model Driven Engineering* – Engenharia Dirigida por Modelos para elicitar e unificar os principais conceitos dos modelos de capacidade de processo e utilizá-los para gerar perfis de capacidade de processo. PRO2PI é uma metodologia de melhoria de processo baseada em múltiplos modelos.

Um Perfil de Capacidade de Processo é um conjunto de áreas de processos e seus correspondentes níveis de capacidade (SEI 2010). Neste trabalho, os perfis de capacidade de processo são gerados a partir de combinações e transformações entre modelos. Uma transformação entre modelos é a geração automática de um modelo destino a partir de um modelo de origem, segundo uma definição de transformação. Transformação é um conjunto de regras de transformação que juntas descrevem como um modelo na linguagem de origem pode ser transformado em um modelo na linguagem de destino. Regra de transformação é uma descrição de como uma ou mais construções em uma linguagem de origem podem ser transformadas em uma ou mais construções em uma linguagem de destino (Kleppe, Warmer, and Bast 2003).

Com o intuito de contribuir com a identificação dos princípios fundamentais dos modelos, e de tratar o fator referente ao uso de múltiplos modelos, este projeto de pesquisa apresenta uma proposta que leva em consideração os conceitos unificados dos modelos de capacidade representados pelo componente Geraes para especificar e desenvolver um Metamodelo de Perfis de Capa-

cidade de Processo. Este metamodelo deve ser capaz de representar e transformar elementos de múltiplos modelos que estão documentados em diferentes arquiteturas, para promover a integração destes elementos. Os modelos podem ter diferentes objetivos ou mesmo ter base em diferentes conceitos para Melhoria de Processo de Software em organizações intensivas em software. Este trabalho é facilitado pelo fato de os modelos compartilharem alguns princípios básicos e possuírem aspectos arquitetônicos semelhantes provenientes da ISO/IEC 15504 (SPICE).

Esta pesquisa e desenvolvimento foram realizados como uma cooperação entre a FEEC – UNICAMP – Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas e o CTI Renato Archer – Centro de Tecnologia da Informação Renato Archer. O CTI Renato Archer é uma Unidade de Pesquisa do Ministério da Ciência, Tecnologia e Inovação (MCTI) que desenvolve, entre outras iniciativas, uma Metodologia para Melhoria de Processo dirigida por Perfis de Modelagem de Processo (PRO2PI – *Process Modeling Profile to Process Improvement*) da qual esta pesquisa e desenvolvimento é parte integrante. Durante grande parte da pesquisa, o mestrando fez parte da equipe da DMPS – Divisão de Melhoria de Processo de Software do CTI. Esta pesquisa e desenvolvido também está relacionada com a participação do CTI Renato Archer no INCoD[1] que é o Instituto Nacional de Ciência e Tecnologia para Convergência Digital.

1.1 Motivação e contribuição da pesquisa

Os processos de desenvolvimento de software utilizados em organizações intensivas de software dependem do domínio de aplicação, da estrutura, do tamanho, dos objetivos e das necessidades de negócio da organização. Domínio de aplicação é um grupo de aplicações atuais e futuras que compartilham um conjunto comum de capacidades e dados. É uma classe de conhecimento, funções e características, que são comuns a uma família de sistemas (Armitage 1993). Exemplos de domínios são: domínio bancário, de companhias aéreas e de jornais (Rio and Abreu 2010). Os modelos são utilizados como referência na definição e melhoria de processos de desenvolvimento de software utilizados nas organizações. Porém os modelos são representações simplificadas do mundo; eles não contemplam todos os domínios de aplicação e todas as caracte-

[1] INCoD – http://www.incod.ufsc.br/

rísticas das organizações. Dessa forma existe a necessidade de utilizar múltiplos modelos para definir e melhorar o processo de desenvolvimento de software das organizações. A necessidade de se utilizar múltiplos modelos para definir e melhorar processos de desenvolvimento de software foi identificada também em outras pesquisas (Siviy, Penn, and Stoddard 2008), (Rocha et al. 2009), (Thiry, Zoucas, and Tristão 2010), (Enterprise SPICE 2010), (Unterkalmsteiner et al. 2012) e (Kelemen 2013).

No manifesto de Melhoria de Processo de Software (2010), um grupo de especialistas em Melhoria de Processo de Software de diversas partes do mundo reforçou a necessidade de utilizar múltiplos modelos no princípio "Foco no negócio" com o valor "Utilizar modelos dinâmicos e adaptáveis". Segundo o grupo de especialistas em Melhoria de Processo de Software, "A Melhoria de Processo de Software não é amarrada a nenhum modelo. MPS é primeira e principalmente ligada aos objetivos de negócio e as necessidades da organização. Modelos como CMMI, SPICE, Six Sigma ou normas como ISO 9001, técnicas como o SCRUM, métodos ágeis ou modelos de ciclo de vida como Cascata, Incremental, Espiral, Modelo V ou muitos outros podem conter contribuições valiosas para um esforço de melhoria. Entretanto, a experiência tem mostrado que em muitos casos, não se pode apenas seguir um modelo para obter os melhores resultados. Ao invés, os modelos e as ideias neles contidas podem e deveriam ser combinadas para alcançar os objetivos de negócio da melhor forma possível.".

Os modelos de capacidade (ou de maturidade da capacidade) de processo, tais como: CMMI-DEV – *Capability Maturity Model Integration for Development* (SEI 2010), o ASPICE – *Automotive Software Process Improvement Capability dEtermination* (SPICE 2010) e o MR-MPS-SW – Modelo de Referência de Melhoria de Processo do Software para Software (SOFTEX 2012a) estão documentados em uma linguagem específica de modelo, ou seja, possuem uma arquitetura específica, com um objetivo e com conceitos específicos. Estas características inerentes aos modelos dificultam a melhoria de processo de software com múltiplos modelos em organizações intensivas em software. Porém estes modelos compartilham alguns princípios básicos e possuem aspectos arquitetônicos semelhantes provenientes da Norma Internacional ISO/IEC 15504, também conhecido como SPICE – *Software Process Improvement and Capability dEtermination* (ISO/IEC 2006).

Conforme mencionado, existem semelhanças, porém as diferenças são significativas e não podem ser desconsideradas. Segundo o CMMI (SEI 2010), arquitetura de um modelo é o conjun-

to de estruturas necessárias para inferir sobre um modelo. Essas estruturas são compostas de elementos, das relações entre os elementos e das propriedades de ambos. Um componente de um modelo CMMI é qualquer um dos principais elementos arquitetônicos que compõem o modelo CMMI. Alguns dos principais elementos de um modelo CMMI incluem práticas específicas, práticas genéricas, metas específicas, metas genéricas, áreas de processo, níveis de capacidade e níveis de maturidade. Neste trabalho, a arquitetura de um modelo é utilizada para gerar perfis de capacidade de processo por meio de combinações e transformações entre modelos. Os perfis de capacidade são utilizados para melhoria de processo em organizações intensivas em software.

Com base na premissa de que o uso de múltiplos modelos ajuda a Melhoria de Processo de Software, pois ajudam a alcançar os objetivos de negócio da organização da melhor forma possível, surge a seguinte questão de pesquisa: Como promover a utilização integrada de elementos de múltiplos modelos, que estão documentados em diferentes arquiteturas, com diferentes objetivos e conceitos para melhoria de processo de software em organizações intensivas em software?

Para resolver esse problema foi formulada a seguinte hipótese de pesquisa: A utilização de um Metamodelo de Perfis de Capacidade de Processo como um modelo de uma linguagem de modelos, no caso, linguagem de modelos de capacidade de processo pode promover a utilização integrada de elementos de múltiplos modelos para a melhoria de processos de software em organizações intensivas em software, respeitando as características inerentes a cada modelo.

A criação de perfis de capacidade de processo baseada em múltiplos modelos traz benefícios para melhoria de processo de software nas organizações, tais como: redução de custos e melhoria da eficácia por meio de uma orientação integrada. Como exemplos de redução de custos, não ter a necessidade de lidar com melhorias simultâneas oriundas de diversos modelos, evitar a duplicação de esforço e retrabalhos. Como exemplo de melhoria da eficácia, alinhar os processos de negócios e processos técnicos com base em um único perfil de capacidade de processo.

Este projeto é a segunda fase de uma estratégia de três fases para a pesquisa e desenvolvimento de um Metamodelo de Perfis de Capacidade de Processo como um componente metodológico da metodologia PRO2PI. A primeira fase foi a pesquisa e desenvolvimento dos conceitos básicos unificados chamado de Geraes (Salviano and Figueiredo 2008). Este projeto evoluiu o componente Geraes da metodologia PRO2PI, que trata da necessidade levantada por Card (2004) referente à identificação dos princípios fundamentais dos modelos da área de melhoria de proces-

so de software; implementou um software como prova de conceito, que proporciona a utilização de forma dinâmica dos componentes dos modelos, respeitando suas características inerentes; definiu as regras de transformações utilizadas pelo software durante o processo de transformação de modelos e apresentou uma nova abordagem para se trabalhar com múltiplos modelos em melhoria de processos de software. O software permite a criação de Perfis de Capacidade de Processo, a partir da representação e transformação de elementos de múltiplos modelos para melhoria de processo de software. Estes perfis são utilizados para dirigir ciclos de melhoria de processos em organizações intensivas em software. A terceira fase irá tratar do relacionamento de elementos de múltiplos modelos para então permitir a integração dos elementos.

1.2 Objetivo

O objetivo deste projeto de pesquisa é propor e mostrar que é viável utilizar um Metamodelo de Perfis de Capacidade de Processo para representar e transformar elementos de múltiplos Modelos de Capacidade de Processo de Software com arquiteturas diferentes para promover a integração destes elementos.

1.3 Etapas do trabalho

Para alcançar o objetivo do trabalho, seis etapas (E) foram definidas:

E 1) Estudar a metodologia PRO2PI e suas referências conceituais (Salviano 2009) e (Salviano et al. 2010). O estudo da metodologia PRO2PI e de suas referências conceituais é muito importante para a realização do trabalho, pois o trabalho trata da evolução e do desenvolvimento de componentes metodológicos da metodologia PRO2PI, que é uma das bases do trabalho.

E 2) Identificar novas referências conceituais para a realização do trabalho e fazer uma revisão sistemática da literatura para entender como desenvolver e utilizar metamodelos como um meio na resolução de problemas no paradigma da MDE.

E 3) Evoluir o componente metodológico Geraes do PRO2PI de sua versão 2008, que representa os conceitos básicos unificados de perfis e de modelos de capacidade de processo (Salviano and Figueiredo 2008), para um Metamodelo de Perfis de Capacidade de Processo. A

versão 2008 do componente Geraes foi exercitada para elementos do modelo CMMI-DEV versão 1.2 (SEI 2006) e do modelo ISO/IEC 15504-5 versão 2005 (ISO/IEC 2005). Além disso, a evolução do Geraes visa contemplar os conceitos básicos de outros modelos, tais como: o ASPICE versão 2.5 (SPICE 2010), o MR-MPS-SW versão agosto/2012 (SOFTEX 2012a), o CMMI-DEV versão 1.3 (SEI 2010) e o modelo ISO/IEC 15504-5 versão 2006 (ISO/IEC 2006).

E 4) Desenvolver uma ferramenta (software), como prova de conceito que implemente o Metamodelo de Perfis de Capacidade de Processo para representação de múltiplos modelos nos termos do metamodelo e para a transformação de elementos para perfis de capacidade de processo a partir da integração dos elementos destes múltiplos modelos.

E 5) Aplicar a ferramenta de geração de perfis de capacidade de processo para melhoria de processo de software, após povoar o banco de dados do software com modelos representativos.

E 6) Analisar os resultados obtidos e consolidar a documentação do projeto de pesquisa.

As etapas 1 e 2 são utilizados como preparação para a etapa 3. A etapa 3 (metamodelo) representa o resultado principal da pesquisa. As etapas 4 e 5 são validações do metamodelo.

1.4 Metodologia da pesquisa

No início do projeto de pesquisa, no último trimestre de 2009, existiam algumas questões a responder, que se não respondidas logo no início do projeto poderiam inviabilizar a pesquisa. As duas principais questões a serem respondidas eram: 1) Como implementar o metamodelo de perfis de capacidade de processo? e 2) Como exercitar o metamodelo de perfis de capacidade de processo? Depois de algumas discussões, chegou-se à conclusão de que para responder estas questões seria necessário implementar o protótipo de um software que permitisse exercitar o metamodelo de perfis de capacidade.

Baseado nessa ideia as seguintes etapas foram realizadas: o Geraes passou por uma revisão inicial em 2010 para atender parte dos conceitos dos modelos eSCM – *eSourcing Capability Model* (Hyder, Heston, and Paulk 2004), MoProSoft – Modelo de Processos para a Indústria de Software (Oktaba et al. 2005) e MR-MPS – Modelo de Referência para Melhoria de Processo de Software (SOFTEX 2009) e as principais funcionalidades do software foram identificadas e des-

critas. A partir das funcionalidades, um banco de dados relacional e algumas consultas foram implementadas no sistema gerenciador de banco de dados relacional MySQL versão 5.1 e para exercitar o banco de dados foi utilizado o aplicativo phpMyAdmin versão 3.1.3, ambos para Windows XP. Esse protótipo de banco de dados permitiu a realização de testes referentes à transformação de modelos e também permitiu a geração de um perfil, além de ajudar a completar e a melhorar as funcionalidades do software. Essa fase inicial referente à implementação do protótipo de banco de dados e a realização de testes foi denominada fase exploratória.

Uma metodologia sistemática baseada em classificações foi adotada para atingir o objetivo proposto e para conduzir o projeto de pesquisa (Silva and Menezes 2005). A natureza do projeto foi classificada como Pesquisa Aplicada, pois objetiva gerar conhecimentos para aplicações práticas e dirigidos à solução de problemas específicos referentes à melhoria de processo de software. A forma de abordagem do problema foi classificada como Pesquisa Qualitativa, pois não requer o uso de métodos e técnicas estatísticas. O ambiente fornecido pelo software do metamodelo de perfis de capacidade é a fonte direta para coleta de dados e o pesquisador é o instrumento-chave.

O objetivo do projeto se enquadra como Pesquisa Exploratória, pois visam proporcionar maior familiaridade com o problema para torná-lo explícito e para construir hipóteses. Envolve levantamento bibliográfico, consultar especialistas em modelos que tiveram experiências práticas com o problema pesquisado e a análise de exemplos que estimulem a compreensão do problema. A classificação do projeto de pesquisa quanto ao Método Científico é Hipotético-Dedutivo, pois os conhecimentos disponíveis sobre a melhoria de processo de software a partir de múltiplos modelos são insuficientes e demandam por pesquisa.

Em relação ao procedimento para a elaboração do trabalho, o projeto de pesquisa foi realizado em quatro ciclos evolutivos. Cada ciclo tinha uma ênfase diferente e foi composto por atividades realizadas em iterações sucessivas. O primeiro ciclo tinha ênfase nos estudos das bases conceituais, o segundo ciclo no desenvolvimento do software do metamodelo de perfis de capacidade de processo, o terceiro ciclo no uso do software e o quarto ciclo na validação e documentação dos resultados do projeto de pesquisa.

1.5 Público alvo

Este trabalho possui como público alvo empresas de desenvolvimento de software que tenham por objetivo melhorar os seus processos de software. O metamodelo de perfis de capacidade de processo pode ser utilizado por grupos de melhoria de processos que buscam atender os objetivos e as necessidades da organização por meio da melhoria de processo de software e que precisam gerar perfis de capacidade de processo a partir de múltiplos modelos para implementar seu ciclo de melhoria.

O metamodelo também pode ser utilizado por grupos de melhoria de processo que desenvolvem modelos que serão utilizados em organizações intensivas em software para melhoria de processo de software.

1.6 Estrutura da dissertação

Esta dissertação está organizada em seis capítulos. No Capítulo 1 é abordado como a atual Melhoria de Processo de Software está estabelecida e quais os fatores que levam à sua evolução. A motivação apresenta a necessidade do uso de múltiplos modelos na melhoria de processo de software e a contribuição cita alguns resultados importantes obtidos no projeto. As etapas detalham os principais alvos a serem alcançados pelo projeto e a metodologia apresenta como a pesquisa foi conduzida.

O Capítulo 2 descreve a fundamentação teórica, que inclui a revisão sistemática da literatura para entender como desenvolver e utilizar metamodelos como um meio na resolução de problemas no paradigma da Engenharia Dirigida por Modelos, além dos conceitos e teorias relevantes ao trabalho, tais como: a Engenharia Dirigida por Modelos, a Transformação de Modelos, os Modelos de Capacidade de Processo utilizados no trabalho: CMMI-DEV, ASPICE e o MR-MPS-SW, a Melhoria de Processo de Software, as Abordagens para múltiplos modelos e a Metodologia PRO2PI.

No Capítulo 3 é apresentada a estratégia utilizada para desenvolver o trabalho, os passos do desenvolvimento do PRO2PI-MMC – Metamodelo de Perfis de Capacidade de Processo, os

três componentes que compõem o PRO2PI-MMC, um detalhamento de cada componente e um exemplo de modelagem com o PRO2PI-MMC.

O Capítulo 4 apresenta a implementação do software PRO2PI-MMCA como prova de conceito, as suas funcionalidades, sua arquitetura e o modelo de dados, as regras de transformação das arquiteturas dos modelos e um exemplo de utilização do software.

O Capítulo 5 apresenta as avaliações dos resultados do PRO2PI-MMCA. É composto pelos critérios de seleção dos especialistas em modelos para formar o painel de especialistas, por uma visão geral dos perfis de capacidade de processo que foram avaliados, pelos resultados obtidos nas avaliações, pela análise do painel de especialistas, pelas ameaças à validade da avaliação e pela discussão.

O Capítulo 6 apresenta as conclusões da pesquisa, a análise das etapas do trabalho, resultados adicionais e trabalhos futuros.

O Apêndice A apresenta um perfil de capacidade de processo de múltiplos modelos, que faz uso da arquitetura CMMI.

O Apêndice B apresenta o questionário e os resultados do PRO2PI-MMCA enviados aos especialistas em modelos durante o processo de avaliação.

"Aquele que duvida e não pesquisa torna-se não só infeliz, mas também injusto."

(Blaise Pascal)

2 Fundamentação Teórica

Este capítulo apresenta a revisão sistemática da literatura, desenvolvida com o intuito de apoiar e verificar a viabilidade da hipótese proposta para o projeto de pesquisa. A revisão serviu como base para entender como os metamodelos são desenvolvidos e utilizados no contexto da MDE. O capítulo apresenta também as referências conceituais e teóricas utilizadas como base para o desenvolvimento do trabalho. As referências são: a Engenharia Dirigida por Modelos, a Transformação de Modelos, os Modelos de Capacidade de Processo, a Melhoria de Processo de Software, as Abordagens para múltiplos modelos e a Metodologia PRO2PI. Além dessas referências, o trabalho utilizou quatorze referências bibliográficas diretas e sete indiretas encontradas no processo de revisão sistemática da literatura.

2.1 Revisão Sistemática da Literatura

Uma revisão sistemática da literatura foi desenvolvida de acordo com o processo proposto por Brereton et al. (2006) e Kitchenham (2007), com o intuito de apoiar e verificar a viabilidade da hipótese proposta para o projeto de pesquisa. Esse processo de revisão sistemática é composto por três fases: planejamento, realização e apresentação dos resultados da revisão sistemática da literatura. Segundo Petticrew e Roberts (2006), uma revisão sistemática da literatura é mais adequada para o propósito de responder perguntas específicas e testar hipóteses do que uma revisão tradicional.

A fase de planejamento da revisão sistemática da literatura é composta pelas atividades: especificar a questão de pesquisa, desenvolver o protocolo de revisão e validar o protocolo de revisão. A fase de realização é onde a revisão sistemática é conduzida; é composta pelas atividades: identificar fontes relevantes, selecionar estudos primários, avaliar a qualidade dos estudos, extrair os dados necessários e sintetizar os dados. E, por último, a fase de apresentação dos resul-

tados da revisão sistemática envolve documentar a revisão; é composta pelas atividades: escrever o relatório da revisão e validar o relatório da revisão.

A questão de pesquisa da revisão sistemática da literatura, elaborada com base na hipótese do trabalho de pesquisa é: *Como desenvolver e utilizar metamodelos como um meio para resolver problemas no paradigma da engenharia dirigida por modelos (MDE)*? O escopo da revisão sistemática apresentado na Tabela 2.1 foi definido de acordo com o processo PICOC do Guia Prático de Revisões Sistemáticas em Ciências Sociais (Petticrew and Roberts 2006). O processo é denominado PICOC, pois considera cinco componentes: *Population*, *Intervention*, *Comparison*, *Outcomes* e *Context* para definir o escopo da revisão sistemática.

Tabela 2.1 – Escopo da Revisão Sistemática da Literatura.

Componente	**Descrição**
População (*Population*)	Organizações ou Grupos de Pesquisas interessados em Melhoria de Processo de Software.
Intervenção (*Intervention*)	Desenvolvimento e utilização de metamodelos no paradigma da MDE.
Comparação (*Comparison*)	Avaliação e/ou Melhoria de Processo de Software.
Resultados (*Outcomes*)	Metamodelos desenvolvidos e utilizados no paradigma da MDE.
Contexto (*Context*)	Publicações realizadas a partir de janeiro de 1995.

O protocolo da revisão sistemática segundo Kitchenham (2007) é apresentado na Tabela 2.2. O protocolo da revisão sistemática foi validado por meio de seus resultados, uma vez que retornou artigos relevantes do paradigma da Engenharia Dirigida por Modelos. As fontes de pesquisa: IEEE Xplore, ACM Digital Library e ScienceDirect utilizadas no processo de revisão sistemática foram escolhidas baseado no trabalho de Brereton et al. (2006) e Hauck (2011).

Tabela 2.2 – Protocolo da Revisão Sistemática da Literatura.

Passo	Descrição
Background	Mostrar a viabilidade da utilização de um Metamodelo de Perfis de Capacidade de Processo para a Melhoria de Processo de Software a partir de Múltiplos Modelos em organizações intensivas em software.
Questão de Pesquisa	Como desenvolver e utilizar metamodelos como um meio para resolver problemas no paradigma da engenharia dirigida por modelos (MDE)?
Estratégia de Pesquisa	Fontes de Pesquisa: IEEE Xplore, ACM Digital Library e ScienceDirect. Termos de Pesquisa: Para IEEE Xplore, ACM Digital Library e ScienceDirect: (“*industry*” OR ”*company*” OR ”*organization*” OR “*business model*“ OR “*business domain*“) AND (”MDE” OR ”*model-driven engineering*” OR ”MDA” OR ”*model-driven architecture*” OR ”MDD” OR ”*model-driven development*”) AND (”*assessment*” OR ”*improvement*” OR ”*capability*” OR ”*maturity*” OR ”*development process*” OR “*software process*”) AND (”*metamodel*” OR ”*meta-model*” OR ”*meta model*” OR ”*metamodel-based*”) published since 1995
Critérios Primários de Seleção	1) Artigos publicados em inglês sobre metamodelos em MDE disponíveis em bases confiáveis e validadas na web, a partir de janeiro de 1995. 2) Somente artigos revisados e publicados em revistas ou anais de conferências. 3) Excluir as publicações que não descrevem como os metamodelos foram elaborados.
Critérios de Qualidade	1) O metamodelo é detalhadamente apresentado? 2) São apresentados detalhes da elaboração do metamodelo? 3) São apresentados detalhes do desenvolvimento do metamodelo? 4) O metamodelo foi utilizado como um meio para resolver um problema no paradigma da MDE?

As palavras chave de pesquisa utilizadas nos termos de pesquisa apresentadas na Tabela 2.2 foram extraídas dos termos constantes na questão de pesquisa do protocolo da revisão sistemática da literatura, segundo Brereton et al. (2006) e Kitchenham (2007). Além disso, foram i-

dentificados ao menos três sinônimos para cada um dos critérios PICOC (Petticrew and Roberts 2006). Os sinônimos de cada um dos critérios PICOC são associados utilizando "OR" e os termos de pesquisa são associados em um só termo utilizando "AND". Os termos são utilizados em inglês para permitir a busca nas fontes de pesquisa internacionais e tem sua sintaxe adaptada (Brereton et al. 2006).

2.1.1 Resultados da Revisão Sistemática da Literatura

Realizada a pesquisa com os termos de pesquisa nas fontes escolhidas, obteve-se o retorno de 1542 artigos, 878 da IEEE Xplore, 328 da ACM Digital Library e 336 da ScienceDirect. Uma verificação inicial entre os artigos das três fontes de pesquisa permitiu identificar e eliminar 50 artigos duplicados, o que resultou num total de 1492 artigos únicos.

Em uma primeira etapa os títulos, os resumos, as introduções e as conclusões dos artigos foram revisados com base nos critérios primários de seleção e critérios de qualidade contidos no protocolo da revisão sistemática. Essa revisão resultou na seleção de 14 artigos. Os artigos selecionados foram lidos e classificados de acordo com três grupos de utilização do metamodelo, que são: Criação de Modelos, Desenvolvimento e Linguagem de Transformação/Modelagem. Além da classificação dos artigos, foram selecionados mais 7 artigos indiretos para o trabalho, ou seja, artigos selecionados com base nas referências dos 14 artigos diretos selecionados na revisão sistemática.

Os grupos utilizados na classificação dos artigos foram identificados durante o processo de leitura e revisão dos artigos. A revisão sistemática da literatura permitiu responder a questão de pesquisa do protocolo da revisão sistemática, além de confirmar que a hipótese proposta para o projeto de pesquisa é viável.

Isso foi constatado pelo fato dos metamodelos apresentados nos artigos terem sido desenvolvidos e utilizados como um meio, ou seja, como um instrumento na resolução de problemas no paradigma da engenharia dirigida por modelos (MDE) com diferentes focos de utilização. Verificou-se também que um metamodelo de perfis de capacidade de processo ainda não havia sido proposto e desenvolvido.

A Figura 2.1 apresenta a classificação dos artigos baseado na utilização do metamodelo.

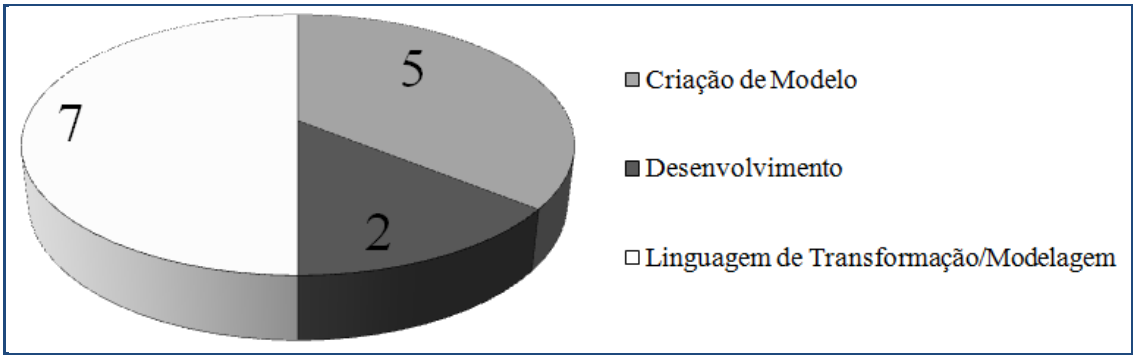


Figura 2.1 – Classificação dos artigos baseada na utilização do Metamodelo.

A Tabela 2.3 apresenta a classificação detalhada dos artigos selecionados no processo de revisão sistemática da literatura.

Tabela 2.3 – Classificação dos artigos da revisão sistemática.

Id.	Classificação	Referência do Artigo Direto	Referência do Artigo Indireto
1	1	(Atkinson, Gutheil, and Kennel 2009)	--
2	1	(Chen et al. 2006)	--
3	1	(Henderson-Sellers 2010)	(Bézivin 2005), (Favre 2004a) e (Favre 2005)
4	1	(Favre 2004b)	--
5	1	(Passerone et al. 2009)	--
6	1	(Kurtev et al. 2006)	(Jouault and Bézivin 2006)
7	1	(Atkinson and Kühne 2003)	--
8	2	(Monperrus, Beugnard, and Champeau 2009)	--
9	2	(Zhang et al. 2008)	--
10	2	(Shiliang and Qin 2010)	--
11	2	(Mohagheghi, Dehlen, and Neple 2009)	(Mohagheghi and Dehlen 2008)
12	2	(Bézivin and Gerbé 2001)	--
13	3	(Gutierrez et al. 2009)	--
14	3	(Cruz and Faria 2010)	(Cruz and Faria 2008) e (Cruz and Faria 2009)

Legenda: 1 Linguagem de Transformação/Modelagem, 2 Criação de Modelo e 3 Desenvolvimento.

Existe uma relação muito forte entre os três grupos: Criação de Modelo, Desenvolvimento e Linguagem de Transformação/Modelagem; porém, cada artigo tinha um foco diferente no desenvolvimento e na utilização do metamodelo para resolver um determinado problema.

O grupo de Criação de Modelo trata do uso de metamodelos para criar modelos que podem ser utilizados em diferentes contextos; como exemplo foi possível identificar nos artigos deste grupo metamodelos utilizados para a união de outros modelos, para a especificação de modelos de qualidade e para a definição de modelos de negócio.

O grupo de Desenvolvimento trata do uso de metamodelos para desenvolver software e casos de testes; como exemplo foi possível identificar artigos neste grupo que utilizavam metamodelos para automatizar a geração de interfaces de usuário e para a geração de casos de testes.

E, por último, o grupo de Linguagem de Transformação/Modelagem trata do uso de metamodelos para a criação de linguagens de transformação de modelos e linguagens de modelagem, que são utilizadas para transformar um modelo de origem em um modelo de destino para resolver um problema específico. Este trabalho faz parte deste grupo.

A revisão sistemática mostrou que os metamodelos são elaborados e representados por diagramas de classes em UML e são desenvolvidos em ambientes computacionais que ainda estão em evolução para suportar de forma completa e amigável todo o paradigma da Engenharia Dirigida por Modelos.

2.2 Engenharia Dirigida por Modelos

Segundo Rossini et al. (2011), "Engenharia Dirigida por Modelos – *Model Driven Engineering* – MDE é um ramo da engenharia de software que visa a melhorar a produtividade, qualidade e custo-efetividade de software, mudando do paradigma centrado em código para o centrado em modelo. MDE promove modelos e linguagens de modelagem como artefatos principais do processo de desenvolvimento e a transformação de modelos como a técnica principal para gerar (partes de) sistemas de software a partir de modelos. Modelos permitem que os desenvolvedores raciocinem em um nível maior de abstração, enquanto a transformação de modelos alivia os desenvolvedores de tarefas repetitivas e propensa a erros, tais como codificação".

A MDE está baseada em três conceitos fundamentais: sistema, modelo e metamodelo, e em duas relações básicas: conformidade e representação. A relação básica de representação surge por meio da relação dos conceitos fundamentais de sistema e modelo; em MDE diz-se que um modelo é a representação de um sistema. Já a relação básica de conformidade surge por meio da relação entre modelo e metamodelo; em MDE diz-se que um modelo está em conformidade com o seu metamodelo. Um modelo está em conformidade com um metamodelo, se e somente se, cada elemento do modelo tem seu metaelemento definido no metamodelo (Bézivin 2006).

Em MDE Sistema ou Sistema em Estudo é o objeto alvo da investigação. Modelo é uma abstração de um sistema em estudo para um determinado propósito (Favre 2004a). Metamodelo define um acordo consensual sobre a forma como os elementos de um sistema devem ser selecionados para produzir um determinado modelo (Bézivin 2003). Um metamodelo define uma linguagem capaz de expressar modelos (Bézivin 2005), (Favre 2005) e (Seidewitz 2003).

Ao combinar os conceitos fundamentais e as relações básicas da MDE surge um padrão de conformidade. Um exemplo do padrão de conformidade é o relacionamento entre a língua inglesa, um dicionário de inglês e um texto escrito em inglês. O dicionário de inglês é um modelo da língua inglesa. Nesta relação a língua inglesa é o Sistema em Estudo. A língua inglesa pode ser definida em outra relação como uma linguagem de modelos da qual qualquer texto em inglês é um modelo que pertence a esta linguagem. Como consequência, o dicionário de inglês é um metamodelo do texto escrito em inglês. É importante notar que ser um sistema, modelo ou metamodelo não é uma característica intrínseca, mas sim uma função em uma relação.

Na MDE, segundo os autores Atkinson e Kühne (2002), (2003) e (2005) e Henderson-Sellers (2007) existem dois tipos de metamodelagem que devem ser considerados; a linguística (física) e a ontológica (lógica). A metamodelagem linguística utiliza metamodelos linguísticos e foca em entender e tratar "como" a informação é armazenada (forma), ao passo que a metamodelagem ontológica utiliza metamodelos ontológicos e foca em entender e tratar "qual" informação é armazenada (conteúdo). O dicionário de inglês é um exemplo de metamodelo ontológico, a gramática de inglês é um exemplo de metamodelo linguístico.

2.3 Transformação de Modelos

Segundo Kleppe, Warmer e Bast (2003), "Uma transformação de modelos é a geração automática de um modelo destino a partir de um modelo de origem, segundo uma definição de transformação. Uma definição de transformação é um conjunto de regras de transformação que juntas descrevem como um modelo na linguagem de origem pode ser transformado em um modelo na linguagem de destino. Regra de transformação é uma descrição de como uma ou mais construções em uma linguagem de origem podem ser transformadas em uma ou mais construções em uma linguagem de destino".

Existem quatro dimensões ortogonais de transformações de modelos: Endógena – quando a transformação entre modelos é expressa na mesma linguagem; Exógena – quando a transformação é expressa usando linguagens diferentes; Horizontal – quando a transformação possui o mesmo nível de abstração; e Vertical – quando a transformação possui diferentes níveis de abstração.

A Tabela 2.4 apresenta alguns exemplos de transformação de modelos (Mens, Czarnecki, and Gorp 2005).

Tabela 2.4 – Exemplos de transformação de modelos.

	Horizontal	**Vertical**
Endógena	Refatoração	Refinamento formal
Exógena	Migração de linguagem	Geração de código

Refatoração é uma alteração na estrutura interna do software para melhorar certas características da qualidade do software, tais como: manutenibilidade, reusabilidade e modularidade. Essa alteração não muda o comportamento observável do software. A migração de linguagem é a escrita de um determinado código fonte em outra linguagem de programação, porém mantendo o mesmo nível de abstração. O refinamento formal é o aperfeiçoamento gradual do código fonte de modo que o resultado final continue atendendo a especificação original usando a mesma linguagem de programação. A geração de código é a compilação de um código fonte para obtenção de um código executável.

2.4 Modelos de Capacidade de Processo

Modelos de capacidade de processo são coleções de boas práticas que ajudam as organizações a melhorar os seus processos (SEI 2010). Um modelo de capacidade de processo não é um processo. Os modelos são utilizados como referência para definir e melhorar os processos de desenvolvimento de software das organizações intensivas em software. Um modelo indica o que uma organização deve praticar para ser tida como uma boa organização em relação ao desenvolvimento de software. Os processos organizacionais baseados em modelos por sua vez indicam o que uma determinada organização deve fazer para desenvolver software. No contexto deste trabalho são utilizados três modelos relevantes: o CMMI-DEV versão 1.3 (SEI 2010), o ASPICE versão 2.5 (SPICE 2010) e o MR-MPS-SW versão de agosto/2012 (SOFTEX 2012a).

O modelo CMMI-DEV foi escolhido devido ao fato de ser um modelo reconhecido, disseminado e utilizado internacionalmente pela indústria e organizações intensivas em software para o domínio de engenharia de software. O modelo MR-MPS-SW foi escolhido, pois tem sido

muito utilizado e difundido no Brasil para o domínio de engenharia de software. Além de ser um modelo adequado para pequenas e médias empresas. O modelo ASPICE foi escolhido, pois faz uso da arquitetura e dos conceitos da ISO/IEC 15504 (SPICE) e tem sido utilizado para atender o domínio automotivo. Portanto, a combinação desses três modelos permitiu exercitar o metamodelo de perfis de capacidade de processo.

As próximas seções apresentam de forma geral os modelos e suas principais características.

2.4.1 CMMI-DEV – Capability Maturity Model Integration for Development

Os modelos CMMI – *Capability Maturity Model Integration* são coleções de boas práticas que ajudam as organizações a melhorar os seus processos. Estes modelos de capacidade são desenvolvidos por equipes de produto com membros da indústria, do governo e do Instituto de Engenharia de Software (SEI) da Universidade Carnegie Mellon. Existem três modelos: CMMI-SVC – CMMI *for Services*, CMMI-ACQ – CMMI *for Acquisition* e CMMI-DEV – CMMI *for Development*. O modelo CMMI-SVC versão 1.2 (SEI 2009) provê um guia para aplicar as boas práticas do CMMI para fornecer serviços, o CMMI-ACQ versão 1.2 (SEI 2007) prove um guia para aplicar as boas práticas do CMMI para adquirir produtos e serviços, e o CMMI-DEV versão 1.3 (SEI 2010) prove um guia para aplicar as boas práticas do CMMI em uma organização de desenvolvimento.

Essas práticas focam em atividades para desenvolver produtos e serviços de qualidade que atendam as necessidades dos clientes e usuários finais. Os modelos CMMI são produzidos a partir do Framework do CMMI, que contém todas as metas e práticas necessárias para produzir um modelo do CMMI. Estas metas e práticas estão organizadas em uma arquitetura específica, a arquitetura CMMI.

A Figura 2.2 apresenta a arquitetura CMMI, que é composta por componentes que estão relacionados para formar uma área de processo ou um nível de capacidade.

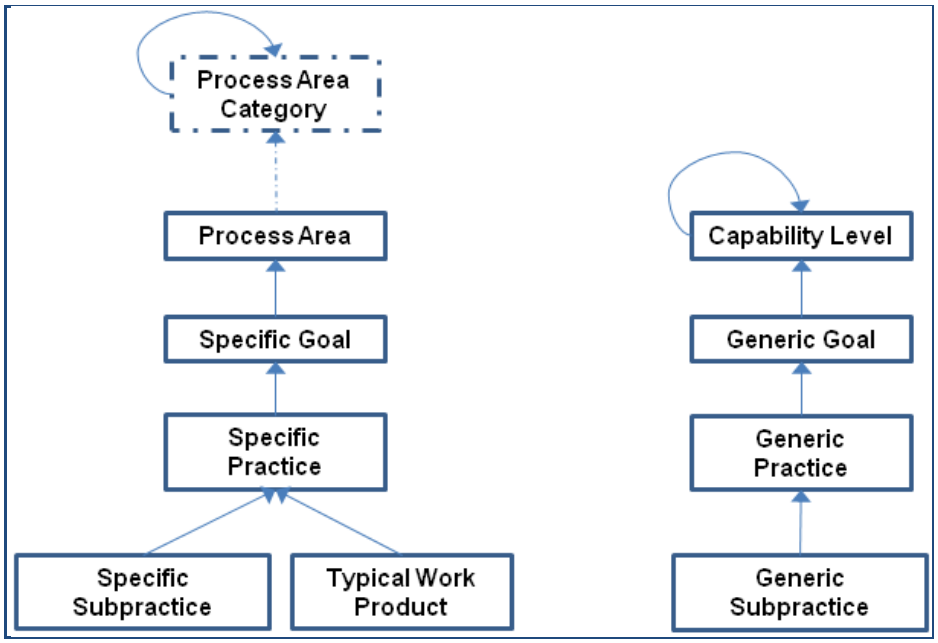


Figura 2.2 – Arquitetura CMMI.

O modelo CMMI-DEV é composto por 22 áreas de processo e quatro níveis de capacidade. Uma área de processo é um conjunto de práticas relacionadas em uma área que, quando implementadas coletivamente, satisfazem um conjunto de metas consideradas importantes para fazer melhorias naquela área. Um nível de capacidade é utilizado para medir as melhorias que são feitas dentro de uma área de processo específica.

O CMMI possui duas abordagens distintas para melhoria de processo: a representação contínua e a estagiada. A representação contínua é utilizada quando o foco da organização é melhorar um nível de capacidade. A estagiada é utilizada quando o foco da organização é melhorar um nível de maturidade. A Figura 2.3 apresenta os perfis de capacidade de processo do modelo CMMI-DEV versão 1.3. Um perfil de capacidade de processo é a relação entre uma ou mais áreas de processo e seus níveis de capacidade. A primeira coluna apresenta as 22 áreas de processo do modelo, a segunda coluna apresenta a abreviação das áreas de processo, a terceira coluna os níveis de maturidade e as demais colunas os níveis de capacidade.

Para obter o nível de maturidade 2, todas as áreas de processo referentes ao nível de maturidade 2 devem atingir o nível de capacidade 2 ou 3. Para obter o nível de maturidade 3, todas as

áreas de processo referentes ao nível de maturidade 2 e 3 devem atingir o nível de capacidade 3. Para obter o nível de maturidade 4, todas as áreas de processo referentes ao nível de maturidade 2, 3 e 4 devem atingir o nível de capacidade 3. Para atingir o nível de maturidade 5, todas as áreas de processo do modelo devem atingir o nível de capacidade 3. Um nível de maturidade é a obtenção da melhoria de processo quando um conjunto predefinido de áreas de processo atingem todas as suas metas específicas e genéricas.

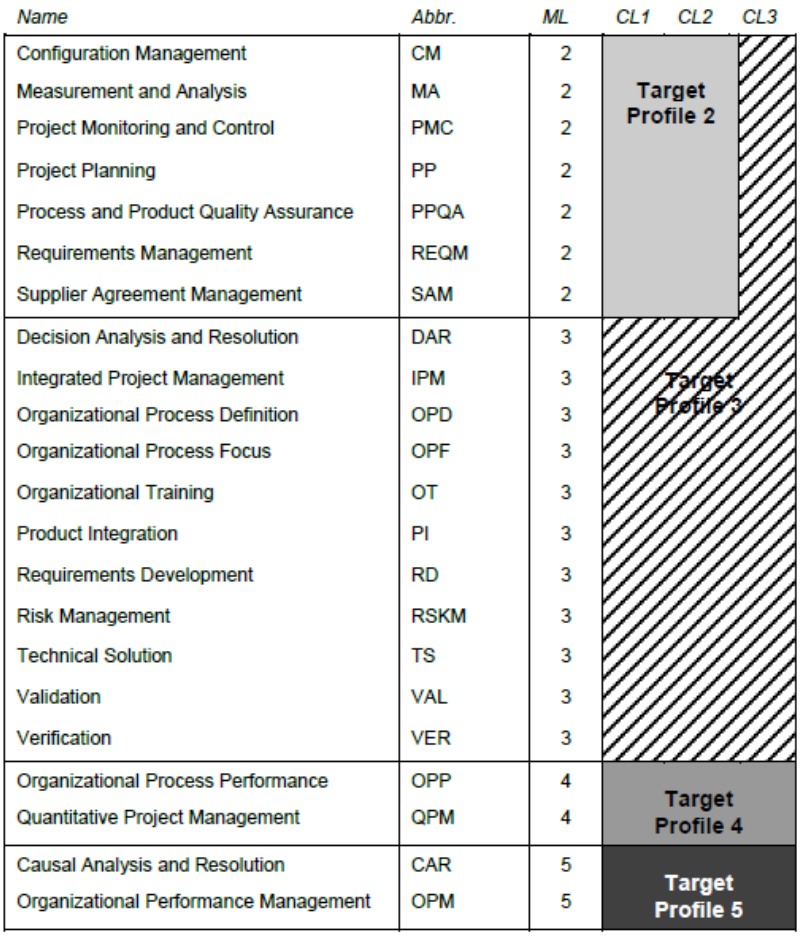


Figura 2.3 – Perfis do CMMI-DEV.[2]

[2] Extraído do CMMI-DEV versão 1.3 (SEI 2010).

A Figura 2.4 apresenta uma comparação entre os níveis de capacidade da representação contínua e os níveis de maturidade da representação estagiada.

Level	*Continuous Representation Capability Levels*	*Staged Representation Maturity Levels*
Level 0	Incomplete	
Level 1	Performed	Initial
Level 2	Managed	Managed
Level 3	Defined	Defined
Level 4		Quantitatively Managed
Level 5		Optimizing

Figura 2.4 – Comparação dos níveis do CMMI-DEV.[2]

Os níveis de capacidade são: Nível de capacidade 0 – Incompleto (*Incomplete*) – é um processo que não é realizado ou é parcialmente realizado. Nível de capacidade 1 – Realizado (*Performed*) – é um processo que executa o trabalho necessário para produzir os produtos de trabalho, o processo satisfaz as metas específicas da área de processo. Nível de capacidade 2 – Gerenciado (*Managed*) – é um processo realizado que é planejado e executado de acordo com uma política. Nível de capacidade 3 – Definido (*Defined*) – é um processo gerenciado que é customizado a partir de um conjunto de processos padrões da organização de acordo com os guias de customização da organização.

Em relação aos níveis de maturidade. Nível de maturidade 1 – Inicial (*Initial*) – é quando não existe um processo ou os processos são caóticos. Nível de maturidade 2 – Gerenciado (*Managed*) – os projetos baseiam-se em processos que são planejados e executados de acordo com uma política. Nível de maturidade 3 – Definido (*Defined*) – os processos são organizacionais e são bem caracterizados e entendidos, são descritos por meio de padrões, procedimentos, ferramentas e métodos. Nível de maturidade 4 – Gerenciado Quantitativamente (*Quantitatively Managed*) – a organização e os projetos estabelecem objetivos quantitativos para qualidade e para o desempenho do processo e utilizam estes objetivos como critérios na gestão dos projetos. Nível de maturidade 5 – Melhoria contínua (*Optimizing*) – a organização melhora continuamente seus

processos baseado no entendimento quantitativo dos seus objetivos de negócio e de suas necessidades de desempenho (SEI 2010).

2.4.2 ASPICE – Automotive SPICE

A Norma Internacional ISO/IEC 15504 *Information technology – Process assessment*, também conhecida como SPICE (*Software Process Improvement and Capability dEtermination*), é um conjunto de documentos normativos técnicos utilizados para definir e avaliar processos de desenvolvimento de software e funções de gestão de negócios relacionadas. Existem modelos, por exemplo, o Enterprise SPICE (Enterprise SPICE 2010) e o ASPICE (SPICE 2010), que foram definidos de acordo com a SPICE.

O modelo Enterprise SPICE reúne processos de outros modelos e normas amplamente aceitos em uma abordagem integrada para facilitar a avaliação e melhoria de processos baseada em múltiplos modelos. ASPICE – *Automotive Software Process Improvement Capability dEtermination* é um modelo específico utilizado para definir e avaliar os processos de desenvolvimento de software para a indústria automotiva. O modelo de avaliação de processo define duas dimensões de modelo de processo de capacidade baseadas na SPICE. Dimensão de processo e dimensão de capacidade.

Na dimensão de processo, os processos são definidos e classificados em categorias de processo. Cada categoria de processo é dividida em grupos de processo de acordo com o tipo de atividade. Na dimensão de capacidade, um conjunto de atributos de processo é agrupado para criar níveis de capacidade. Estes atributos de processo proveem características mensuráveis da capacidade de processo.

A Figura 2.5 apresenta as dimensões do modelo ASPICE versão 2.5.

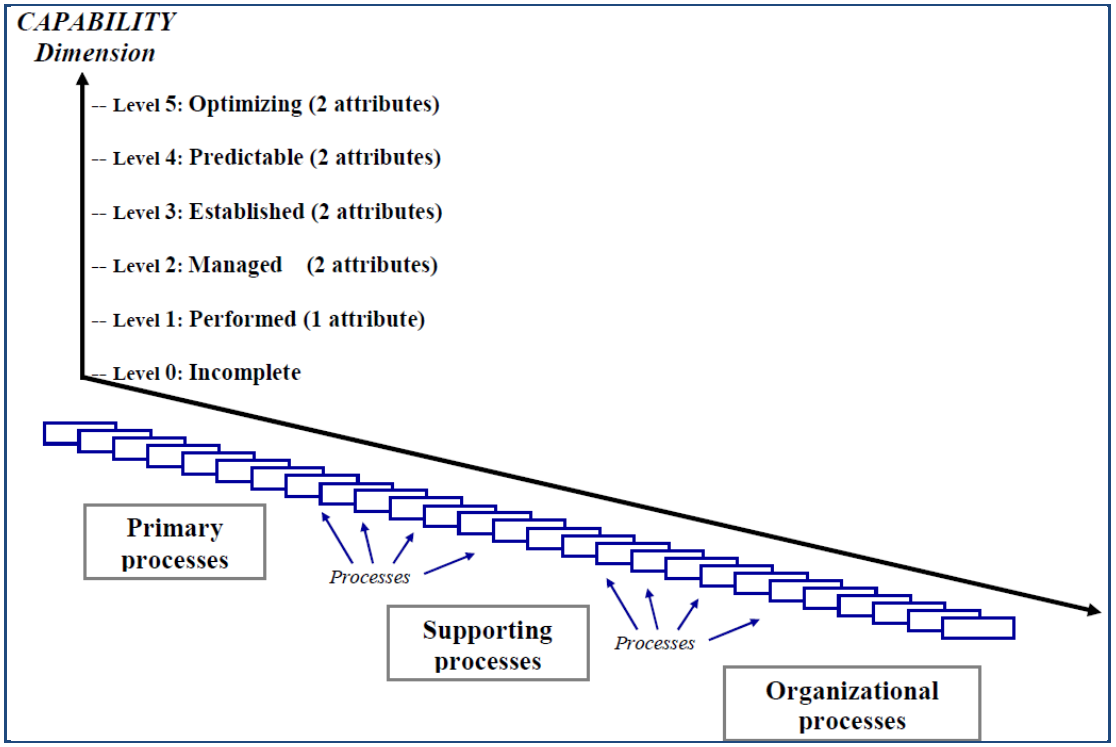


Figura 2.5 – Dimensões do modelo ASPICE.[3]

O modelo ASPICE também utiliza a arquitetura da Norma Internacional ISO/IEC 15504 (SPICE) para representar os componentes dos seus processos e dos níveis de capacidade.

[3] Extraído do ASPICE versão 2.5 (SPICE 2010).

A arquitetura SPICE utilizada pelo ASPICE é apresentada na Figura 2.6.

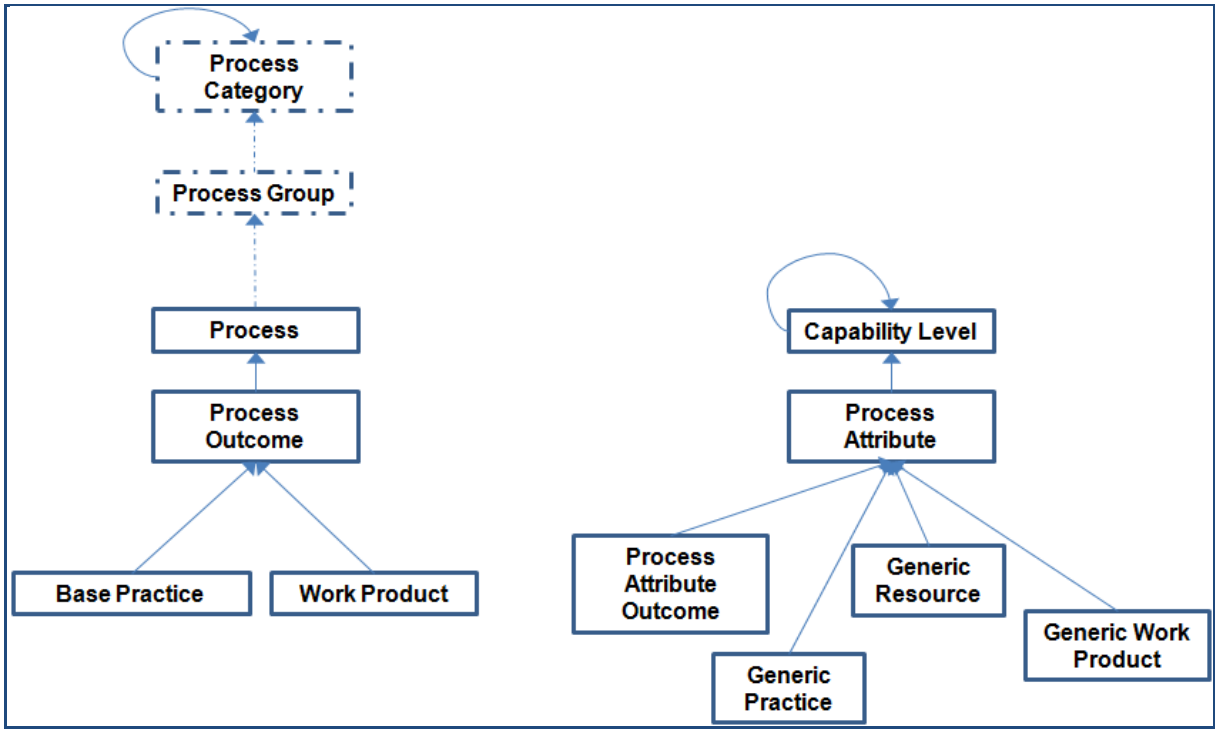


Figura 2.6 – Arquitetura SPICE.

O modelo ASPICE é composto por três categorias de processo: Processos Primários, Processos de Suporte e Processos Organizacionais. A categoria de processos primários é composta por três grupos de processos: grupo de aquisição (ACQ), grupo de fornecimento (SPL) e grupo de engenharia (ENG). O grupo de processo de aquisição é composto por sete processos conforme apresentado na Figura 2.7.

Process Identification	PRM Process name
ACQ.3	Contract agreement
ACQ.4	Supplier monitoring
ACQ.11	Technical requirements
ACQ.12	Legal and administrative requirements
ACQ.13	Project requirements
ACQ.14	Request for proposals
ACQ.15	Supplier qualification

Figura 2.7 – Processos do grupo de processos ACQ do modelo ASPICE.[3]

O grupo de processo de fornecimento é composto por dois processos conforme apresentado na Figura 2.8.

Process Identification	Process name
SPL.1	Supplier tendering
SPL.2	Product release

Figura 2.8 – Processos do grupo de processos SPL do modelo ASPICE.[3]

O grupo de processos de engenharia é composto por dez processos conforme apresentado na Figura 2.9.

Process Identification	Process name
ENG.1	Requirements elicitation
ENG.2	System requirements analysis
ENG.3	System architectural design
ENG.4	Software requirements analysis
ENG.5	Software design
ENG.6	Software construction
ENG.7	Software integration test
ENG.8	Software testing
ENG.9	System integration test
ENG.10	System testing

Figura 2.9 – Processos do grupo de processos ENG do modelo ASPICE.[3]

A categoria de processo de suporte é composta pelo grupo de processo de suporte (SUP). O grupo de suporte é composto por sete processos conforme apresentado na Figura 2.10.

Process Identification	Process name
SUP.1	Quality assurance
SUP.2	Verification
SUP.4	Joint review
SUP.7	Documentation
SUP.8	Configuration management
SUP.9	Problem resolution management
SUP.10	Change request management

Figura 2.10 – Processos do grupo de processos SUP do modelo ASPICE.[3]

A categoria de processo organizacional é composta por três grupos de processos: grupo de gestão (MAN), grupo de melhoria de processo (PIM) e grupo de reuso (REU). O grupo de processos de gestão é composto por três processos conforme apresentado na Figura 2.11.

Process Identification	Process name
MAN.3	Project management
MAN.5	Risk management
MAN.6	Measurement

Figura 2.11 – Processos do grupo de processos MAN do modelo ASPICE.[3]

O grupo de processos de melhoria de processo é composto pelo processo PIM.3, conforme apresentado na Figura 2.12.

Process Identification	Process name
PIM.3	Process improvement

Figura 2.12 – Processo do grupo de processos PIM do modelo ASPICE.[3]

O grupo de processos de reuso é composto pelo processo REU.2, conforme apresentado na Figura 2.13.

Process Identification	Process name
REU.2	Reuse program management

Figura 2.13 – Processo do grupo de processos REU do modelo ASPICE.[3]

Em relação aos níveis de capacidade, o modelo ASPICE possui seis níveis. Nível de capacidade 0 – Incompleto (*Incomplete*) – o processo não está implementado ou falha para atingir os seus propósitos. Nível de capacidade 1 – Realizado (*Performed*) – o processo implementado atinge os seus propósitos. Nível de capacidade 2 – Gerenciado (*Managed*) – o processo realizado é planejado, monitorado e ajustado e seus produtos de trabalho são apropriadamente estabelecidos, controlados e mantidos. Nível de capacidade 3 – Estabelecido (*Established*) – o processo gerenciado é implementado utilizando um processo definido que é capaz de atingir os seus resultados de processo. Nível de capacidade 4 – Previsível (*Predictable*) – o processo estabelecido passa a operar dentro de limites definidos para atingir os resultados de processo. Nível de capacidade 5 –

Otimizado (*Optimizing*) – o processo previsível é melhorado continuamente para atender os objetivos de negócio atuais e os projetados (SPICE 2010).

2.4.3 MR-MPS-SW – Modelo de Referência de Melhoria de Processo do Software para Software

O MPS.BR – Melhoria de Processo do Software Brasileiro é um programa mobilizador, de longo prazo, criado em dezembro de 2003, coordenado pela Associação para Promoção da Excelência do Software Brasileiro (SOFTEX), que conta com apoio do Ministério da Ciência, Tecnologia e Inovação (MCTI), Financiadora de Estudos e Projetos (FINEP), Serviço Brasileiro de Apoio às Micro e Pequenas Empresas (SEBRAE) e Banco Interamericano de Desenvolvimento (BID/FUMIN). O objetivo do programa MPS.BR é a Melhoria de Processo de Software e Serviços. Existem, por exemplo, dois modelos MPS.BR, o MR-MPS-SV (SOFTEX 2012b) e o MR-MPS-SW (SOFTEX 2012a). O MR-MPS-SV – Modelo de Referência de Melhoria de Processo do Software para Serviços é um guia destinado a organizações interessadas na melhoria de seus processos de serviços.

O MR-MPS-SW – Modelo de Referência de Melhoria de Processo do Software para Software está baseado nos conceitos de maturidade e capacidade de processo para a avaliação e melhoria da qualidade e produtividade de software e serviços correlatos e também para a melhoria da qualidade e produtividade dos serviços prestados. O modelo utiliza como referência a Norma Internacional ISO/IEC 15504 (SPICE) e o CMMI-DEV. A Figura 2.14 apresenta a arquitetura utilizada para representar os componentes dos processos e dos níveis de capacidade do modelo MR-MPS-SW.

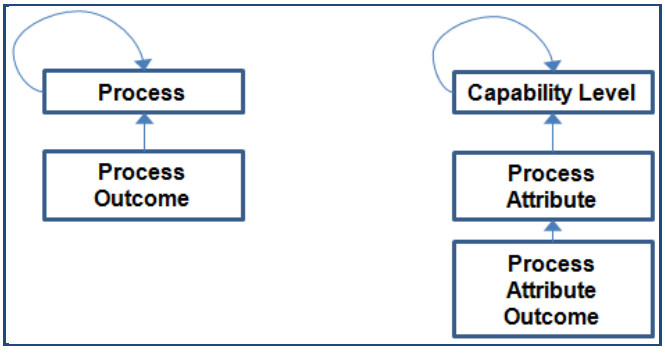


Figura 2.14 – Arquitetura MPS.BR.

A Figura 2.15 apresenta os sete níveis de maturidade com seus atributos de processo e os 19 processos do MR-MPS-SW. Os processos são descritos em termos de propósito e resultados esperados. O propósito descreve o objetivo geral a ser atingido durante a execução do processo. Os resultados esperados do processo estabelecem os resultados a serem obtidos com a efetiva implementação do processo. A capacidade do processo é representada por um conjunto de atributos de processo, que são resultados esperados que expressam o grau de refinamento e institucionalização com que o processo é executado na organização. À medida que a organização evolui nos níveis de maturidade, um maior nível de capacidade para desempenhar o processo deve ser atingido (SOFTEX 2012a).

Nível	Processos	Atributos de Processo
A		AP 1.1, AP 2.1, AP 2.2, AP 3.1, AP 3.2, AP 4.1, AP 4.2 , AP 5.1 e AP 5.2
B	Gerência de Projetos – GPR (evolução)	AP 1.1, AP 2.1, AP 2.2, AP 3.1 e AP 3.2, AP 4.1 e AP 4.2
C	Gerência de Riscos – GRI Desenvolvimento para Reutilização – DRU Gerência de Decisões – GDE	AP 1.1, AP 2.1, AP 2.2, AP 3.1 e AP 3.2
D	Verificação – VER Validação – VAL Projeto e Construção do Produto – PCP Integração do Produto – ITP Desenvolvimento de Requisitos – DRE	AP 1.1, AP 2.1, AP 2.2, AP 3.1 e AP 3.2
E	Gerência de Projetos – GPR (evolução) Gerência de Reutilização – GRU Gerência de Recursos Humanos – GRH Definição do Processo Organizacional – DFP Avaliação e Melhoria do Processo Organizacional – AMP	AP 1.1, AP 2.1, AP 2.2, AP 3.1 e AP 3.2
F	Medição – MED Garantia da Qualidade – GQA Gerência de Portfólio de Projetos – GPP Gerência de Configuração – GCO Aquisição – AQU	AP 1.1, AP 2.1 e AP 2.2
G	Gerência de Requisitos – GRE Gerência de Projetos – GPR	AP 1.1 e AP 2.1

Figura 2.15 – Níveis de maturidade, processos e atributos de processos do MR-MPS-SW.[4]

[4] Extraído do MR-MPS-SW versão agosto/2012 (SOFTEX 2012a).

2.5 Melhoria de Processo de Software

Processo é um conjunto de atividades inter-relacionadas que transformam entradas (insumos) em saídas (produtos) para atingir um determinado propósito (SEI 2010). Melhoria de Processo de Software (MPS) é uma abordagem sistemática para aumentar, por meio do estabelecimento e melhoria contínua dos processos, a eficiência e a eficácia de uma organização intensiva em desenvolvimento de software para melhorar os produtos de software (Unterkalmsteiner et al. 2012). De acordo com o CMMI-DEV, a ideia da MPS é: “Melhorar os processos para desenvolver melhores produtos e serviços”.

O manifesto de MPS (Pries-Heje and Johansen 2010) destaca três valores fundamentais para a MPS. As pessoas, o foco no negócio e a mudança organizacional. Pessoas – as pessoas devem ser envolvidas ativamente e suas atividades devem ser afetadas diariamente. Foco no negócio – é o que você faz para o sucesso do negócio, não para atender um modelo ou norma, alcançar um nível de maturidade ou obter um certificado. Mudança organizacional – MPS está intrinsecamente ligado com mudança, ela é inevitável. Para apoiar estes três valores, o manifesto de MPS desenvolveu dez princípios.

Para pessoas:

1) Conhecer a cultura e focar em suas necessidades,
2) Motivar todas as pessoas envolvidas,
3) Embasar a melhoria em experiência e medições, e
4) Criar uma organização que aprende.

Foco no negócio:

1) Suportar a visão da organização e seus objetivos de negócio,
2) Utilizar modelos dinâmicos e adaptáveis, e
3) Realizar a gestão de riscos.

Mudança organizacional:

1) Gerenciar a mudança organizacional no seu esforço de melhoria,
2) Assegurar que todas as partes entendam e concordem com o processo, e
3) Não perder o foco.

2.6 Abordagens para múltiplos modelos

O principal desafio adicional de uma Melhoria de Processo de Software com múltiplos modelos em relação à atual Melhoria de Processo de Software baseada em um único modelo, é a utilização integrada de elementos de múltiplos modelos que estão documentados em diferentes arquiteturas, com diferentes objetivos ou mesmo com base em diferentes conceitos. A adoção de uma abordagem para múltiplos modelos pode ajudar na condução do projeto de melhoria de processo da organização. Abordagem no contexto deste trabalho é uma estratégia utilizada para conduzir um projeto de melhoria de processo em uma organização intensiva de software.

Com base em uma pesquisa exploratória e em estudos em melhoria de processos foram identificados quatro tipos de abordagens para integrar elementos de múltiplos modelos. Essas abordagens foram denominadas: Integração de Modelos, Modelo Guia, Mapeamento de Modelos e Framework de Modelos. As mesmas abordagens foram identificadas como iniciativas de melhoria de processo de software e com nomenclaturas diferentes na revisão sistemática da literatura sobre avaliação e medição de melhoria de processo de software (Unterkalmsteiner et al. 2012). As iniciativas são: Frameworks Estabelecidos, Framework Combinados e Frameworks Derivados.

A abordagem de Integração de Modelos equivale a iniciativa de Frameworks Derivados, as abordagens Modelo Guia e Mapeamento de Modelos equivalem a iniciativa de Frameworks Combinados e a abordagem de Framework de Modelos equivale a iniciativa de Frameworks Estabelecidos. Para definir as abordagens três conceitos são utilizados, modelo, framework e perfil de capacidade de processo. Segundo o CMMI (SEI 2010), modelos são coleções de boas práticas que ajudam as organizações a melhorar os seus processos. Framework é uma estrutura básica que organiza os componentes do modelo, bem como as regras e métodos para a criação de um modelo. Um perfil de capacidade de processo é um conjunto de áreas de processos e seus correspondentes níveis de capacidade.

Na abordagem **Integração de Modelos**, um novo modelo é construído a partir de outros modelos e normas. Um perfil de capacidade de processo é selecionado deste modelo resultante e a melhoria de processo é feita com base neste perfil. Um exemplo é o modelo MR-MPS-SW – Modelo de Referência de Melhoria de Processo do Software para Software (SOFTEX 2012a). A base técnica para a construção e aprimoramento desse modelo de melhoria e avaliação de proces-

so de software é composta pelas normas ISO/IEC 12207 (ISO/IEC 2008), ISO/IEC 15504-2 (ISO/IEC 2003) e pelo modelo CMMI-DEV (SEI 2010).

Na abordagem **Modelo Guia**, um perfil de capacidade de processo de um modelo é escolhido para guiar a melhoria de processo na organização; por meio deste modelo garante-se que no momento apropriado outros perfis de outros modelos sejam incluídos no projeto de melhoria de processo da organização. Essa abordagem busca harmonizar os diferentes conceitos dos modelos em um único perfil ao longo do desenvolvimento do projeto de melhoria (Mendes 2010).

Na abordagem **Mapeamento de Modelos**, o mapeamento de modelos é usado para identificar sobreposições de abrangência, a cobertura, as semelhanças e diferenças existentes entre os modelos e padrões adotados pela organização para melhoria de processos. Um perfil de capacidade de processo é criado a partir deste mapeamento e a melhoria de processo é feita com base neste perfil. Mapear modelos não é uma tarefa fácil, pois os modelos e padrões possuem arquiteturas, sintaxes e terminologias diferentes, que devem ser respeitadas (Thiry, Zoucas, and Tristão 2010).

Na abordagem **Framework de Modelos**, uma estrutura padrão é utilizada para organizar e combinar os componentes do modelo. Os modelos criados a partir de um dado framework atendem a finalidades diferentes, porém compartilham a mesma arquitetura e conceitos, já estão preparados para serem utilizados de forma integrada com outros modelos do mesmo framework. Um exemplo é o Framework CMMI (SEI 2010) e seus modelos, o CMMI-DEV (SEI 2010), o CMMI-ACQ (SEI 2007) e o CMMI-SVC (SEI 2009). Estes modelos foram criados a partir do Framework CMMI. Cada um deles com objetivos diferentes: desenvolver produtos e serviços, adquirir produtos e serviços e fornecer serviços. Cada modelo é desenvolvido considerando os outros modelos e já preparado para ser utilizado junto com os outros modelos. Nessa abordagem, um perfil de capacidade de processo pode conter áreas de processo e níveis de capacidade do CMMI-DEV, CMMI-ACQ e do CMMI-SVC e a melhoria de processo é feita com base neste perfil.

Estes quatro tipos de abordagens para integrar elementos de múltiplos modelos têm uma característica comum: se baseiam no paradigma da atual melhoria de processo de software, elas focam na utilização de modelos pré-definidos e por isso, limitam a utilização plena de múltiplos. Com isso faz se necessário utilizar um novo paradigma para se trabalhar com múltiplos modelos em melhoria de processo. Este novo paradigma foi proposto pela metodologia PRO2PI.

Em um artigo com resultados parciais desta dissertação (Banhesse, Salviano, and Jino 2012a), foram propostos critérios para comparar as abordagens para múltiplos modelos. Os modelos de capacidade são constituídos de boas práticas organizadas em suas respectivas arquiteturas, portanto podem ser considerados como repositórios de conhecimento, ou seja, meios utilizados para armazenar o conhecimento. Desta maneira, as abordagens podem ser comparadas em termos das características destas mídias. Armour (2004) define cinco características das quais três delas são mais adequadas para comparação das abordagens: **Frequência de Atualização** – quão rápido o conhecimento armazenado na mídia pode ser atualizado; **Intencionalidade** – quanto do armazenamento e da modificação de conhecimento na mídia pode ser feito de maneira deliberada; e **Habilidade de Auto-modificação** – quanto do conhecimento armazenado na mídia pode modificar a si mesmo.

2.7 Metodologia PRO2PI

PRO2PI – *Process Modeling Profile to drive Process Improvement* – Perfil de Modelagem de Processo para dirigir uma Melhoria de Processos versão 4.0 (Salviano 2006) e (Salviano 2009) – é uma metodologia de melhoria de processo multi-modelo dirigida por perfis de modelagem de processo. Um Perfil de Modelagem de Processo é composto por três tipos de modelos: Modelo de Perfil de Capacidade de Processo, Modelo de Descrição Formal de Processo e Modelo Indicador de Desempenho de Processo (Salviano et al. 2010).

Um Modelo de Perfil de Capacidade de Processo é composto por um processo ou por uma área de processo em um determinado nível de capacidade, como exemplo: processo ENG.1 – *Requirements elicitation* no nível de capacidade 2 – Processo Gerenciado da ISO/IEC 15504-5 (ISO/IEC 2006) ou a área de processo REQM – *Requirements Management* no nível de capacidade 2 – Gerenciado do CMMI-DEV (SEI 2010). Modelo de Descrição Formal de Processo é composto por um ciclo de vida, papéis, atividades e artefatos. O SPEM – *Software & Systems Process Engineering Metamodel specification* é um exemplo. Modelo Indicador de Desempenho de Processo é composto por necessidades de informação, informações sobre o produto, indicador e medidas, como exemplo, o PSM – *Practical Software and System Measurement* e ISO/IEC 15939 *Software Measurement Process* (Salviano 2011).

PRO2PI apoia a melhoria de processos usando elementos de múltiplos modelos de referência e de outras fontes. Estes elementos são selecionados ou definidos e integrados como um perfil de modelagem de processo. A Metodologia PRO2PI é composta por oito componentes metodológicos, dentre eles existe o PRO2PI-MMC, que é o metamodelo para perfis de capacidade de processo da metodologia PRO2PI. O PRO2PI-MMC é composto por três componentes: o Geraes – é um metamodelo que representa os princípios fundamentais dos modelos de capacidade de processo; o metamodelo de arquitetura de modelos de capacidade e o metamodelo de regras de transformação de arquitetura de modelos de capacidade.

"Os problemas significativos que enfrentamos não podem ser resolvidos no mesmo nível de pensamento em que estávamos quando os criamos."

(Albert Einstein)

3 PRO2PI-MMC – Metamodelo de Perfis de Capacidade de Processo

Este capítulo apresenta as três fases estratégicas utilizadas para desenvolver o trabalho. A primeira fase tratou da identificação e sistematização de um conjunto unificado de conceitos básicos para representar as boas práticas dos modelos de capacidade de processo (Salviano and Figueiredo 2008). A segunda fase enfatiza o desenvolvimento e a utilização de um protótipo de software para representar arquiteturas, modelos e regras adicionais de transformação de arquiteturas, baseado nos conceitos unificados. O escopo deste trabalho de mestrado cobre esta segunda fase. A terceira fase é um trabalho futuro, que vai tratar da evolução do protótipo de software para representar os relacionamentos com graus de similaridade entre as partes de dois modelos.

Além disso, este capítulo apresenta o Metamodelo de Perfis de Capacidade de Processo denominado PRO2PI-MMC – *Process Modeling Profile to drive Process Improvement MetaModel for Process Capability Profile*. O PRO2PI-MMC é formado por três componentes, o Metamodelo de Conceitos Unificados (Geraes), um Metamodelo de Arquitetura de Modelos de Capacidade de Processo e um Metamodelo de Regras Adicionais de Transformação de Arquitetura de Modelos de Capacidade de Processo.

3.1 Estratégia de Desenvolvimento do Trabalho

Os modelos de capacidade de processo em estudo, tais como: o CMMI-DEV versão 1.3 (SEI 2010), o MR-MPS-SW versão agosto/2012 (SOFTEX 2012a) e o ASPICE versão 2.5 (SPICE 2010) são repositórios de boas práticas; onde as práticas (conteúdo) estão organizadas sob uma determinada arquitetura (forma). A arquitetura de um modelo tem um papel fundamental, pois permite inferir sobre o modelo. Devido a esta característica inerente aos modelos e seguindo a orientação de Atkinson e Kühne (2005) e de Henderson (2007) de não misturar o meta-

modelo linguístico (forma) com o metamodelo ontológico (conteúdo), três fases estratégicas foram estabelecidas para desenvolver o Metamodelo de Perfis de Capacidade de Processo.

A primeira fase tratou da identificação e sistematização de um conjunto unificado de conceitos básicos para representar as boas práticas dos modelos de capacidade de processo. Como, por exemplo: o modelo CMMI-DEV versão 1.3 (SEI 2010) e um modelo de Perfil de Capacidade de Processo, exemplo: o nível de maturidade três do CMMI-DEV. Como parte da pesquisa do PRO2PI, Salviano e Figueiredo (2008) propuseram um conjunto de conceitos unificados que trata do desafio dessa primeira fase. Essa fase é a base fundamental para o desenvolvimento das outras duas fases.

A segunda fase enfatiza o desenvolvimento e a utilização de um protótipo de software para representar arquiteturas, modelos e regras adicionais de transformação de arquiteturas, baseado nos conceitos unificados. Com esta aplicação de software, um modelo pode ser desacoplado de sua arquitetura original e então ser automaticamente transformado de uma arquitetura para outra. O desafio desta segunda fase é transformar automaticamente modelos com diferentes arquiteturas em uma única arquitetura. O objetivo da segunda fase é desenvolver um metamodelo linguístico capaz de metamodelar as arquiteturas dos modelos de capacidade.

A terceira fase é um trabalho futuro, que vai tratar da evolução do protótipo de software para representar os relacionamentos com graus de similaridade entre as partes de dois modelos. Um relacionamento pode ser alguma coisa tal como uma parte A de um modelo X é totalmente, largamente, parcialmente ou não similar com a parte B de um modelo Y. Com estes relacionamentos, o software pode gerar uniões, intersecções e diferenças de partes de modelos distintos. O objetivo da terceira fase é desenvolver um metamodelo ontológico capaz de metamodelar o conteúdo dos modelos de capacidade.

3.2 Metamodelo de Perfis de Capacidade de Processo

Em 2008, Salviano e Figueiredo (2008) propuseram um conjunto de conceitos básicos unificados denominado Geraes, que são princípios fundamentais para perfis e modelos de capacidade de processo. Na segunda fase do projeto, o Geraes que já contemplava os conceitos do modelo CMMI-DEV versão 1.2 (SEI 2006) e do modelo ISO/IEC 15504-5 versão 2005 (ISO/IEC 2005) foi revisado para garantir que os seus conceitos básicos fossem suficientes para contemplar também os conceitos do modelo CMMI-DEV versão 1.3 (SEI 2010), ISO/IEC 15504-5 versão 2006 (ISO/IEC 2006), ASPICE (SPICE 2010) e do modelo MR-MPS-SW (SOFTEX 2012a). A revisão considerou os conceitos representados pelas arquiteturas destes modelos, por isso, o processo de revisão foi dividido em três etapas. Na primeira etapa, as arquiteturas dos modelos foram representadas graficamente. Na segunda etapa, as arquiteturas foram analisadas em relação ao Geraes e, na terceira etapa, o Geraes foi alterado para contemplar os novos conceitos. A revisão foi realizada com a ferramenta ArgoUML versão 0.30 para Windows 7.

Para criar um Metamodelo de Perfis de Capacidade de Processo capaz de representar as arquiteturas dos modelos de capacidade de processo e capaz de representar as regras adicionais de arquiteturas necessárias para realizar as transformações de modelos para melhoria de processo com múltiplos modelos, neste trabalho foram criados mais dois componentes complementares ao Geraes. Dessa forma, o Metamodelo de Perfis de Capacidade de Processo denominado PRO2PIMMC – *Process Modeling Profile to drive Process Improvement MetaModel for Process Capability Profile* é formado por três componentes, pelo Metamodelo de Conceitos Unificados (Geraes), por um Metamodelo de Arquitetura de Modelos de Capacidade de Processo e por um Metamodelo de Regras Adicionais de Transformação de Arquitetura de Modelos de Capacidade de Processo.

A Figura 3.1 apresenta um diagrama de pacotes em UML com o relacionamento entre os três componentes que em conjunto formam o PRO2PI-MMC.

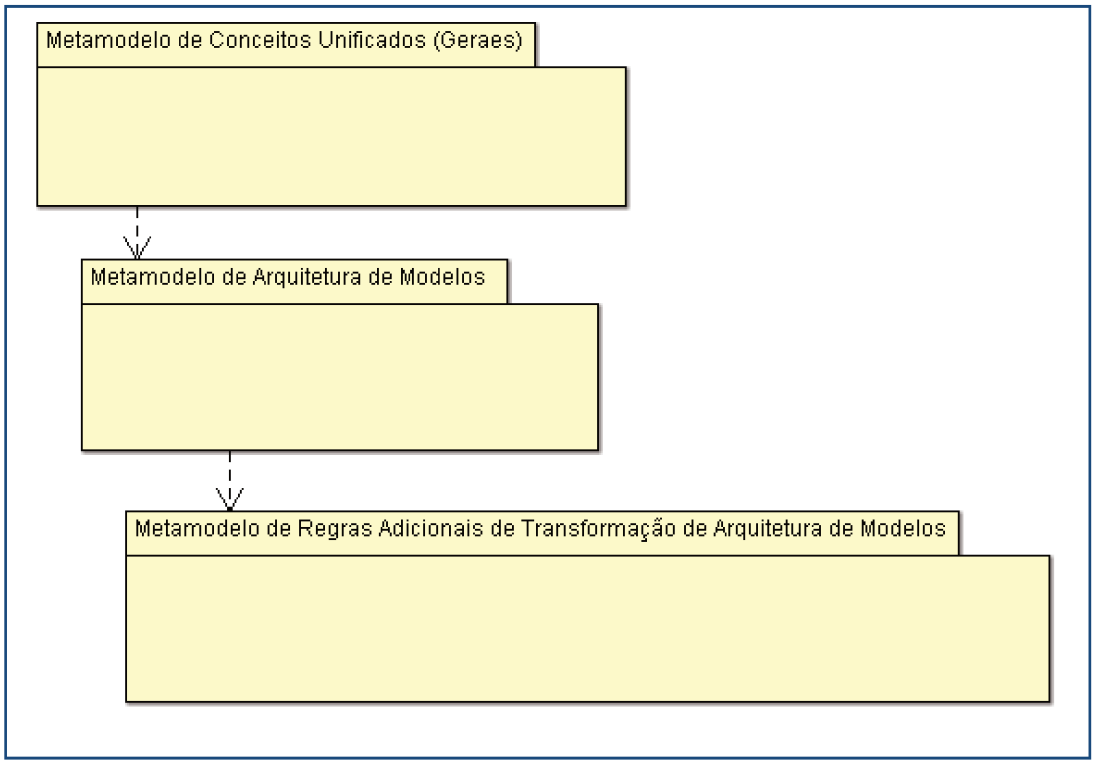


Figura 3.1 – PRO2PI-MMC – Metamodelo de Perfis de Capacidade de Processo.

Cada componente é um metamodelo com uma função específica, o Metamodelo de Conceitos Unificados (Geraes) é responsável por representar os conceitos utilizados para criar perfis e modelos de capacidade de processo. O Metamodelo de Arquitetura de Modelos de Capacidade de Processo é responsável por representar os elementos arquitetônicos dos modelos e os seus respectivos relacionamentos utilizados para criar as arquiteturas dos modelos de capacidade. E o Metamodelo de Regras Adicionais de Transformação de Arquitetura de Modelos de Capacidade de Processo é responsável por representar as regras utilizadas para definir as transformações de arquiteturas de modelos, ou seja, são regras que definem como um modelo em sua arquitetura de origem pode ser escrito na arquitetura de outro modelo de destino. Os componentes são detalhados a seguir.

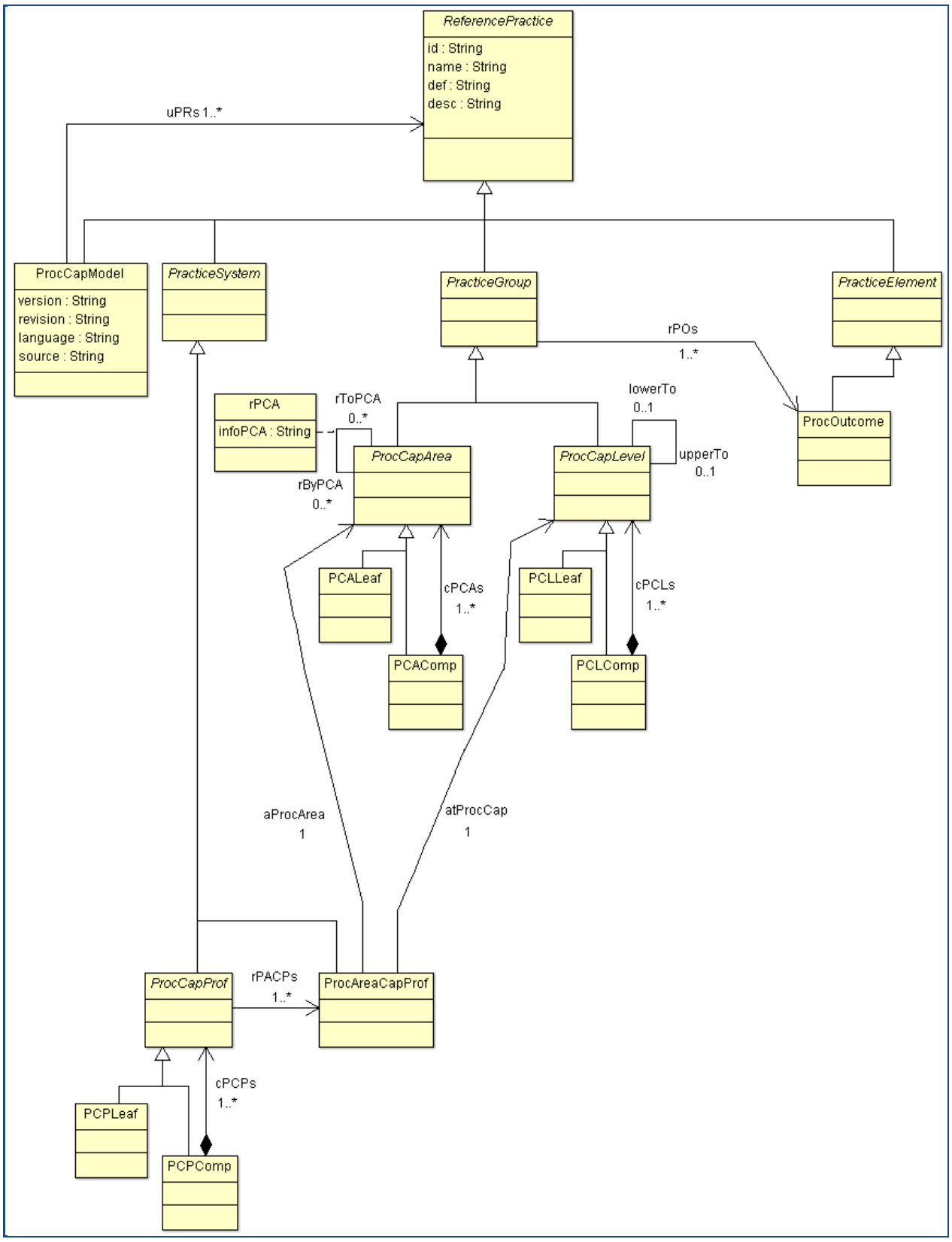


Figura 3.2 – Geraes – Metamodelo de Conceitos Unificados.

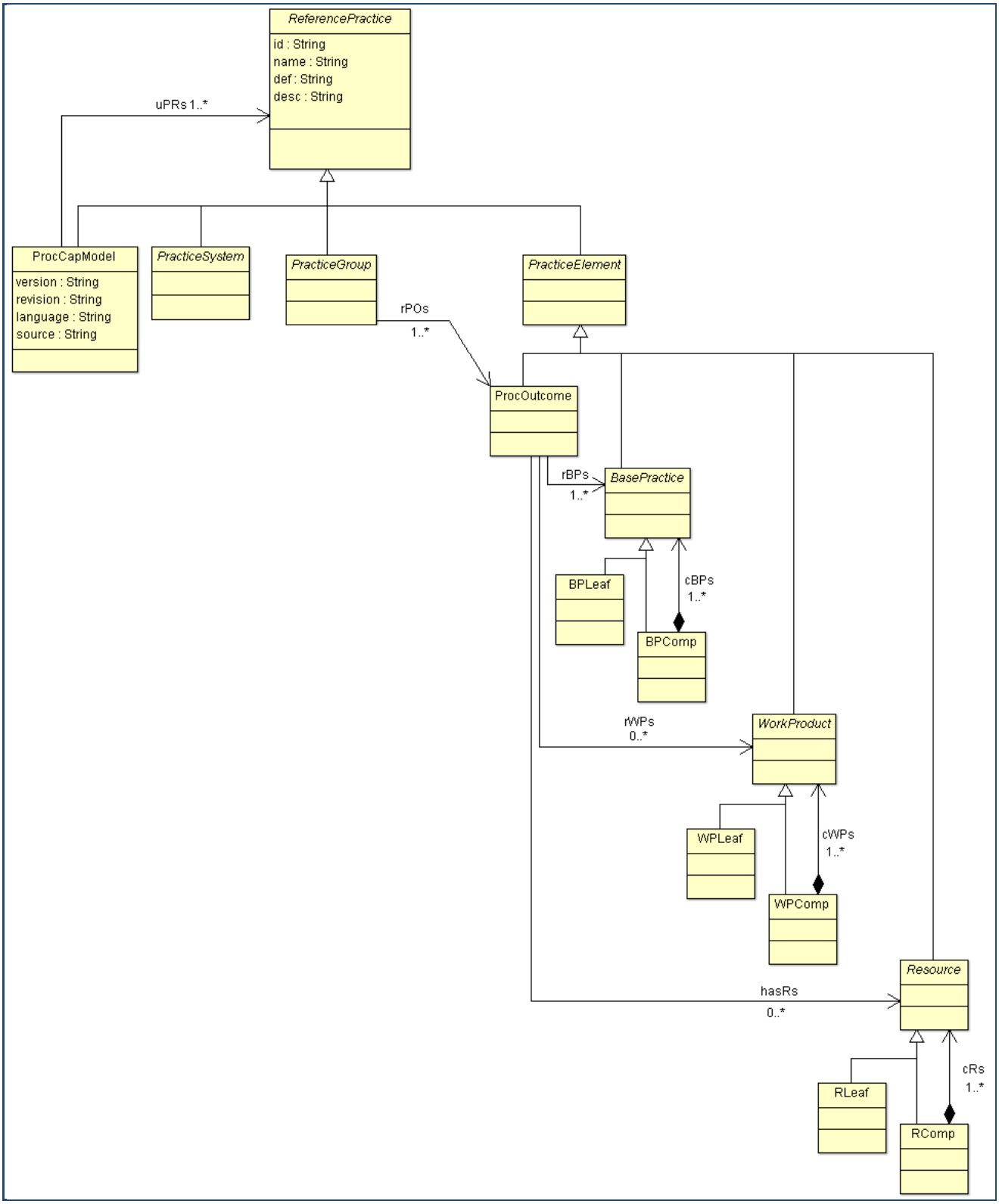


Figura 3.2 (Continuação)

O Geraes apresentado na Figura 3.2 é um diagrama de classes em UML. Em relação a sua versão original de 2008 (Salviano and Figueiredo 2008:Figure-1, Page-174) foi incluído a classe *rPCA – related Process Capability Area* para possibilitar a representação das áreas de capacidade de processo que estão relacionadas. O Geraes é formado por uma superclasse denominada Prática de Referência, que por sua vez é especializada em quatro subclasses denominadas Modelo de Capacidade de Processo, Sistema de Prática, Grupo de Prática e Elemento de Prática. A classe Sistema de Prática é especializada em duas subclasses denominadas Perfil de Capacidade de Processo e Perfil de Capacidade de Área de Processo. A classe Grupo de Prática é especializada em duas subclasses denominadas Área de Capacidade de Processo e Nível de Capacidade de Processo. E a classe Elemento de Prática é especializada em quatro subclasses denominadas Resultado de Processo, Prática Base, Produto de Trabalho e Recurso.

As classes Perfil de Capacidade de Processo, Área de Capacidade de Processo, Nível de Capacidade de Processo, Prática Base, Produto de Trabalho e Recurso são modeladas com duas ou mais subclasses cada, usando o *Design Pattern Composite* (Gamma et al. 1994). Este padrão de projeto trata da necessidade de compor objetos dentro de estruturas de árvore para representar hierarquias parte-todo. O *Composite* permite que os objetos individuais e as composições de objetos sejam tratados uniformemente. Cada uma destas classes é modelada como uma classe abstrata com duas subclasses concretas cada, a primeira classe modela a relação de composição (com o sufixo *Comp*) e a segunda classe modela o elemento folha (com o sufixo *Leaf*). A composição e o elemento folha usam como nome a agregação de iniciais do nome da sua classe principal (PCP, PCA, PCL, BP, WP e R) acompanhado do sufixo (*Comp*) ou (*Leaf*).

O Geraes possui quatro princípios fundamentais, ou seja, quatro conceitos básicos baseados em Capacidade de Processo. Capacidade de Processo é a capacidade que um processo possui em satisfazer a qualidade de produto especificada, a qualidade de serviço, e os objetivos de desempenho de processo (SEI 2010). Os conceitos básicos são: Modelo de Capacidade de Processo, Área de Capacidade de Processo, Nível de Capacidade de Processo e Perfil de Capacidade de Processo.

O Modelo de Capacidade de Processo representa todas as boas práticas de modelos de referência organizadas sob o conceito de Capacidade de Processo, por exemplo, o modelo CMMI-DEV, a ISO/IEC 15504-5, o ASPICE e o MR-MPS-SW. Área de Capacidade de Processo é um conjunto de boas práticas relacionadas especificamente a “o que fazer”, como exemplo, REQM –

Gerência de Requisitos do modelo CMMI-DEV. Nível de Capacidade de Processo é um conjunto de boas práticas relacionadas genericamente a "quão bem fazer", como exemplo, o nível de capacidade 2 – Gerenciado do modelo CMMI-DEV. Perfil de Capacidade de Processo é um modelo de um processo sob o aspecto de capacidade de processo, é um conjunto de processos ou áreas de processo em determinados níveis de capacidade.

O nível de maturidade 2 – Gerenciado da representação estagiada do modelo CMMI-DEV é um exemplo de um Perfil de Capacidade de Processo formado por sete áreas de processo CM – Gerência de Configuração, MA – Medição e Análises, PMC – Controle e Monitoramento de Projeto, PP – Planejamento de Projeto, PPQA – Garantia de Qualidade do Processo e Produto REQM – Gerência de Requisitos e SAM – Gerência de Acordo com o Fornecedor no nível de capacidade 2 – Gerenciado.

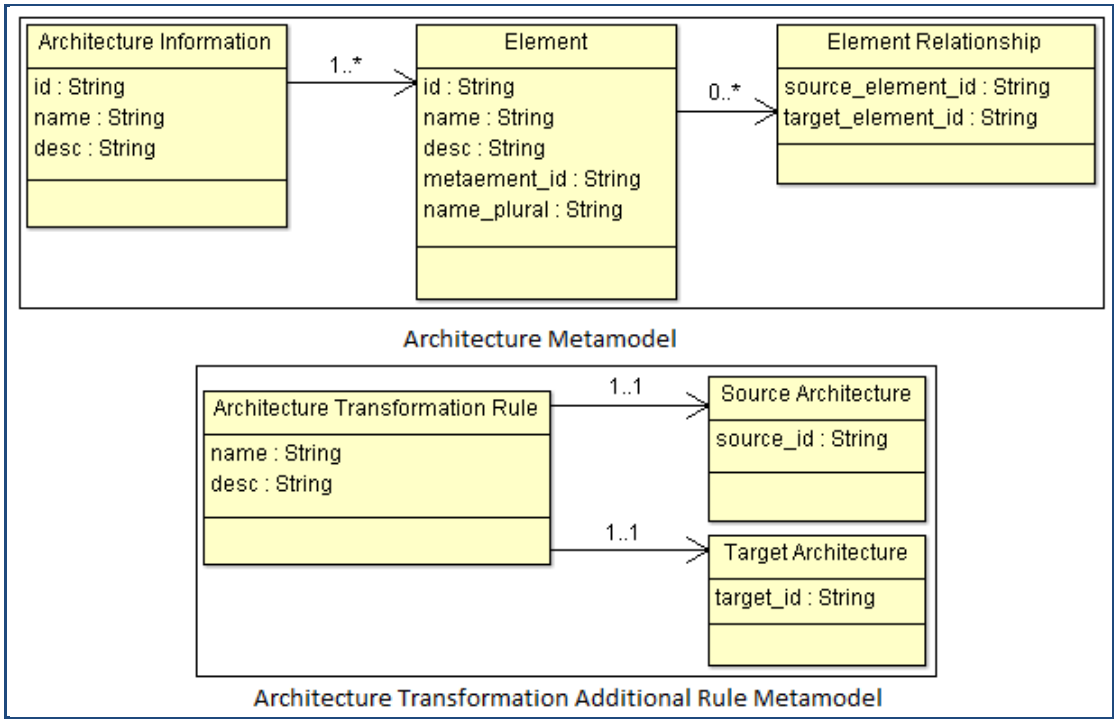


Figura 3.3 – Metamodelo de Arquitetura e Metamodelo de Regra Adicional de Transformação de Arquitetura.

A Figura 3.3 apresenta o detalhamento dos dois últimos componentes do PRO2PI-MMC, o Metamodelo de Arquitetura de Modelos de Capacidade de Processo é um diagrama de classes em UML formado por três classes, a classe Informação da Arquitetura que identifica a arquitetura do modelo, como por exemplo, a arquitetura CMMI. A classe Elemento que identifica o elemento que pertence à arquitetura do modelo e que tem um relacionamento direto com o Geraes, como

por exemplo, a Área de Processo. E a classe de Relacionamento de Elementos que estabelece a relação entre dois elementos que formam uma arquitetura de modelo, por exemplo, uma Área de Processo é formada por uma ou mais Metas Específicas.

O Metamodelo de Regras Adicionais de Transformação de Modelos de Capacidade de Processo é um diagrama de classe em UML formado por três classes, a classe Regra de transformação de Arquitetura que identifica a regra de transformação adicional de modelos, a classe Origem e a classe de Destino da Arquitetura que identificam o relacionamento entre o elemento de origem e o elemento de destino da arquitetura de dois modelos envolvidos em um processo de transformação. O elemento Área de Processo da arquitetura de origem do modelo CMMI-DEV, por exemplo, é transformado no elemento Processo da arquitetura de destino do modelo ASPICE. As regras adicionais de transformação são criadas por especialistas em modelos e são adicionais, pois complementam ou alteram as regras gerais de transformação de arquiteturas, que são geradas automaticamente a partir do processo de cadastro da arquitetura dos modelos. Isso ocorre, pois os elementos arquitetônicos dos modelos compartilham os conceitos unificados representados pelo Geraes.

3.3 Exemplo de Modelagem com o PRO2PI-MMC

Esta seção apresenta um exemplo simplificado da modelagem da área de processo REQM – *Requirements Management* – Gerência de Requisitos do modelo CMMI-DEV versão 1.3 na arquitetura CMMI e sua transformação para a arquitetura MPS.BR. A modelagem deste exemplo será realizada em três etapas. A primeira etapa trata da modelagem da arquitetura CMMI e da arquitetura MPS.BR com o PRO2PI-MMC, a segunda etapa trata da modelagem de parte dos elementos da área de processo REQM na arquitetura CMMI. E por último, a terceira etapa trata da transformação dos elementos da área de processo REQM na arquitetura CMMI para a arquitetura MPS.BR.

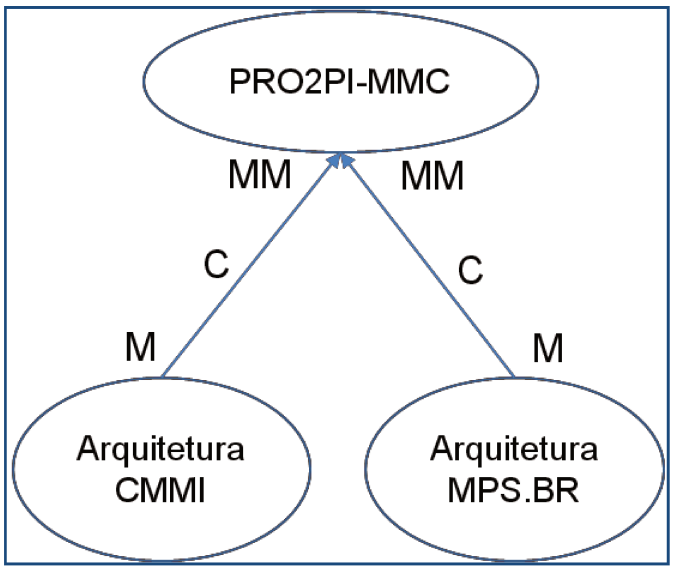


Figura 3.4 – Relação de conformidade entre as arquiteturas CMMI e MPS.BR e o PRO2PI-MMC.

Segundo Bézivin (2006), um modelo (M) está em conformidade (C) com um metamodelo (MM), se e somente se, cada elemento do modelo tem seu metaelemento definido no metamodelo. A Figura 3.4 apresenta que os modelos das arquiteturas CMMI e MPS.BR estão em conformidade com o metamodelo PRO2PI-MMC. Desta forma, para atender a primeira etapa do exemplo, os elementos dos modelos que compõem a arquitetura CMMI e a arquitetura MPS.BR serão modelados com os metaelementos que compõem o metamodelo PRO2PI-MMC.

A arquitetura CMMI de uma área de processo é composta pelos elementos *Process Area*, *Specific Goal*, *Specific Practice*, *Specific Subpractice* e *Typical Work Product*. Onde uma *Process Area* possui um ou mais *Specific Goals*, um *Specific Goal* possui uma ou mais *Specific Practices*, e uma *Specific Practice* possui uma ou mais *Specific Subpractices* e *Typical Work Products*. Os metaelementos e seus respectivos atributos do PRO2PI-MMC utilizados na modelagem da arquitetura são *ProcCapArea*, *ProcOutcome*, *BasePractice* e *WorkProduct* do Metamodelo de Conceitos Unificados (Geraes), *Architecture Information*, *Element* e *Element Relationship* do Metamodelo de Arquitetura de Modelos de Capacidade de Processo.

A Tabela 3.1 apresenta a modelagem detalhada da arquitetura CMMI via PRO2PI-MMC. No exemplo, para aumentar a capacidade de modelagem do Geraes foram utilizados metaelementos dos tipos *Comp* e *Leaf*.

Tabela 3.1 – Modelagem da arquitetura CMMI via PRO2PI-MMC.

Metaelemento do PRO2PI-MMC	**Atributo do Metaelemento PRO2PI-MMC**	**Elemento da Arquitetura CMMI**
Architecture Information	*id*	CMMI
	name	*Capability Maturity Model Integration*
	desc	CMMI *are collections of best practices* (...)
Element	*id*	PA
	name	*Process Area*
	desc	*An element of CMMI architecture.*
	metaelement_id	*ProcCapArea (Comp)*
	name_plural	*Process Areas*
	relationship belongs to	CMMI
Element	*id*	SG
	name	*Specific Goal*
	desc	*An element of CMMI architecture.*
	metaelement_id	*ProcCapArea (Leaf)*
	name_plural	*Specific Goals*
	relationship belongs to	CMMI

Tabela 3.1 (Continuação)

Metaelemento do PRO2PI-MMC	**Atributo do Metaelemento PRO2PI-MMC**	**Elemento da Arquitetura CMMI**
Element	*id*	SP
	name	*Specific Practice*
	desc	*An element of CMMI architecture.*
	metaelement_id	*ProcOutcome*
	name_plural	*Specific Practices*
	relationship belongs to	CMMI
Element	*id*	SSP
	name	*Specific Subpractice*
	desc	*An element of CMMI architecture.*
	metaelement_id	*BasePractice (Leaf)*
	name_plural	*Specific Subpractices*
	relationship belongs to	CMMI
Element	*id*	TWP
	name	*Typical Work Product*
	desc	*An element of CMMI architecture.*
	metaelement_id	*WorkProduct (Leaf)*
	name_plural	*Typical Work Products*
	relationship belongs to	CMMI
Element Relationship	*source_element_id*	SG
	target_element_id	PA
Element Relationship	*source_element_id*	SP
	target_element_id	SG
Element Relationship	*source_element_id*	SSP
	target_element_id	SP
Element Relationship	*source_element_id*	TWP
	target_element_id	SP

"(...)" – Significa que o texto ou parte do modelo foi omitido para melhorar a legibilidade.

A arquitetura MPS.BR de um processo é composta pelos elementos *Process* e *Process Outcome*. Onde um *Process* possui um ou mais *Process Outcomes*. Os metaelementos e seus respectivos atributos do PRO2PI-MMC utilizados na modelagem da arquitetura são *ProcCapArea* e *ProcOutcome* do Metamodelo de Conceitos Unificados (Geraes), *Architecture Information*, *Element* e *Element Relationship* do Metamodelo de Arquitetura de Modelos de Capacidade de Processo. A Tabela 3.2 apresenta a modelagem detalhada da arquitetura MPS.BR via PRO2PI-MMC.

Tabela 3.2 – Modelagem da arquitetura MPS.BR via PRO2PI-MMC.

Metaelemento do PRO2PI-MMC	**Atributo do Metaelemento PRO2PI-MMC**	**Elemento da Arquitetura MPS.BR**
Architecture Information	*id*	MPS.BR
	name	Melhoria de Processo do Software Brasileiro
	desc	O MPS.BR é um programa mobilizador, de longo prazo, criado em dezembro de 2003, coordenado pela Associação para Promoção da Excelência (...)
Element	*id*	P
	name	*Process*
	desc	*An element of MPS.BR architecture.*
	metaelement_id	*ProcCapArea (Leaf)*
	name_plural	*Processes*
	relationship belongs to	MPS.BR
Element	*id*	PO
	name	*Process Outcome*
	desc	*An element of MPS.BR architecture.*
	metaelement_id	*ProcOutcome*
	name_plural	*Process Outcomes*
	relationship belongs to	MPS.BR
Element Relationship	*source_element_id*	PO
	target_element_id	P

"(...)" – Significa que o texto ou parte do modelo foi omitido para melhorar a legibilidade.

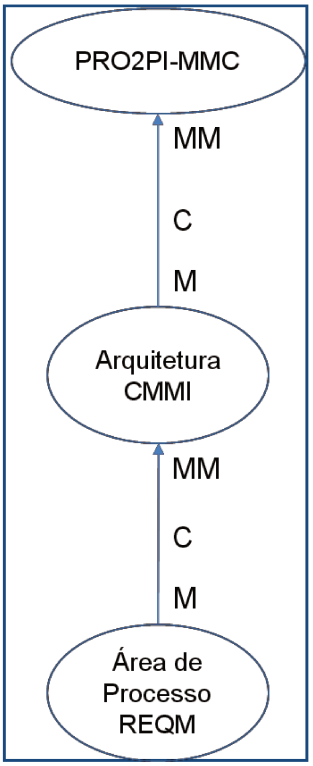


Figura 3.5 – Relação de conformidade entre a área de processo REQM e a arquitetura CMMI.

A Figura 3.5 apresenta que a área de processo REQM está em conformidade com o metamodelo da arquitetura CMMI. Desta forma, para atender a segunda etapa do exemplo, os elementos do modelo que compõem a área de processo REQM serão modelados com os metaelementos que compõem o metamodelo da arquitetura CMMI. A área de processo REQM é composta pelos elementos *Process Area*, *Specific Goal*, *Specific Practice*, *Specific Subpractice* e *Typical Work Product*. Onde uma *Process Area* possui um ou mais *Specific Goals*, um *Specific Goal* possui uma ou mais *Specific Practices*, e uma *Specific Practice* possui uma ou mais *Specific Subpractices* e *Typical Work Products*.

Os metaelementos e seus respectivos atributos da arquitetura CMMI utilizados na modelagem da arquitetura são *Process Area*, *Specific Goal*, *Specific Practice*, *Specific Subpractice* e *Typical Work Product*, que por sua vez foram modelados com os metaelementos e seus respectivos atributos do PRO2PI-MMC, que são *ProcCapArea*, *ProcOutcome*, *BasePractice* e *WorkProduct* do Metamodelo de Conceitos Unificados (Geraes), *Architecture Information*, *Element* e *Element Relationship* do Metamodelo de Arquitetura de Modelos de Capacidade de Processo.

A Tabela 3.3 apresenta a modelagem detalhada da área de processo REQM via arquitetura CMMI.

Tabela 3.3 – Modelagem da área de processo REQM via arquitetura CMMI.

Metaelemento da Arquitetura CMMI / Metaelemento do PRO2PI-MMC	**Atributo do Metaelemento do PRO2PI-MMC**	**Elemento da Área de Processo REQM**
Process Area / ProcCapArea(Comp)	*id*	REQM
	name	*Requirements Management*
	def	*The purpose of Requirements Management (REQM) is to manage requirements of the* (...)
	desc	*Requirements management processes manage all requirements received or generated by the project* (...)
Specific Goal / ProcCapArea(Leaf)	*id*	SG 1
	name	*Manage Requirements*
	def	*Requirements are managed and inconsistencies* (...)
	desc	*The project maintains a current and approved set of requirements* (...)
Specific Practice / ProcOutcome	*id*	SP 1.1
	name	*Understand Requirements*
	def	*Develop an understanding with the requirements providers on the meaning of the requirements*.
	desc	*As the project matures and requirements are derived, all activities or disciplines will* (...)
Specific Subpractice / BasePractice(Leaf)	*id*	1
	name	*Establish criteria for distinguishing appropriate requirements providers*.
	def	--
	desc	--
Typical Work Product / WorkProduct(Leaf)	*id*	1
	name	*Lists of criteria for distinguishing appropriate requirements providers*.
	def	--
	desc	--

"(...)" – Significa que o texto ou parte do modelo foi omitido para melhorar a legibilidade.

A Tabela 3.4 apresenta o relacionamento entre as arquiteturas CMMI e MPS.BR via PRO2PI-MMC.

Tabela 3.4 – Relacionamento entre as arquiteturas CMMI e MPS.BR via PRO2PI-MMC.

Tipo de Regra de Transformação	**Metaelemento do PRO2PI-MMC**	**Elemento da Arquitetura CMMI**	**Metaelemento do PRO2PI-MMC**	**Elemento da Arquitetura MPS.BR**
Adicional	*ProcCapArea(Comp)*	*Process Area*	*ProcCapArea(Leaf)*	*Process*
Adicional	*ProcCapArea(Leaf)*	*Specific Goal*	--	*Empty*
Geral	*ProcOutcome*	*Specific Practice*	*ProcOutcome*	*Process Outcome*
Adicional	*BasePractice(Leaf)*	*Specific Subpractice*	--	*Empty*
Adicional	*WorkProduct(Leaf)*	*Typical Work Product*	--	*Empty*

De acordo com os relacionamentos apresentados, uma regra de transformação do tipo geral foi criada, pois os elementos arquitetônicos *Specific Practice* da arquitetura CMMI e *Process Outcome* da arquitetura MPS.BR compartilham o mesmo conceito unificado representado pelo Geraes, no caso o metaelemento *ProcOutcome*.

Desta forma, para atender a terceira etapa do exemplo, quatro regras adicionais devem ser criadas para completar e alterar as regras gerais de transformação de arquiteturas. Os metaelementos e seus respectivos atributos do PRO2PI-MMC utilizados na modelagem das regras adicionais são *Architecture Transformation Rule*, *Source Architecture* e *Target Architecture* do Metamodelo de Regras Adicionais de Transformação de Modelos de Capacidade de Processo.

A Tabela 3.5 apresenta a modelagem detalhada das regras adicionais de transformação entre as arquiteturas CMMI e MPS.BR via PRO2PI-MMC.

Tabela 3.5 – Modelagem das regras adicionais entre as arquiteturas CMMI e MPS.BR via PRO2PI-MMC.

Metaelemento do PRO2PI-MMC	**Atributo do Metaelemento PRO2PI-MMC**	**Elemento da Arquitetura CMMI e MPS.BR**
Architecture Transformation Rule	*name*	CMMI-2-MPS.BR
	desc	*Additional transformation rule from CMMI Architecture Element to MPS.BR Architecture Element.*
Source Architecture	*source_id*	PA – *Process Area*
Target Architecture	*target_id*	P – *Process*
Architecture Transformation Rule	*name*	CMMI-2-MPS.BR
	desc	*Additional transformation rule from CMMI Architecture Element to MPS.BR Architecture Element.*
Source Architecture	*source_id*	SG – *Specific Goal*
Target Architecture	*target_id*	*Empty*
Architecture Transformation Rule	*name*	CMMI-2-MPS.BR
	desc	*Additional transformation rule from CMMI Architecture Element to MPS.BR Architecture Element.*
Source Architecture	*source_id*	SSP – *Specific Subpractice*
Target Architecture	*target_id*	*Empty*
Architecture Transformation Rule	*name*	CMMI-2-MPS.BR
	desc	*Additional transformation rule from CMMI Architecture Element to MPS.BR Architecture Element.*
Source Architecture	*source_id*	TWP – *Typical Work Product*
Target Architecture	*target_id*	*Empty*

Após realizar as três etapas do exemplo, o resultado da transformação de parte da área de processo REQM – *Requirements Management* – Gerência de Requisitos do modelo CMMI-DEV versão 1.3 na arquitetura CMMI para a arquitetura MPS.BR é apresentado na Tabela 3.6.

Tabela 3.6 – Transformação de parte da área de processo REQM do CMMI-DEV versão 1.3 para a arquitetura MPS.BR.

Arquitetura MPS.BR	**Conteúdo da Área de Processo REQM**
Process	*id*: REQM
	name: *Requirements Management*
	def: *The purpose of Requirements Management (REQM) is to manage requirements of the* (...)
	desc: *Requirements management processes manage all requirements received or generated by the project* (...)
Process Outcome	*id*: SP 1.1
	name: *Understand Requirements*
	def: *Develop an understanding with the requirements providers on* (...)
	desc: *As the project matures and requirements are derived, all activities* (...)

"(...)" – Significa que o texto ou parte do modelo foi omitido para melhorar a legibilidade.

"Uma experiência nunca é um fracasso, pois sempre vem demonstrar algo."
(Thomas Edison)

4 Implementação do PRO2PI-MMCA: uma Prova de Conceito

Uma prova de conceito é um modelo prático que é utilizado para testar ou para validar um modelo teórico ou um conceito estabelecido por uma pesquisa ou por um artigo técnico. Atualmente a prova de conceito tem sido amplamente utilizada como uma ferramenta para testar e validar teorias e conceitos. Um protótipo de software pode ser implementado como prova de conceito; para realizar um estudo de viabilidade (Femmer et al. 2011), testar uma nova abordagem baseada em computação paralela (Morana and Mikkilineni 2011), para mostrar a aplicação de conceitos chave relacionados à computação em nuvens (Sarathy, Narayan, and Mikkilineni 2010) e para validar uma abordagem que utiliza um metamodelo para geração automática de interfaces (Cruz 2010).

Na segunda fase do projeto foi utilizada uma abordagem semelhante a dos trabalhos citados acima, ou seja, um protótipo de software denominado PRO2PI-MMCA – *Process Modeling Profile to drive Process Improvement MetaModel for Process Capability Profile Application* foi implementado com o intuito de validar o modelo teórico, que é o Metamodelo de Perfis de Capacidade de Processo denominado PRO2PI-MMC – *Process Modeling Profile to drive Process Improvement MetaModel for Process Capability Profile*. As próximas seções apresentam uma visão geral da implementação do PRO2PI-MMCA.

4.1 Aplicação prática do PRO2PI-MMCA

Além de ser uma prova de conceito para validar o metamodelo de perfis de capacidade de processo. A ferramenta PRO2PI-MMCA pode ser utilizada na prática para gerar perfis de capacidade de processo a partir de múltiplos modelos. Estes perfis podem ser utilizados para definir e melhorar os processos de desenvolvimento de software de uma organização. Outro uso da ferramenta refere-se ao aproveitamento de conhecimento existente numa organização. Uma organização que implementou o MR-MPS-SW e não conhece o CMMI-DEV, caso tenha que implementar

uma área de processo do CMMI-DEV, pode usar a ferramenta para transformar a área de processo desejada em um processo do MR-MPS-SW e utilizar os conhecimentos para entender as práticas da área de processo na linguagem do modelo já conhecido.

4.2 Funcionalidades do PRO2PI-MMCA

No paradigma da Engenharia Dirigida por Modelos (MDE) o desenvolvimento de software é baseado na construção de modelos e metamodelos em ambientes computacionais e em transformações automáticas entre modelos para a geração de código executável. Porém estes ambientes ainda estão em evolução para uma utilização mais intensa pela indústria de software (Cruz 2010). No trabalho sobre o Estado da Prática em Engenharia Dirigida por Modelos (Wittle, Hutchinson, and Rouncefield 2013), o estado da arte das técnicas e ferramentas de modelagem ainda contribui pouco no suporte as atividades de desenvolvimento de software. Não foi encontrado um ambiente de desenvolvimento que suporte completamente de forma amigável todo o paradigma da MDE. Os ambientes de desenvolvimento atuais envolvem duas grandes atividades: a modelagem e a codificação. Como o foco do projeto de pesquisa é desenvolver um Metamodelo de Perfis de Capacidade de Processo e não um metamodelo para desenvolvimento de software, optou-se por desenvolver o software PRO2PI-MMCA no paradigma orientado a objetos seguindo os conceitos da MDE.

O software foi implementado por meio da ferramenta de desenvolvimento rápido CakePHP[5] versão 1.3 com a linguagem de programação PHP – *Hypertext Preprocessor* – Pré-processador de Hipertexto versão 5.3.8 e com o sistema gerenciador de banco de dados relacional MySQL versão 5.5. O banco de dados do PRO2PI-MMCA foi exercitado por meio de consultas em SQL – *Structured Query Language* – Linguagem de Consulta Estruturada com o phpMyAdmin versão 3.5 e com o MySQL Workbench, ambos em Windows 7. O sistema operacional utilizado na implementação foi o Windows versão 7 e o servidor Web foi o Apache versão 2.2.21. A aplicação foi testada nos sistemas operacionais Windows versões XP e 7, e no Linux Ubuntu versão 12.10. Os navegadores utilizados para testes foram o Internet Explorer, versão 8 para Windows XP e 9 para Windows 7, e o Mozilla FireFox versão 20 (para Windows e Linux).

[5] *Cake Software Foundation* – http://cakephp.org/

As transformações de modelos são realizadas por meio de funcionalidades específicas implementadas em CakePHP e que fazem o uso da DSL – *Domain Specific Language* – Linguagem Específica de Domínio SQL – *Structured Query Language* – Linguagem de Consulta Estruturada. Uma DSL é uma linguagem projetada para ser útil para um conjunto limitado de tarefas em um domínio específico de problema, em contraste a uma linguagem de propósito geral, que é útil para muitas tarefas genéricas nos mais diversos tipos de domínios de aplicação. A SQL é um exemplo de DSL, útil para tarefas relacionadas a problemas específicos de um banco de dados relacional (Jouault and Bézivin 2006).

A definição das funcionalidades do software PRO2PI-MMCA levou em consideração as relações do padrão de conformidade de Favre (2005) e os testes realizados na fase exploratória do projeto de pesquisa. As funcionalidades especificadas para o software estão relacionadas e descritas a seguir agrupadas em cinco grupos funcionais, cada grupo funcional contém uma série de operações. Nesta descrição o termo gerenciar significa um conjunto de quatro operações: cadastrar, consultar, editar e remover. As operações referentes à verificação e validação são realizadas automaticamente pelo software. As funcionalidades e operações são:

1. Representar arquiteturas de modelos de capacidade de processo. São as arquiteturas específicas de modelos de capacidade de processo descritas na linguagem do metamodelo de capacidade de processo.
    a. Gerenciar as informações de uma arquitetura de modelos.
    b. Gerenciar os elementos de uma arquitetura de modelos.
    c. Gerenciar os relacionamentos entre os elementos de uma arquitetura de modelos.
2. Representar modelos de capacidade de processo. São os modelos de capacidade de processo específicos descritos na linguagem de sua própria arquitetura.
    a. Gerenciar as informações de um modelo.
    b. Gerenciar os componentes de um modelo.
    c. Gerenciar os relacionamentos entre os componentes de um modelo.
3. Representar perfis de capacidade de processo. São os relacionamentos entre as áreas de processos/processos e os níveis de capacidades dos modelos de capacidade de processo.
    a. Gerenciar um perfil de capacidade de processo de um modelo específico.

    b. Gerenciar um perfil de capacidade de processo de múltiplos modelos.

4. Estabelecer regras adicionais de transformação entre os elementos de arquiteturas dos modelos de capacidade de processo. São as correlações entre os elementos de arquiteturas de modelos, que possibilitam que uma arquitetura de um modelo de origem possa ser transformada em uma arquitetura de um modelo de destino. As regras são adicionais, pois o metamodelo de capacidade de processo já define automaticamente algumas correlações entre os elementos de arquiteturas de modelos.
    a. Gerenciar uma regra de transformação entre os elementos de arquiteturas de modelos.
5. Realizar transformações de partes de modelos de capacidade de processo e transformações de perfis de capacidade de processo. São as operações realizadas com base no metamodelo de capacidade de processo e com base nas regras adicionais de transformações entre os elementos de arquiteturas dos modelos.
    a. Apresentar uma área de processo/processo ou um nível de capacidade de um modelo de origem na arquitetura de um modelo de destino.
    b. Apresentar um perfil de capacidade de processo de um modelo específico ou de múltiplos modelos na arquitetura de um modelo de destino.

4.3 Arquitetura do PRO2PI-MMCA

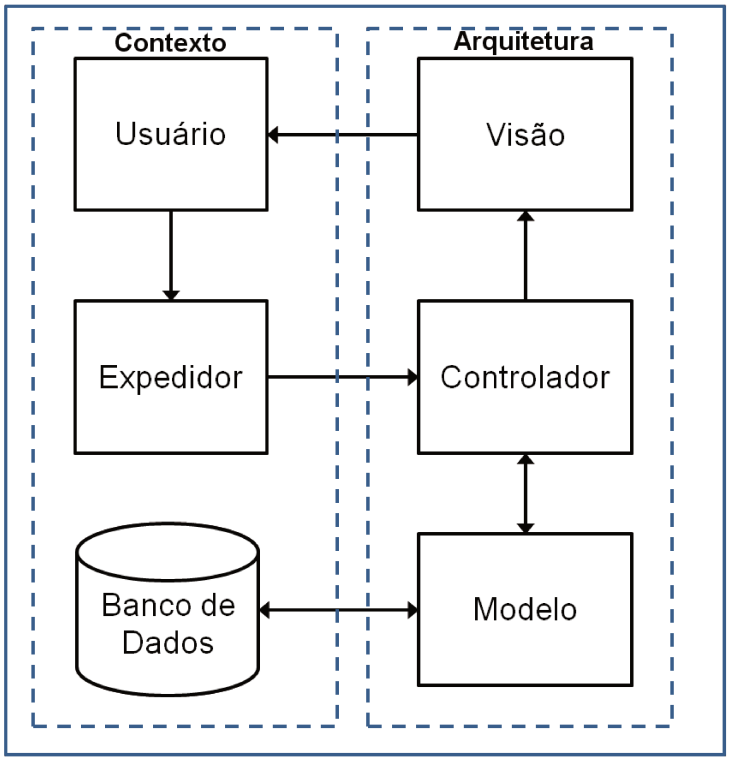


Figura 4.1 – Contexto e arquitetura do PRO2PI-MMCA.

A Figura 4.1 apresenta os componentes envolvidos no contexto do PRO2PI-MMCA e sua arquitetura. O usuário, o expedidor e o banco de dados são os componentes que fazem parte do contexto do PRO2PI-MMCA, enquanto os elementos modelo, visão e controlador compõem a arquitetura da ferramenta. O software foi implementado por meio da ferramenta de desenvolvimento CakePHP, que utiliza o padrão arquitetônico MVC – Modelo-Visão-Controle, que separa as operações em áreas específicas, o modelo contém as classes responsáveis por toda a interação com o banco de dados, a visão contém as classes que tratam de todas as informações de saídas e suas apresentações ao usuário e o controlador contém as classes responsáveis por todos os comandos para entrada e fluxo do programa. O principal objetivo do MVC é certificar que cada função da aplicação seja escrita somente uma vez aperfeiçoando o código pela diminuição de redundâncias (Golding 2008).

As classes do modelo lidam com as consultas que gerenciam as informações do banco de dados referente às arquiteturas dos modelos de capacidade de processo, as regras adicionais de

transformação de arquiteturas dos modelos de capacidade de processo, ao conteúdo dos modelos de capacidade de processo e ao Geraes.

As classes de visão lidam com as interfaces de entradas e com as de saídas disponíveis ao usuário, por exemplo, para tratar a entrada de dados das arquiteturas e dos conteúdos dos modelos de capacidade e para saída, a geração de relatórios contendo um perfil de capacidade de processo de um único modelo ou a transformação de um perfil de capacidade de processo de múltiplos modelos para uma única arquitetura.

As classes de controle gerenciam as funções necessárias para representar e transformar os modelos de capacidade de processo, incluindo todas as operações de cadastro, consulta, edição, remoção, verificação e validação. As classes do MVC implementam o PRO2PI-MMC. A Figura 4.2 apresenta o diagrama de sequência em UML do PRO2PI-MMCA entre os componentes do contexto e os elementos da arquitetura da Figura 4.1.

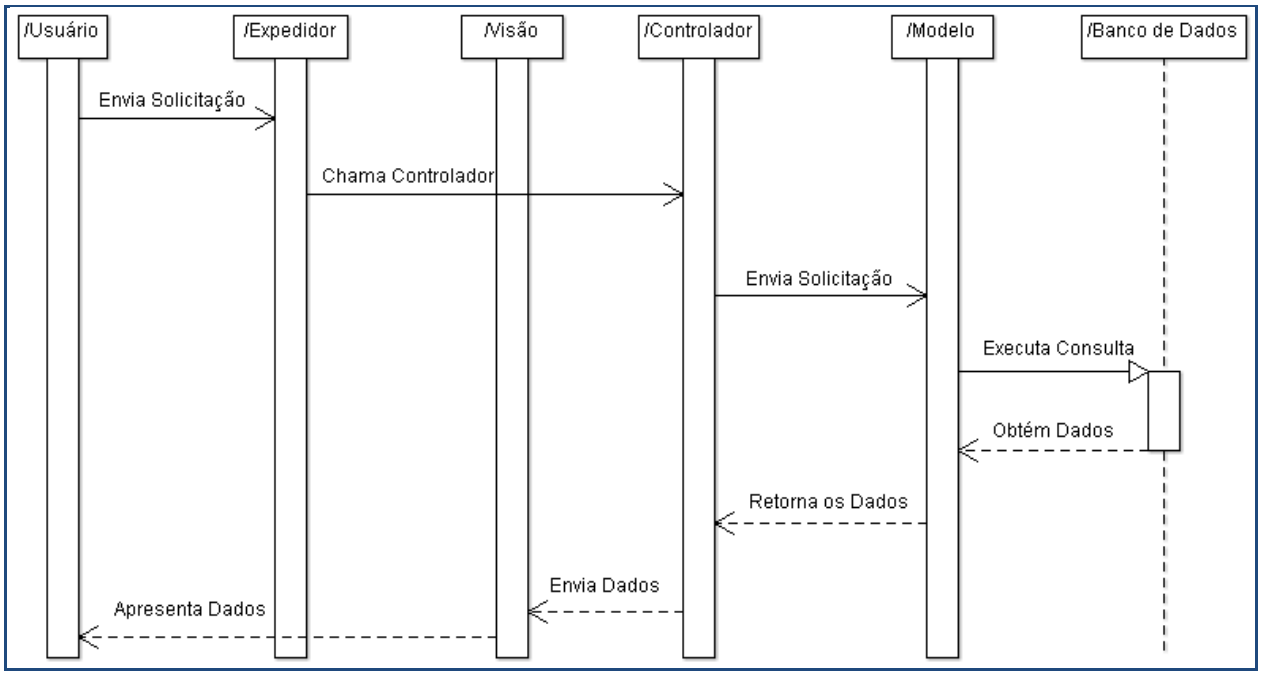


Figura 4.2 – Diagrama de sequência do PRO2PI-MMCA.

As etapas são:

1. O usuário envia uma solicitação de página para a aplicação. Digitando, por exemplo, uma URL – *Uniform Resource Locator*, clicando em um link ou em um botão. Por convenção uma típica URL para aplicações implementadas em CakePHP, segue essa estrutura: [http://{Domínio}/{Aplicação}/{Controlador}/{Ação}/{Parâmetro1, ...}].

2. O expedidor recebe a URL digitada e analisa sua estrutura para determinar qual controlador chamar para executar uma ação.
3. A ação do controlador recebe os parâmetros da URL e envia uma solicitação ao modelo.
4. O modelo determina como interagir com o banco de dados usando as solicitações submetidas pelo controlador. Executar, por exemplo, uma consulta ao banco de dados para obter a arquitetura CMMI.
5. Uma vez que o modelo obteve todos os dados do banco de dados, ele os retorna ao controlador.
6. O controlador processa esses dados e envia para a visão.
7. A visão trata os dados recebidos, ela formata os dados para atender um padrão pré-estabelecido e envia para o navegador do usuário.

4.4 Modelo de dados do PRO2PI-MMCA

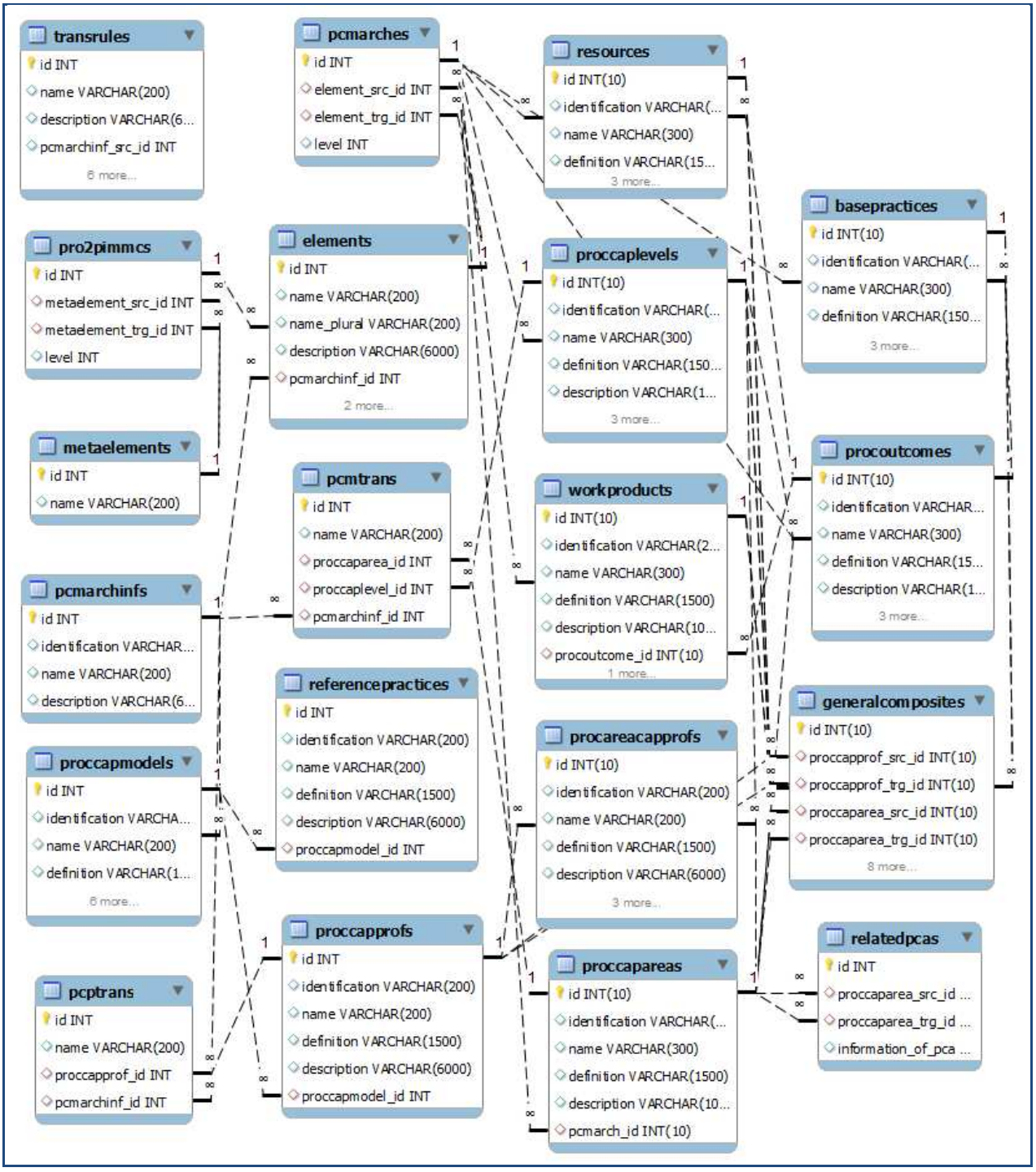


Figura 4.3 – Modelo de Dados do PRO2PI-MMCA.

O modelo de dados do PRO2PI-MMCA apresentado na Figura 4.3 foi gerado pelo MySQL Workbench 5.2 e foi implementado no sistema gerenciador de banco de dados relacional MySQL versão 5.5, o modelo é formado por vinte tabelas que representam o PRO2PI-MMC. O Quadro 4.1 apresenta o dicionário de dados do banco de dados do PRO2PI-MMCA.

Quadro 4.1 – Dicionário de dados do banco de dados do PRO2PI-MMCA.

Nome da Tabela	Descrição
metaelements	Armazena o nome dos metaelementos do Metamodelo de Conceitos Unificados (Geraes).
pro2pimmcs	Armazena o relacionamento entre os metaelementos do Metamodelo de Conceitos Unificados (Geraes).
proccapprofs	Armazena os dados sobre os perfis de capacidade de processo. Os dados são: identificação, nome, definição, descrição e modelo de capacidade de processo.
procareacapprofs	Armazena os dados sobre os perfis de capacidade de área de processo. Os dados são: identificação, nome, definição, descrição, perfil de capacidade de processo, área de capacidade de processo e nível de capacidade de processo.
proccapareas	Armazena os dados sobre um metaelemento do tipo área de capacidade de processo. Os dados são: identificação, nome, definição, descrição e arquitetura do modelo de capacidade de processo.
proccaplevels	Armazena os dados sobre um metaelemento do tipo nível de capacidade de processo. Os dados são: identificação, nome, definição, descrição, nível de capacidade superior, nível de capacidade inferior e arquitetura do modelo de capacidade de processo.
procoutcomes	Armazena os dados sobre um metaelemento do tipo resultado de processo. Os dados são: identificação, nome, definição, descrição, área de capacidade de processo, nível de capacidade de processo e arquitetura do modelo de capacidade de processo.

Quadro 4.1 (Continuação)

Nome da Tabela	Descrição
basepractices	Armazena os dados sobre um metaelemento do tipo prática base. Os dados são: identificação, nome, definição, descrição, resultado de processo e arquitetura do modelo de capacidade de processo.
workproducts	Armazena os dados sobre um metaelemento do tipo produto de trabalho. Os dados são: identificação, nome, definição, descrição, resultado de processo e arquitetura do modelo de capacidade de processo.
resources	Armazena os dados sobre um metaelemento do tipo recurso. Os dados são: identificação, nome, definição, descrição, resultado de processo e arquitetura do modelo de capacidade de processo
generalcomposites	Armazena o relacionamento entre os metaelementos perfil de capacidade de processo, área de capacidade de processo, nível de capacidade de processo, prática base, produto de trabalho e recurso. São os metaelementos do Metamodelo de Conceitos Unificados (Geraes), que utilizam o *Design Pattern Composite* e podem ser do tipo Composto (*Comp*) ou Folha (*Leaf*).
proccapmodels	Armazena os dados sobre um modelo de capacidade de processo. Os dados são: identificação, nome, definição, descrição, versão, revisão, idioma, modelo de capacidade de origem e informação sobre a arquitetura do modelo de capacidade de processo.
referencepractices	Armazena os dados sobre as práticas de referências. Os dados são: identificação, nome, definição, descrição e modelo de capacidade de processo.
relatedpcas	Armazena os relacionamentos e as informações entre as áreas de capacidade de processo.
pcmarchinfs	Armazena os dados sobre a arquitetura do modelo de capacidade de processo. Os dados são: identificação, nome e descrição.

Quadro 4.1 (Continuação)

Nome da Tabela	Descrição
elements	Armazena os dados sobre os elementos das arquiteturas dos modelos de capacidade de processo. Os dados são: nome, plural do nome, descrição, informação sobre a arquitetura do modelo de capacidade de processo, relacionamento entre os metaelementos do Metamodelo de Conceitos Unificados (Geraes).
pcmarches	Armazena os relacionamentos entre os elementos das arquiteturas dos modelos de capacidade de processo.
transrules	Armazena as regras adicionais de transformação das arquiteturas dos modelos de capacidade de processo. Os principais dados são: nome e descrição.
pcmtrans	Armazena as transformações de áreas de capacidade de processo e de níveis de capacidade de processo. Os dados são: nome, área de capacidade de processo, nível de capacidade de processo e informação sobre a arquitetura do modelo de capacidade de processo.
pcptrans	Armazena as transformações de perfis de capacidade de processo. Os dados são: nome, perfil de capacidade de processo e informação sobre a arquitetura do modelo de capacidade de processo.

O Quadro 4.2 apresenta as tabelas do modelo de dados do PRO2PI-MMCA, a finalidade das tabelas e qual componente do PRO2PI-MMC elas representam.

Quadro 4.2 – Finalidade das tabelas do PRO2PI-MMCA.

Nome da Tabela	Finalidade	Componente do PRO2PI-MMC
metaelements, pro2pimmcs	Representar a arquitetura do PRO2PI-MMC.	Metamodelo de Conceitos Unificados (Geraes)
proccapprofs, procareacapprofs	Representar os perfis de capacidade de processo de um modelo e de múltiplos modelos.	
proccapareas, proccaplevels, procoutcomes, basepractices, workproducts, resources, generalcomposites, proccapmodels, referencepractices, relatedpcas	Representar os componentes dos modelos de capacidade de processo.	
pcmarchinfs, elements, pcmarches	Representar as arquiteturas dos modelos de capacidade de processo.	Metamodelo de Arquitetura de Modelos de Capacidade de Processo
transrules, pcmtrans, pcptrans	Representar as regras adicionais de transformação de arquiteturas de modelos, as transformações das partes de modelos, as transformações de perfis de capacidade de processo de um modelo e de múltiplos modelos de capacidade de processo.	Metamodelo de Regras Adicionais de Transformação de Arquitetura de Modelos de Capacidade de Processo

4.5 Regras de transformação das arquiteturas dos modelos

Esta seção apresenta as regras de transformação das arquiteturas dos modelos utilizadas para transformar o conteúdo de um modelo de origem em sua arquitetura original em uma arquitetura de outro modelo de destino. O software PRO2PI-MMCA utiliza regras gerais e regras adicionais para realizar as operações de transformação de partes de modelos ou de perfis de modelos de capacidade de processo. As regras gerais são regras criadas automaticamente pelo PRO2PI-MMCA a partir do cadastro dos elementos das arquiteturas dos modelos. As regras adicionais são criadas por especialistas em modelos com o intuito de obter uma transformação precisa de modelos. As regras de transformação adicionais tem prioridade sobre as regras gerais e por isso substituem as regras gerais.

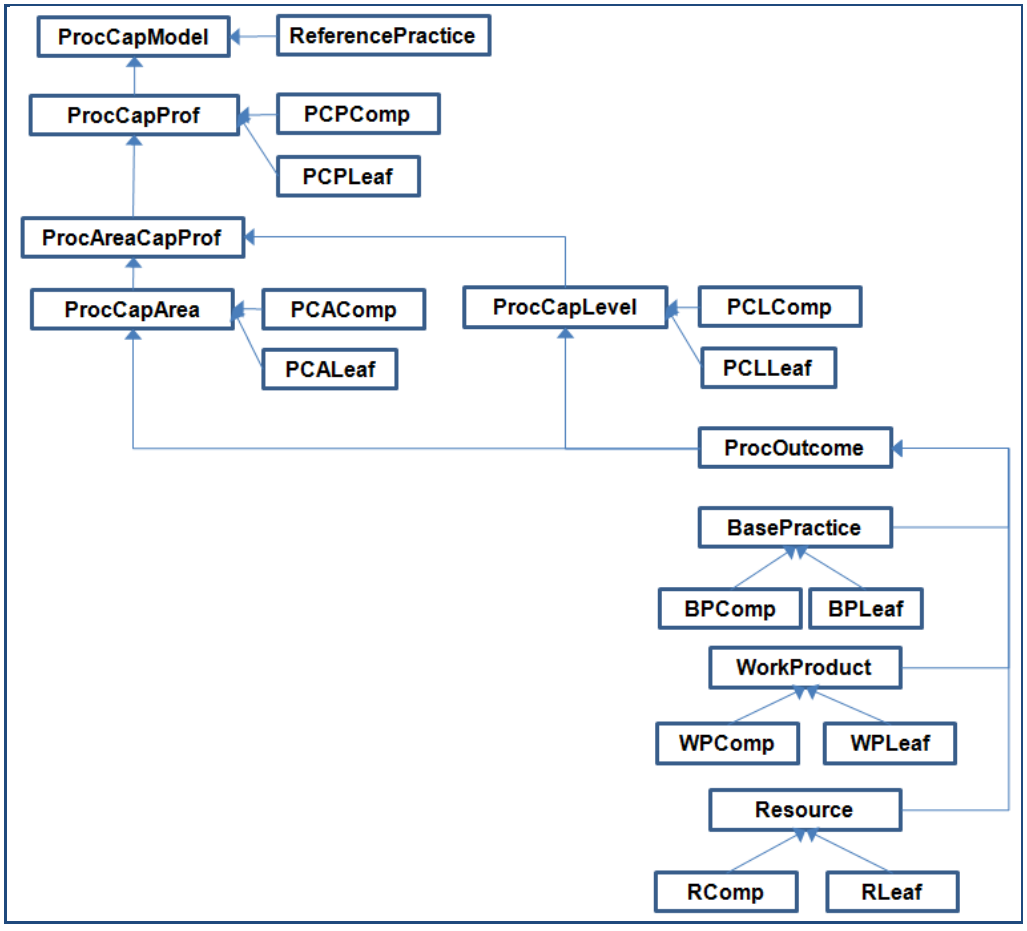


Figura 4.4 – Arquitetura do Geraes.

A Figura 4.4 apresenta a arquitetura do Geraes. Ao cadastrar um elemento da arquitetura de um modelo de capacidade é estabelecido um relacionamento direto com um metaemento da arquitetura do Geraes. O elemento Área de Processo do CMMI, por exemplo, é representado pelo metaelemento *Process Capability Area* (ProcCapArea) do metamodelo Geraes. Desta maneira, ao final do cadastro de toda a arquitetura CMMI, todos os seus elementos terão um relacionamento direto com o Geraes. Estes relacionamentos existem para todas as arquiteturas cadastradas no PRO2PI-MMCA, o que permite que as regras gerais sejam geradas automaticamente pelo PRO2PI-MMCA.

As regras gerais e as adicionais respeitam os relacionamentos definidos para os metaelementos da arquitetura do Geraes. Durante o processo de transformação, um elemento da arquitetura do modelo de origem, representado por um metaelemento da arquitetura do Geraes só pode ser relacionado com um elemento da arquitetura do modelo de destino. O *Design Pattern Composite* (Gamma et al. 1994) foi utilizado para aumentar a capacidade de representação dos metaelementos ProcCapProf (PCP), ProcCapArea (PCA), ProcCapLevel (PCL), BasePractice (BP), WorkProduct (WP) e Resource (R) do Geraes. Estes metaelementos têm classes do tipo composto (Comp) e folha (Leaf).

O software PRO2PI-MMCA verifica se o metaelemento é do tipo composto ou do tipo folha e assegura durante o processo de transformação que os elementos representados por metaelementos do tipo composto não percam informações ao serem transformados para elementos que são representados por metaelementos do tipo folha.

A Figura 4.5 apresenta as transformações (da origem para o destino) possíveis entre os metaelementos do Geraes. Para simplificar o esquema, os metaelementos foram abreviados e os metaelementos que possuem os tipos composto e de folha são representados por uma seta circular que aponta para o próprio metaelemento.

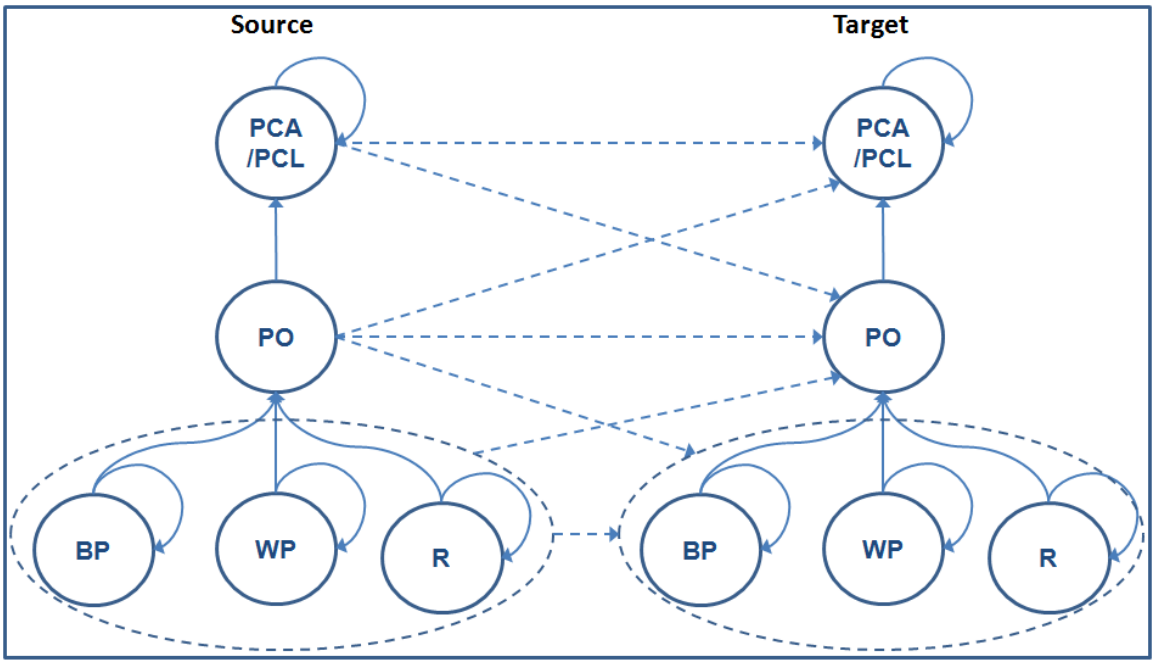


Figura 4.5 – Transformações possíveis entre os metaelementos do Geraes.

Com base nas arquiteturas dos modelos de capacidade de processo apresentados na Seção 2.4 Modelos de Capacidade de Processo e na arquitetura do Geraes apresentada na Figura 4.4, os elementos das arquiteturas foram cadastrados no PRO2PI-MMCA conforme apresentado na Figura 4.6.

Geraes	Type	CMMI
Process Area		
Process Capability Area	Comp	Process Area Category
Process Capability Area	Comp	Process Area
Process Capability Area	Leaf	Specific Goal
Process Outcome	--	Specific Practice
Base Practice	Leaf	Specific Subpractice
Work Product	Leaf	Typical Work Product
Capability Level		
Process Capability Level	Comp	Capability Level
Process Capability Level	Leaf	Generic Goal
Process Outcome	--	Generic Practice
Base Practice	Leaf	Generic Subpractice

Geraes	Type	SPICE
Process		
Process Capability Area	Comp	Process Category
Process Capability Area	Comp	Process Group
Process Capability Area	Leaf	Process
Process Outcome	--	Process Outcome
Base Practice	Leaf	Base Practice
Work Product	Leaf	Work Product
Capability Level		
Process Capability Level	Leaf	Capability Level
Process Outcome	--	Process Attribute
Base Practice	Comp	Process Attribute Outcome
Resource	Leaf	Generic Resource
Work Product	Leaf	Generic Work Product
Base Practice	Leaf	Generic Practice

Geraes	Type	MPS.BR
Process		
Process Capability Area	Leaf	Process
Process Outcome	--	Process Outcome
Capability Level		
Process Capability Level	Leaf	Capability Level
Process Outcome	--	Process Attribute
Base Practice	Leaf	Process Attribute Outcome

Figura 4.6 – Relacionamento entre os metaelementos da arquitetura do Geraes e os elementos das arquiteturas dos modelos de capacidade.

Após o cadastro das arquiteturas é possível consultar as regras gerais e criar as regras adicionais necessárias para realizar as transformações. As próximas seções apresentam as regras para a transformação de modelos. Para todas as regras apresentadas, quando um elemento de origem for vazio, significa que um elemento foi criado no destino. Caso um elemento de origem

existir e passar a ser vazio no destino, significa que o elemento foi excluído durante o processo de transformação. Para criar as regras adicionais, especialistas em modelos foram consultados e os mapeamentos apresentados em trabalhos foram considerados (Enterprise SPICE 2010), (Thiry, Zoucas, and Tristão 2010) e (SOFTEX 2012c).

4.5.1 Do CMMI para SPICE

A Figura 4.7 apresenta as regras gerais de transformação da arquitetura CMMI para a arquitetura SPICE.

TR Type	PCM Architecture Source	PCM Architecture Target
Original General TR	Process Area Category	Process Category
Original General TR	Process Area	Process Group
Original General TR	Specific Goal	Process
Original General TR	Specific Practice	Process Outcome
Original General TR	Specific Subpractice	Base Practice
Original General TR	Typical Work Product	Work Product
Original General TR	Generic Goal	Capability Level
Original General TR	Generic Practice	Process Attribute
Original General TR	Generic Subpractice	Generic Practice

Figura 4.7 – Regras gerais de transformação da arquitetura CMMI para a arquitetura SPICE.

A Figura 4.8 apresenta as regras de transformação da arquitetura CMMI para a arquitetura SPICE. Estas regras representam um conjunto formado por regras gerais e regras adicionais. Ao comparar a Figura 4.7 com a Figura 4.8 é possível visualizar que sete regras gerais foram substituídas por onze regras adicionais.

TR Type	PCM Architecture Source	PCM Architecture Target
General TR	Process Area Category	Process Category
General TR	Typical Work Product	Work Product
Additional TR	Empty	Process Group
Additional TR	Process Area	Process
Additional TR	Specific Goal	Process Outcome
Additional TR	Specific Practice	Base Practice
Additional TR	Capability Level	Capability Level
Additional TR	Generic Goal	Process Attribute
Additional TR	Generic Subpractice	Process Attribute Outcome
Additional TR	Empty	Generic Resource
Additional TR	Empty	Generic Work Product
Additional TR	Generic Practice	Generic Practice
Additional TR	Specific Subpractice	Empty

Figura 4.8 – Regras de transformação da arquitetura CMMI para a arquitetura SPICE.

4.5.2 Do CMMI para MPS.BR

A Figura 4.9 apresenta as regras de transformação da arquitetura CMMI para a arquitetura MPS.BR.

TR Type	PCM Architecture Source	PCM Architecture Target
General TR	Specific Practice	Process Outcome
Additional TR	Process Area	Process
Additional TR	Process Area Category	Empty
Additional TR	Specific Goal	Empty
Additional TR	Specific Subpractice	Empty
Additional TR	Typical Work Product	Empty
Additional TR	Capability Level	Capability Level
Additional TR	Generic Goal	Process Attribute
Additional TR	Generic Practice	Process Attribute Outcome
Additional TR	Generic Subpractice	Empty

Figura 4.9 – Regras de transformação da arquitetura CMMI para a arquitetura MPS.BR.

4.5.3 Do MPS.BR para CMMI

A Figura 4.10 apresenta as regras de transformação da arquitetura MPS.BR para a arquitetura CMMI.

TR Type	PCM Architecture Source	PCM Architecture Target
General TR	Process Outcome	Specific Practice
Additional TR	Process	Process Area
Additional TR	Empty	Specific Goal
Additional TR	Empty	Specific Subpractice
Additional TR	Empty	Typical Work Product
Additional TR	Capability Level	Capability Level
Additional TR	Process Attribute	Generic Goal
Additional TR	Process Attribute Outcome	Generic Practice
Additional TR	Empty	Generic Subpractice
Additional TR	Empty	Process Area Category

Figura 4.10 – Regras de transformação da arquitetura MPS.BR para a arquitetura CMMI.

4.5.4 Do MPS.BR para SPICE

A Figura 4.11 apresenta as regras de transformação da arquitetura MPS.BR para a arquitetura SPICE.

TR Type	PCM Architecture Source	PCM Architecture Target
General TR	Process	Process
General TR	Capability Level	Capability Level
General TR	Process Attribute	Process Attribute
Additional TR	Process Outcome	Base Practice
Additional TR	Empty	Process Category
Additional TR	Empty	Process Group
Additional TR	Empty	Process Outcome
Additional TR	Empty	Work Product
Additional TR	Process Attribute Outcome	Process Attribute Outcome
Additional TR	Empty	Generic Resource
Additional TR	Empty	Generic Work Product
Additional TR	Empty	Generic Practice

Figura 4.11 – Regras de transformação da arquitetura MPS.BR para a arquitetura SPICE.

4.5.5 Do SPICE para CMMI

A Figura 4.12 apresenta as regras de transformação da arquitetura SPICE para a arquitetura CMMI.

TR Type	PCM Architecture Source	PCM Architecture Target
General TR	Process Category	Process Area Category
General TR	Work Product	Typical Work Product
Additional TR	Process	Process Area
Additional TR	Process Outcome	Specific Goal
Additional TR	Base Practice	Specific Practice
Additional TR	Empty	Specific Subpractice
Additional TR	Capability Level	Capability Level
Additional TR	Process Attribute	Generic Goal
Additional TR	Generic Practice	Generic Practice
Additional TR	Process Attribute Outcome	Generic Subpractice
Additional TR	Generic Resource	Empty
Additional TR	Generic Work Product	Empty

Figura 4.12 – Regras de transformação da arquitetura SPICE para a arquitetura CMMI.

4.5.6 Do SPICE para MPS.BR

A Figura 4.13 apresenta as regras de transformação da arquitetura SPICE para a arquitetura MPS.BR.

TR Type	PCM Architecture Source	PCM Architecture Target
General TR	Process	Process
General TR	Capability Level	Capability Level
General TR	Process Attribute	Process Attribute
Additional TR	Base Practice	Process Outcome
Additional TR	Process Category	Empty
Additional TR	Process Group	Empty
Additional TR	Process Outcome	Empty
Additional TR	Work Product	Empty
Additional TR	Process Attribute Outcome	Process Attribute Outcome
Additional TR	Generic Resource	Empty
Additional TR	Generic Work Product	Empty
Additional TR	Generic Practice	Empty

Figura 4.13 – Regras de transformação da arquitetura SPICE para a arquitetura MPS.BR.

4.6 Exemplo de utilização do PRO2PI-MMCA

Esta seção apresenta um exemplo simplificado de utilização do software PRO2PI-MMCA. O exemplo ilustra o cadastro da área de processo REQM – *Requirements Management* – Gerência de Requisitos do modelo CMMI-DEV versão 1.3 na arquitetura CMMI e sua transformação para a arquitetura SPICE. As etapas para o cadastro e consulta seguem as funcionalidades (F) do PRO2PI-MMCA. A apresentação de um exemplo completo e detalhado é inapropriado devido a grande quantidade de passos necessários para cadastrar as informações das arquiteturas e dos conteúdos que compõem os modelos CMMI e SPICE.

A primeira etapa é cadastrar as arquiteturas dos modelos CMMI e SPICE (F 1).

F 1 – Representar arquiteturas de modelos de capacidade de processo.

F 1.a – Cadastrar/consultar as informações da arquitetura de modelos.

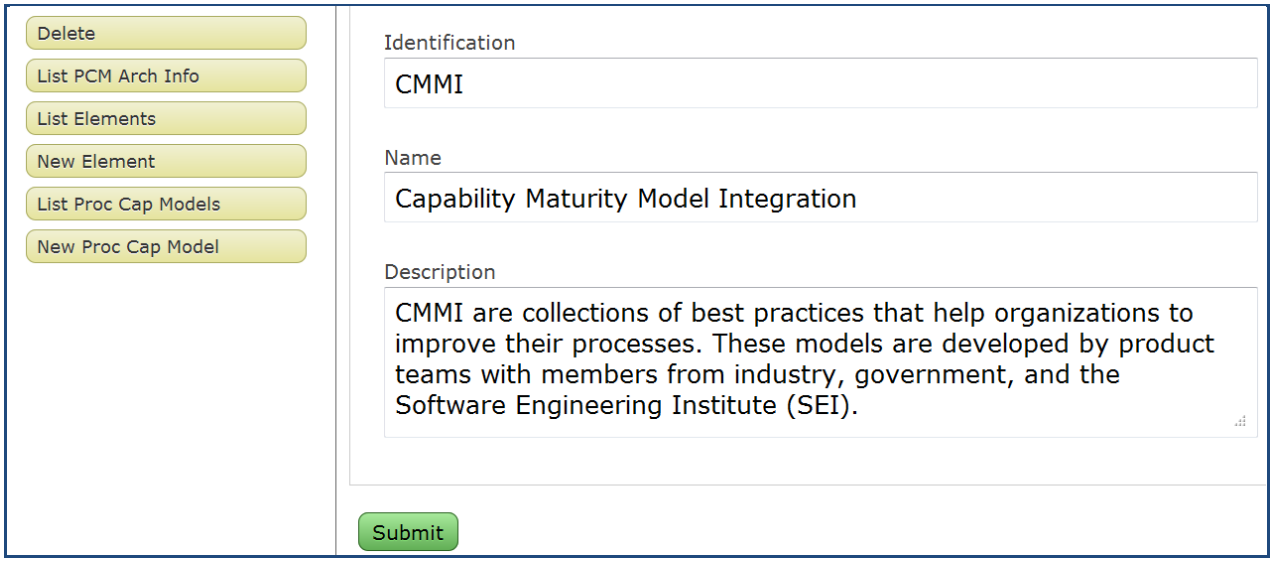


Figura 4.14 – Cadastro da informação da arquitetura CMMI.

A Figura 4.14 apresenta a tela de cadastro da informação da arquitetura CMMI, as informações cadastradas são Identificação, Nome e Descrição.

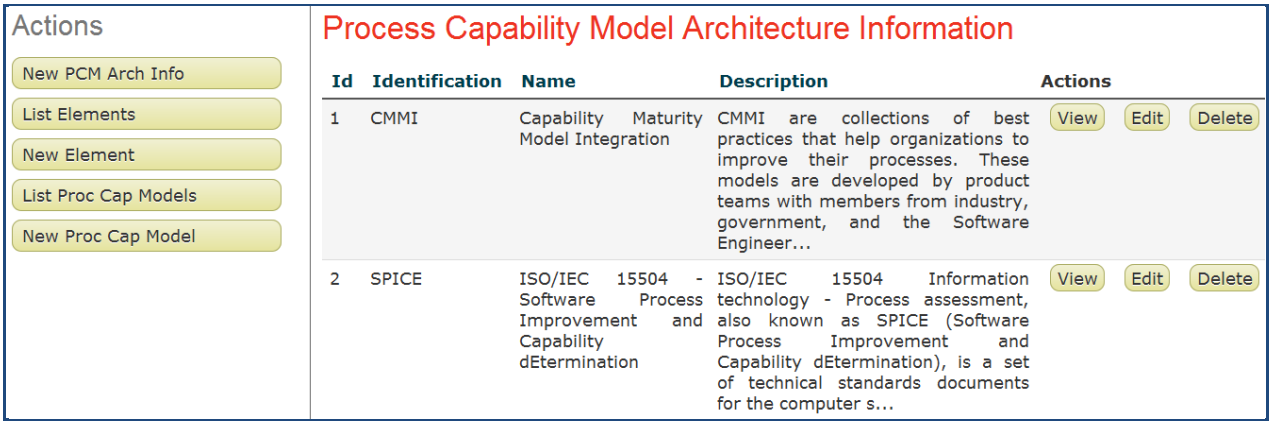


Figura 4.15 – Informação das arquiteturas dos modelos cadastrados no PRO2PI-MMCA.

A Figura 4.15 apresenta a tela de consulta com as informações cadastradas das arquiteturas dos modelos CMMI e SPICE.

F 1.b – Cadastrar/consultar os elementos de uma arquitetura de modelos.

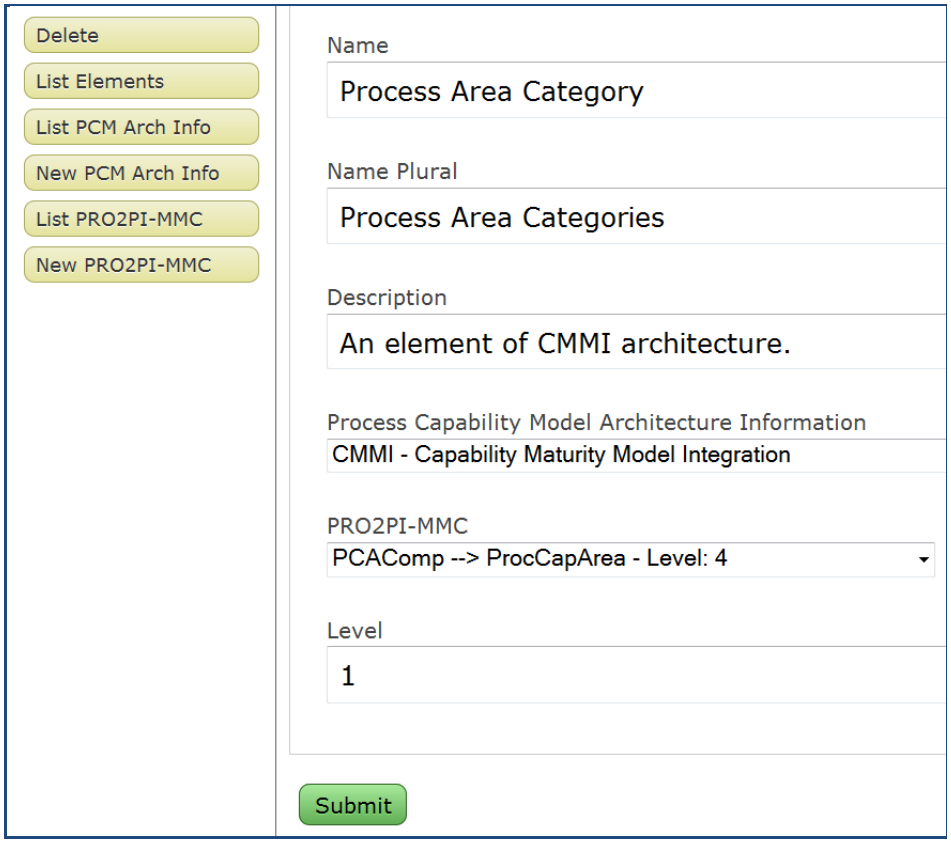


Figura 4.16 – Cadastro do elemento *Process Area Category* da arquitetura CMMI.

A Figura 4.16 apresenta a tela de cadastro do elemento *Process Area Category* da arquitetura CMMI, as informações cadastradas são o Nome, o Plural do Nome, a Descrição, a Informação da Arquitetura do Modelo de Capacidade de Processo, o Metaelemento do Metamodelo PRO2PI-MMC que representa o elemento da arquitetura e o Nível hierárquico que o elemento possui na arquitetura CMMI.

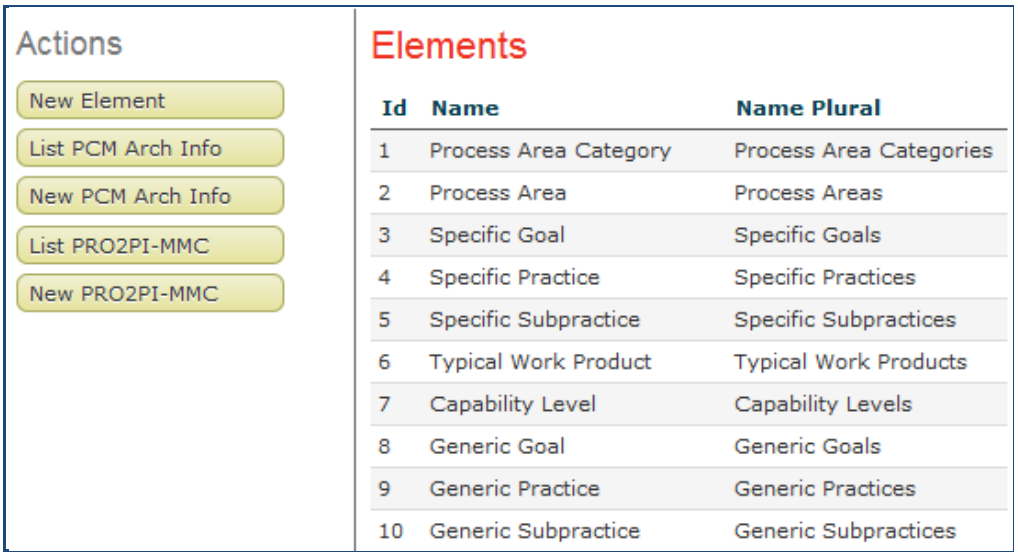


Figura 4.17 – Elementos da arquitetura CMMI.

A Figura 4.17 apresenta a tela de consulta com os elementos cadastrados da arquitetura CMMI.

F 1.c – Cadastrar/consultar os relacionamentos entre os elementos de uma arquitetura de modelos.

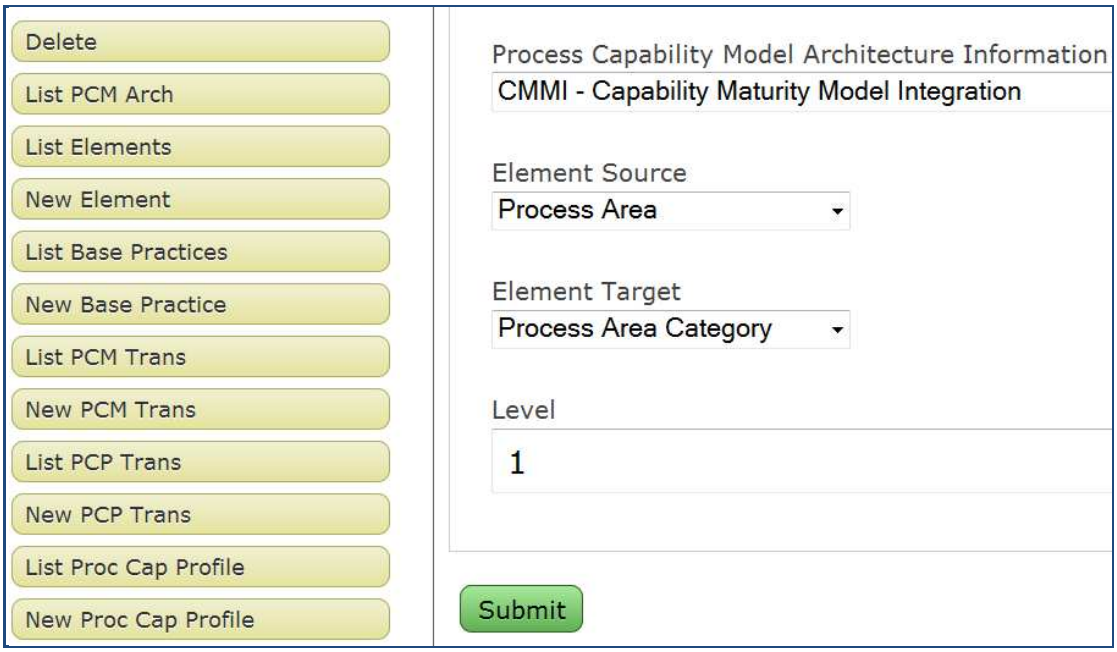


Figura 4.18 – Cadastro de relacionamento entre os elementos *Process Area* e *Process Area Category* da arquitetura CMMI.

A Figura 4.18 apresenta a tela de cadastro entre os elementos *Process Area* e *Process Area Category* da arquitetura CMMI, as informações cadastradas são o Elemento de Origem, o Elemento de Destino e o Nível hierárquico que o par de elementos possui na arquitetura CMMI.

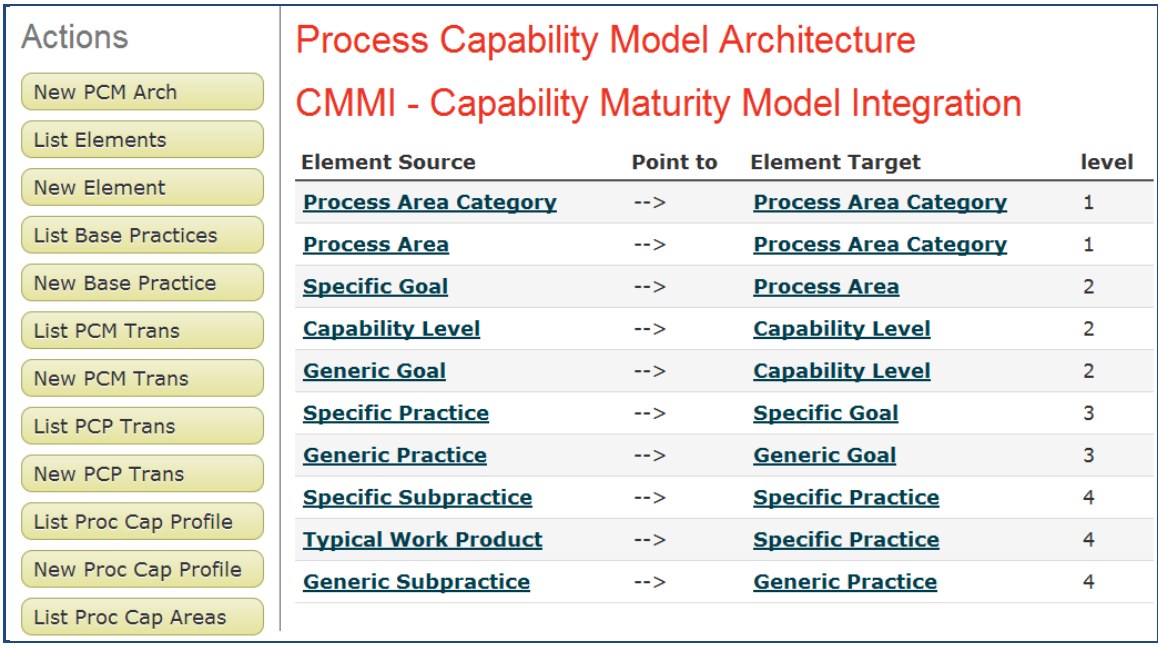


Figura 4.19 – Arquitetura CMMI.

A Figura 4.19 apresenta a arquitetura CMMI, após o cadastro de seus elementos e dos relacionamentos entre esses elementos.

Figura 4.20 – Arquitetura SPICE.

A Figura 4.20 apresenta a arquitetura SPICE, após o cadastro de seus elementos e dos relacionamentos entre esses elementos. Após o cadastro das arquiteturas dos modelos CMMI e SPICE a segunda etapa é cadastrar os componentes do modelo CMMI-DEV versão 1.3 (F 2), ou seja, o conteúdo do modelo de acordo com o seu elemento arquitetônico.

F 2 – Representar modelos de capacidade de processo.

F 2.a – Cadastrar/consultar as informações de um modelo.

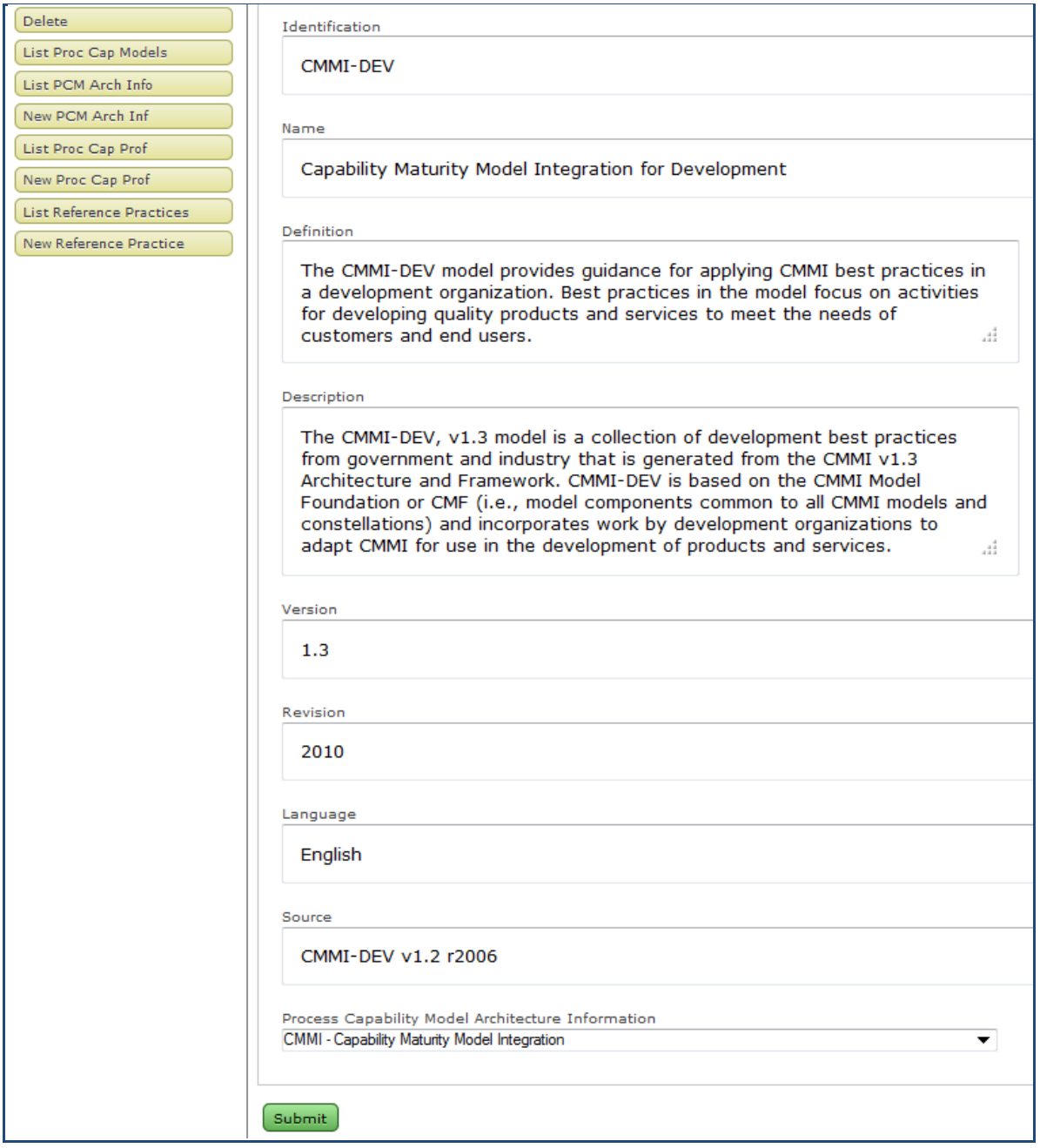


Figura 4.21 – Cadastro das informações do modelo CMMI-DEV versão 1.3.

A Figura 4.21 apresenta a tela de cadastro das informações do modelo CMMI-DEV versão 1.3, as informações cadastradas são a Identificação, o Nome, a Definição, a Descrição, a Versão, a Revisão, o Idioma, a Origem e a Informação da Arquitetura do Modelo de Capacidade de Processo.

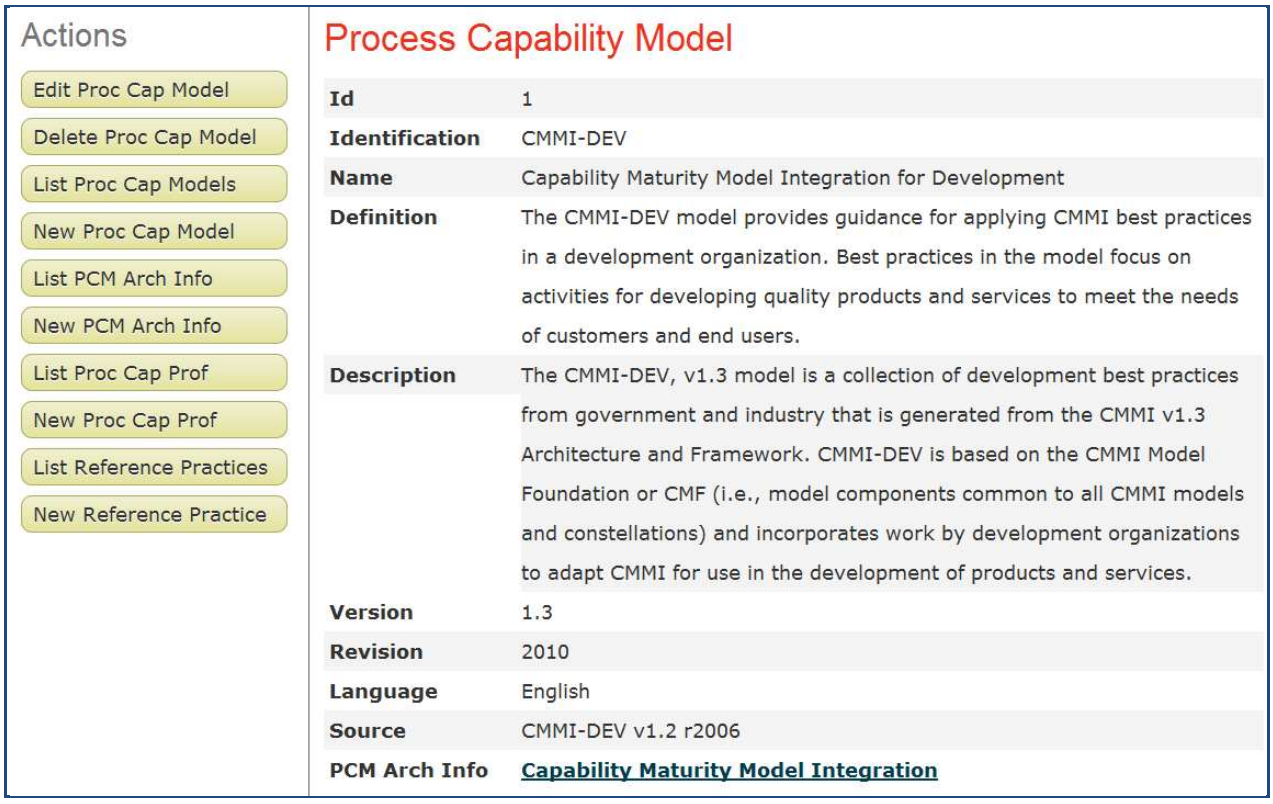


Figura 4.22 – Consulta das informações do modelo CMMI-DEV versão 1.3.

A Figura 4.22 apresenta a tela de consulta das informações do modelo CMMI-DEV versão 1.3.

F 2.b – Cadastrar/consultar os componentes de um modelo.

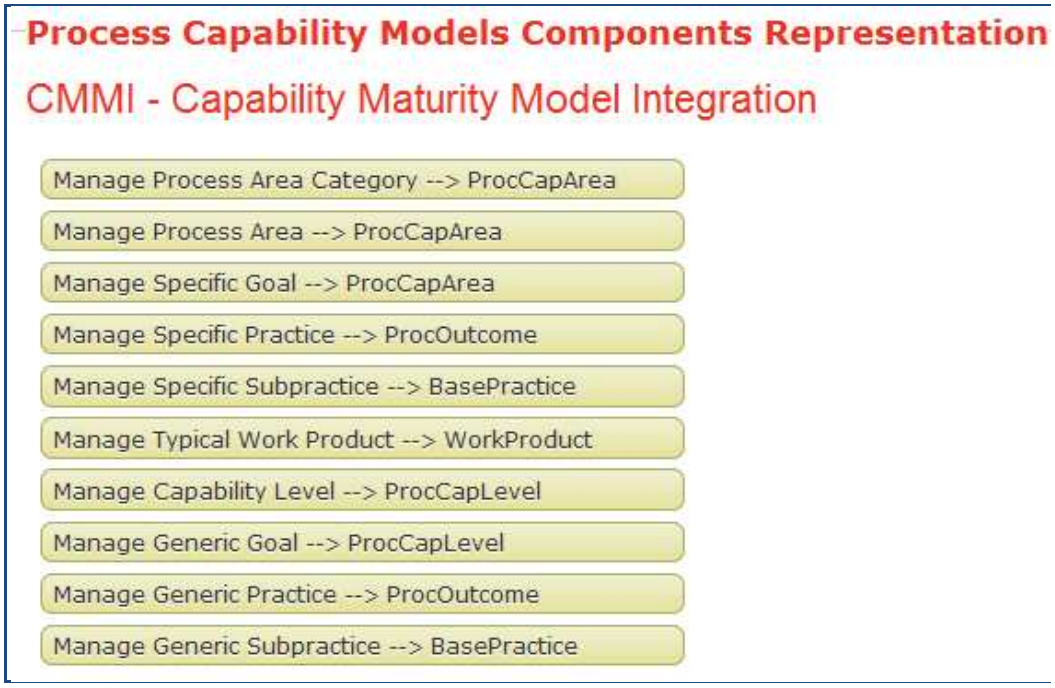


Figura 4.23 – Tela de seleção de cadastro de componentes do modelo CMMI.

A Figura 4.23 apresenta a tela de seleção de cadastro de componentes do modelo CMMI, após a seleção da arquitetura o PRO2PI-MMCA apresenta os componentes que podem ser cadastrados de acordo com a relação com o metamodelo PRO2PI-MMC, essa relação é criada no cadastro dos elementos da arquitetura dos modelos.

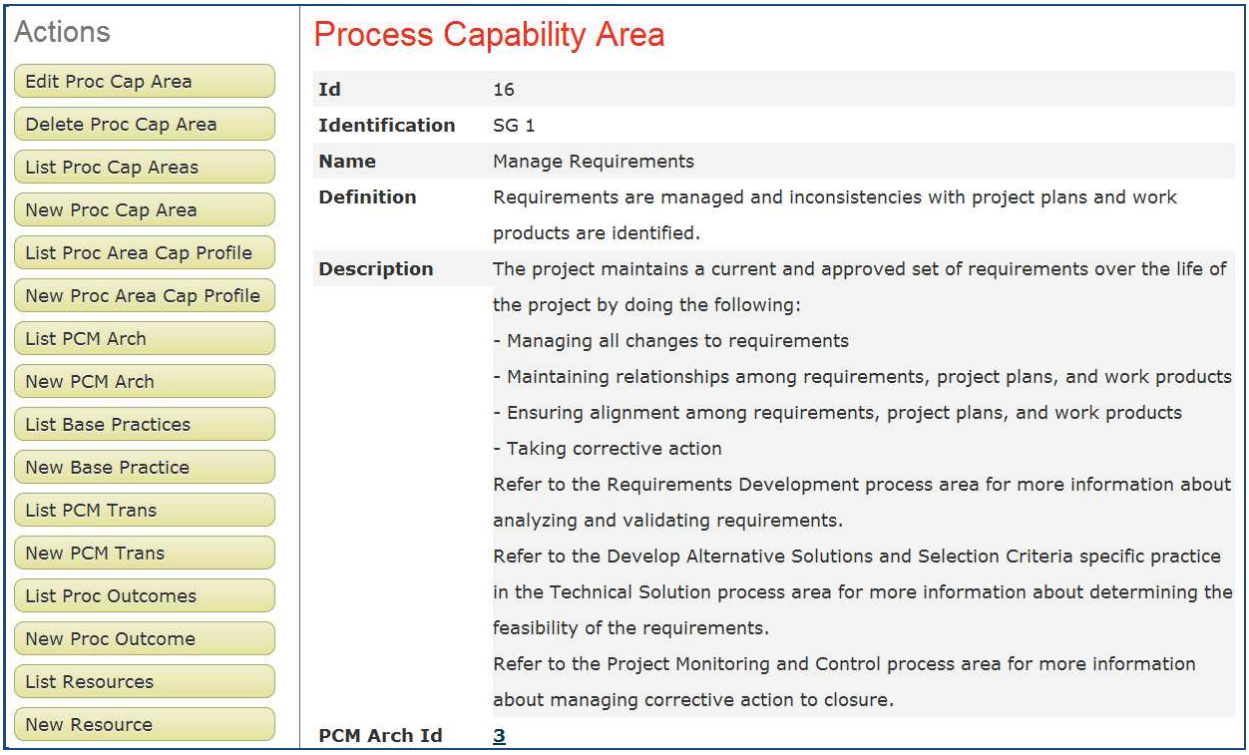


Figura 4.24 – Consulta do componente SG 1 do modelo CMMI-DEV versão 1.3.

A Figura 4.24 apresenta o componente SG 1 – Gerenciar Requisitos, as informações cadastradas foram a Identificação, o Nome, a Definição, a Descrição e o Identificador da Arquitetura do Modelo de Capacidade de Processo.

F 2.c – Cadastrar/consultar os relacionamentos entre os componentes de um modelo.

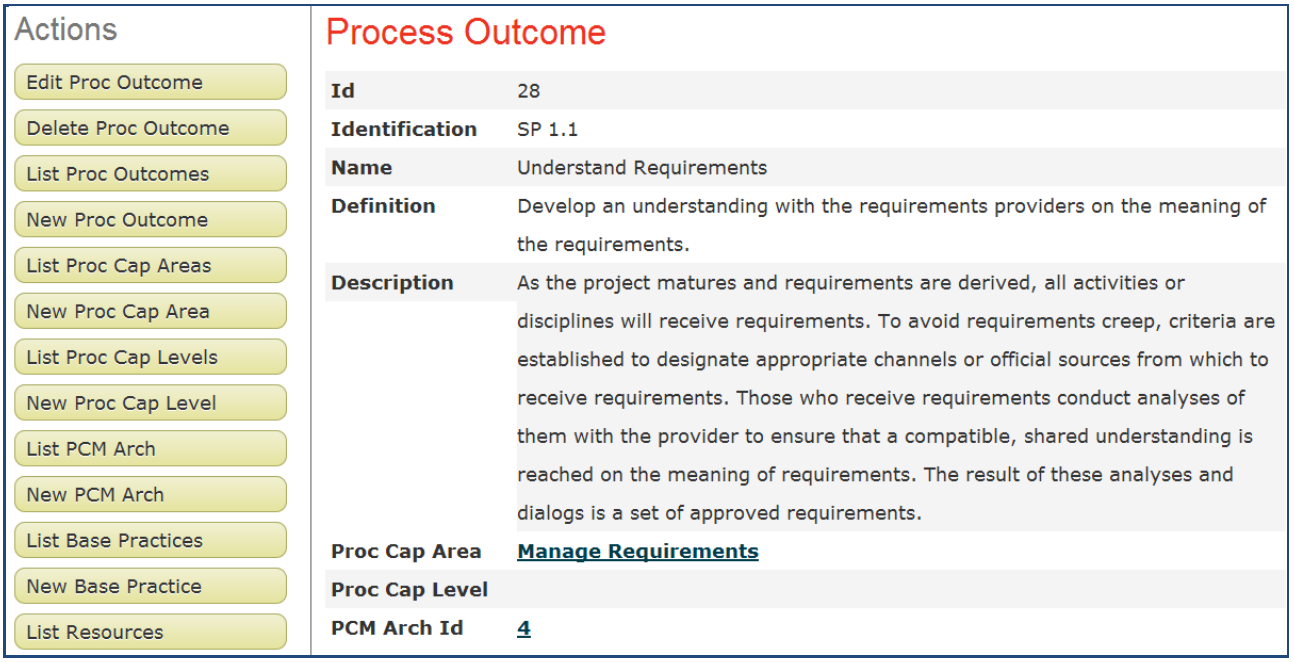


Figura 4.25 – Consulta do relacionamento entre os componentes SP 1.1 e SG 1 do modelo CMMI-DEV versão 1.3.

A Figura 4.25 apresenta a tela de consulta do relacionamento entre os componentes SP 1.1 e SG 1.1 do modelo CMMI-DEV versão 1.3, as informações cadastradas foram a Identificação, o Nome, a Definição, a Descrição do componente SP 1.1, o Relacionamento do SP 1.1 com o SG 1 e o Identificador da Arquitetura do Modelo de Capacidade de Processo do SP 1.1.

F 3 – Representar perfis de capacidade de processo. Essa etapa não foi necessária para ilustrar o exemplo, pois nenhum perfil de capacidade de processo foi criado.

A terceira etapa é a definição de regras adicionais de transformação (F 4), que complementam as regras gerais de transformação criadas automaticamente pelo PRO2PI-MMCA por meio do cadastro dos elementos arquitetônicos dos modelos CMMI e SPICE. As regras gerais e adicionais são utilizadas pelo PRO2PI-MMCA durante o processo de transformação de modelos. No exemplo o conteúdo do modelo CMMI-DEV versão 1.3 é transformado para a arquitetura SPICE.

F 4 – Estabelecer regras adicionais de transformação entre os elementos de arquiteturas dos modelos de capacidade de processo.

F 4.a – Cadastrar/consultar as regras de transformação entre os elementos de arquiteturas de modelos.

TR Type	PCM Architecture Source	PCM Architecture Target
General TR	Process Area Category	Process Category
General TR	Typical Work Product	Work Product
Additional TR	Empty	Process Group
Additional TR	Process Area	Process
Additional TR	Specific Goal	Process Outcome
Additional TR	Specific Practice	Base Practice
Additional TR	Capability Level	Capability Level
Additional TR	Generic Goal	Process Attribute
Additional TR	Generic Subpractice	Process Attribute Outcome
Additional TR	Empty	Generic Resource
Additional TR	Empty	Generic Work Product
Additional TR	Generic Practice	Generic Practice
Additional TR	Specific Subpractice	Empty

Figura 4.26 – Consulta das regras de transformação entre as arquiteturas CMMI e SPICE.

A Figura 4.26 apresenta as regras gerais e adicionais de transformação que são utilizadas durante o processo de transformação entre as arquiteturas dos modelos CMMI e SPICE.

A última etapa é realizar a transformação da área de processo REQM – *Requirements Management* – Gerência de Requisitos na arquitetura CMMI para a arquitetura SPICE (F 5).

F 5 – Realizar transformações de partes de modelos de capacidade de processo.

F 5.a – Apresentar uma área de processo/processo ou um nível de capacidade de um modelo de origem na arquitetura de um modelo de destino.

Quadro 4.3 – REQM – *Requirements Management* do modelo CMMI-DEV versão 1.3 na arquitetura SPICE.

Process Capability Area Transformation

REQM – Requirements Management of the CMMI-DEV – Capability Maturity Model Integration for Development version 1.3 from architecture of the CMMI – Capability Maturity Model Integration to architecture of the SPICE – ISO/IEC 15504 – Software Process Improvement and Capability dEtermination

Process REQM – Requirements Management

Definition: The purpose of Requirements Management (REQM) is to manage requirements of the project's products and product components and to ensure alignment between those requirements and the project's plans and work products.
Description: Requirements management processes manage all requirements received or generated by the project, including both technical and nontechnical requirements as well as requirements levied on the project by the organization. In particular, if the Requirements Development process area is implemented, its processes will generate product and product component requirements that will also be managed by the requirements management processes. Throughout the process areas, where the terms "product" and "product component" are used, their intended meanings also encompass services, service systems, and their components. When the Requirements Management, Requirements Development, and Technical Solution process areas are all implemented, their associated processes can be closely tied and be performed concurrently. The project takes appropriate steps to ensure that the set of approved requirements is managed to support the planning and execution needs of the project. (…)

This ***Process*** has the following ***Process Outcome***:

- ***Process Outcome*** SG 1 – Manage Requirements
  Definition: Requirements are managed and inconsistencies with project plans and work products are identified.
  Description: The project maintains a current and approved set of requirements over the life of the project by doing the following:
  - Managing all changes to requirements
  - Maintaining relationships among requirements, project plans, and work products
  - Ensuring alignment among requirements, project plans, and work products
  - Taking corrective action

  Refer to the Requirements Development process area for more information about analyzing and validating requirements.
  Refer to the Develop Alternative Solutions and Selection Criteria specific practice in the Technical Solution process area for more information about determining the feasibility of the requirements.
  Refer to the Project Monitoring and Control process area for more information about managing corrective action to closure.

  This ***Process Outcome*** has the following ***Base Practices***:

  - ***Base Practice*** SP 1.1 – Understand Requirements
    Definition: Develop an understanding with the requirements providers on the meaning of the requirements.
    Description: As the project matures and requirements are derived, all activities or disciplines will receive requirements. To avoid requirements creep, criteria are established to designate appropriate channels or official sources from which to receive requirements. Those who receive requirements conduct analyses of them with the provider to ensure that a compatible, shared understanding is reached on the meaning of requirements. The result of these analyses and dialogs is a set of approved requirements.

- ***Base Practice*** SP 1.2 – Obtain Commitment to Requirements
  Definition: Obtain commitment to requirements from project participants.
  Description: Refer to the Project Monitoring and Control process area for more information about monitoring commitments. The previous specific practice dealt with reaching an understanding with requirements providers. This specific practice deals with agreements and commitments among those who carry out activities necessary to implement requirements. Requirements evolve throughout the project. As requirements evolve, this specific practice ensures that project participants commit to the current and approved requirements and the resulting changes in project plans, activities, and work products.

- ***Base Practice*** SP 1.3 – Manage Requirements Changes
  Definition: Manage changes to requirements as they evolve during the project.
  Description: Refer to the Configuration Management process area for more information about tracking and controlling changes. Requirements change for a variety of reasons. As needs change and as work proceeds, changes may have to be made to existing requirements. It is essential to manage these additions and changes efficiently and effectively. (...)

- ***Base Practice*** SP 1.4 – Maintain Bidirectional Traceability of Requirements
  Definition: Maintain bidirectional traceability among requirements and work products.
  Description: The intent of this specific practice is to maintain the bidirectional traceability of requirements. (…)

- ***Base Practice*** SP 1.5 – Ensure Alignment Between Project Work and Requirements
  Definition: Ensure that project plans and work products remain aligned with requirements.
  Description: This specific practice finds inconsistencies between requirements and project plans and work products and initiates corrective actions to resolve them.

This ***Process Outcome*** has the following ***Work Products***:

- ***Work Product*** 1 – Lists of criteria for distinguishing appropriate requirements providers
  Definition:
  Description:

- ***Work Product 2 – Criteria for evaluation and acceptance of requirements***
  Definition:
  Description:

- ***Work Product*** 3 – Results of analyses against criteria
  Definition:
  Description:

- ***Work Product*** 4 – A set of approved requirements
  Definition:
  Description:

- ***Work Product*** 1 – Requirements impact assessments
  Definition:
  Description:

- ***Work Product*** 2 – Documented commitments to requirements and requirements changes
  Definition:
  Description:

- ***Work Product*** 1 – Requirements change requests

Definition:
Description:

- ***Work Product*** 2 – Requirements change impact reports
  Definition:
  Description:

- ***Work Product*** 3 – Requirements status
  Definition:
  Description:

- ***Work Product*** 4 – Requirements database
  Definition:
  Description:

- ***Work Product*** 1 – Requirements traceability matrix
  Definition:
  Description:

- ***Work Product*** 2 – Requirements tracking system
  Definition:
  Description:

- ***Work Product*** 1 – Documentation of inconsistencies between requirements and project plans and work products, including sources and conditions
  Definition:
  Description:

- ***Work Product*** 2 – Corrective actions
  Definition:
  Description:

Related Process Capability Areas

Name: Requirements Development

Information: Refer to the Requirements Development process area for more information about eliciting, analyzing, and establishing customer, product, and product component requirements.

Name: Technical Solution

Information: Refer to the Technical Solution process area for more information about selecting, designing, and implementing solutions to requirements.

"(...)" – Significa que o texto ou parte do modelo foi omitido para melhorar a legibilidade.

O Quadro 4.3 apresenta o resultado final do exemplo, ou seja, a área de processo REQM – *Requirements Management* – Gerência de Requisitos do modelo CMMI-DEV versão 1.3 na arquitetura SPICE. A transformação de um perfil de capacidade de processo a partir de múltiplos modelos encontra-se no Apêndice A.

"Você pode encarar um erro como uma besteira a ser esquecida,
ou como um resultado que aponta uma nova direção."
(Steve Jobs)

5 Avaliação dos resultados do PRO2PI-MMCA e análise da avaliação

Este capítulo apresenta a realização de um Painel de Especialistas (*Expert Panel*) simplificado para avaliar os resultados do PRO2PI-MMCA e uma análise da avaliação. O objetivo da avaliação é obter a indicação por parte dos especialistas que a ferramenta gera resultados que atendem a sua finalidade. Além de confirmar o correto mapeamento entre os elementos arquitetônicos dos modelos utilizados no trabalho. Os resultados avaliados são seis Perfis de Capacidade de Processo específicos gerados pelo software PRO2PI-MMCA. São apresentados o que é um Painel de Especialistas, os critérios utilizados na seleção dos especialistas, os resultados avaliados e o instrumento de avaliação, a consolidação e a análise dos dados coletados na avaliação, as ameaças à validade da avaliação e por último uma discussão.

5.1 Critérios de seleção dos especialistas para avaliação por Painel de Especialistas

Painel de Especialistas (*Expert Panel*) é uma técnica baseada na opinião de especialistas utilizada para validar novos métodos e técnicas (Beecham et al. 2005), utilizando-se para isso de instrumentos de avaliações para coleta e análise de dados. A técnica consiste em reunir especialistas de determinada área de conhecimento com o intuito de obter a opinião dos mesmos em relação a algum aspecto do objeto de pesquisa.

Como o projeto de pesquisa foi exercitado para três modelos: CMMI-DEV, ASPICE e MR-MPS-SW, os especialistas devem conhecer e ter experiência prática em pelo menos dois dos três modelos. Isso porque o objetivo é avaliar se o conteúdo representativo de um determinado modelo de capacidade de processo foi expresso corretamente do ponto de vista lógico na arquitetura de outro modelo de capacidade de processo. Na seleção dos especialistas foi considerado também o fato de estes terem publicações na área de Melhoria de Processo de Software.

A Figura 5.1 apresenta o esquema de avaliação que foi utilizado, por exemplo: um especialista devia ter conhecimento e experiência prática no CMMI-DEV e no MR-MPS-SW para avaliar se o conteúdo da área de processo "REQM – *Requirements Management*" do modelo CMMI-DEV versão 1.3 da arquitetura CMMI foi expresso corretamente na arquitetura MPS.BR. O esquema de avaliação apresenta todas as relações possíveis entre as arquiteturas CMMI, SPICE e MPS.BR utilizada pelos respectivos modelos CMMI-DEV versão 1.3, ASPICE versão 2.5 e MR-MPS-SW versão agosto/2012.

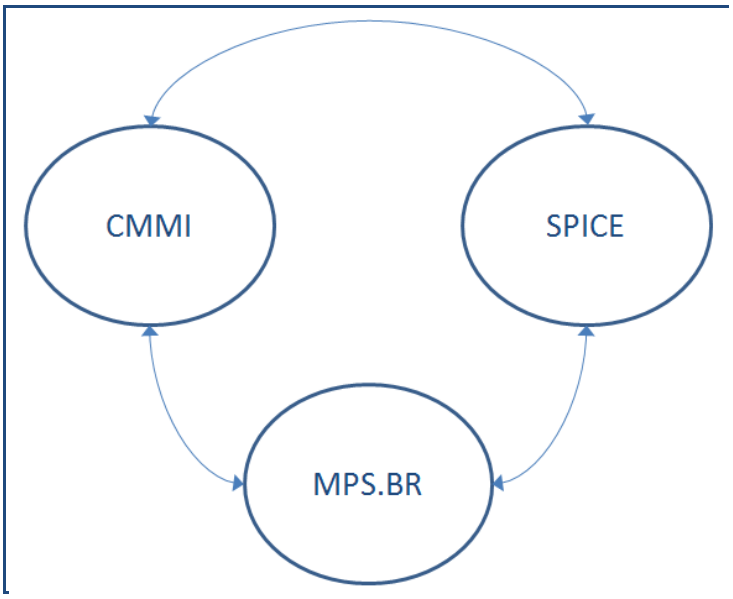


Figura 5.1 – Esquema de Avaliação.

5.2 Perfis de Capacidade de Processo avaliados

A melhoria de processo é feita com base em perfis de capacidade de processo. Um perfil de capacidade de processo é composto por um ou mais pares de área de capacidade de processo/processo e nível de capacidade. A ferramenta PRO2PI-MMCA gera os perfis de capacidade de processo a partir do cadastro de partes específicas dos modelos de capacidade. Estas partes específicas são as áreas de processos/processos e os níveis de capacidade obtidos dos modelos. Para facilitar e agilizar o cadastro das partes específicas dos modelos foram desenvolvidos dois scripts em PHP, o script de extração e o script de cadastro.

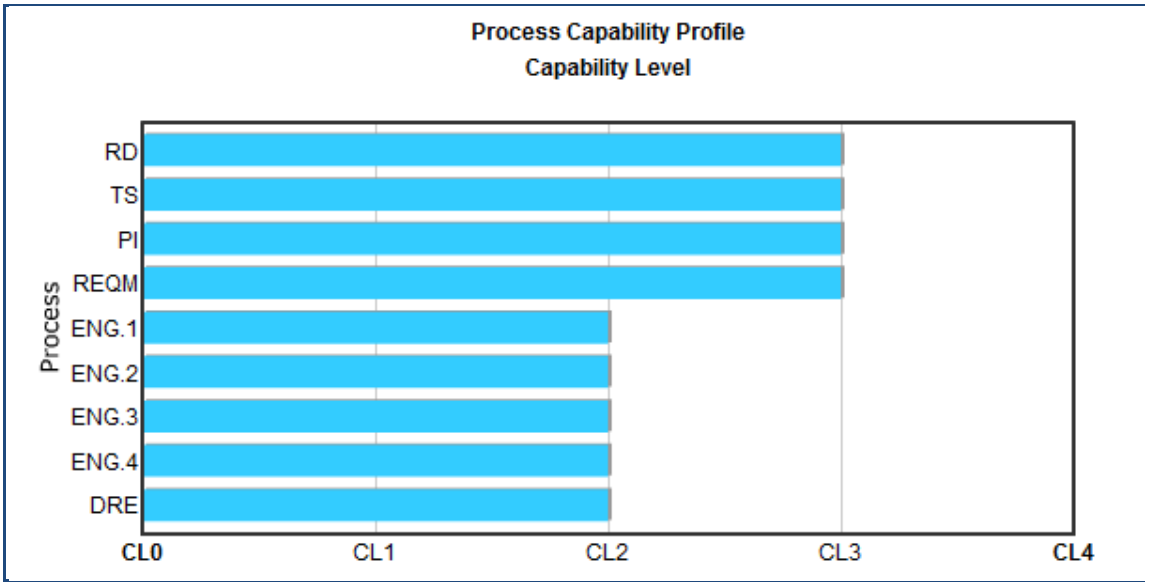
O script de extração dos textos das partes específicas dos modelos faz a leitura de um arquivo texto que contém o conteúdo de uma área de processo/processo ou nível de capacidade do modelo, que foi previamente demarcado com algumas *tags* e gera um novo arquivo texto com o conteúdo já formatado. O script de cadastro dos textos das partes específicas dos modelos faz a leitura do arquivo gerado anteriormente pelo script de extração e cadastra o conteúdo do arquivo no banco de dados do PRO2PI-MMCA. Antes de utilizar o script de cadastro é necessário já ter cadastrado as arquiteturas CMMI, SPICE e MPS.BR. As partes específicas dos modelos cadastradas no PRO2PI-MMCA pelo script de cadastro são apresentadas na Tabela 5.1.

Tabela 5.1 – Áreas de Processo/Processos e Níveis de Capacidade cadastrados no PRO2PI-MMCA.

Modelo	Área de Processo/Processo	Nível de Capacidade
CMMI-DEV (versão 1.3)	RD – *Requirements Development* TS – *Technical Solution* PI – *Product Integration* REQM – *Requirements Management*	Nível 3 – Definido
ASPICE (versão 2.5)	ENG.1 – *Requirements elicitation* ENG.2 – *System requirements analysis* ENG.3 – *System architectural design* ENG.4 – *Software requirements analysis*	Nível 2 – Gerenciado
MR-MPS-SW (versão agosto/2012)	DRE – Desenvolvimento de Requisitos	Nível 2 – Gerenciado

O PRO2PI-MMCA possui uma funcionalidade que permite gerar um gráfico de barras que representa a transformação de um perfil de capacidade de processo específico. A Figura 5.2 apresenta o gráfico de barras do perfil de capacidade de processo de engenharia na arquitetura CMMI, gerado como um exemplo, que contém todas as áreas de processo/processos em seus respectivos níveis de capacidade conforme apresentado na Tabela 5.1.

Figura 5.2 – Perfil de Capacidade de Processo de Engenharia.



Para realizar a avaliação, seis resultados do PRO2PI-MMCA foram gerados. Esses resultados são transformações de perfis de capacidade de processo nas arquiteturas CMMI, SPICE e MPS.BR. Foram escolhidos processos de requisitos, pois os mesmos possuem uma maior independência em relação aos demais processos dos modelos. Os resultados são:

1) A área de processo "REQM – *Requirements Management*" do modelo CMMI-DEV versão 1.3 expressa na arquitetura SPICE;
2) A área de processo "REQM – *Requirements Management*" do modelo CMMI-DEV versão 1.3 expressa na arquitetura MPS.BR;
3) O processo "DRE – Desenvolvimento de Requisitos" do modelo MR-MPS-SW versão agosto/2012 expresso na arquitetura CMMI;
4) O processo "DRE – Desenvolvimento de Requisitos" do modelo MR-MPS-SW versão agosto/2012 expresso na arquitetura SPICE;

5) O processo "ENG.1 – *Requirements elicitation*" do modelo ASPICE versão 2.5 expresso na arquitetura CMMI; e
6) O processo "ENG.1 – *Requirements elicitation*" do modelo ASPICE versão 2.5 expresso na arquitetura MPS.BR.

O Quadro 5.1 apresenta o resultado "2) A área de processo "REQM – *Requirements Management*" do modelo CMMI-DEV versão 1.3 expressa na arquitetura MPS.BR". Os demais resultados do PRO2PI-MMCA avaliados encontram-se no Apêndice B.

Quadro 5.1 – Área de processo "REQM – *Requirements Management*" do modelo CMMI-DEV versão 1.3 expressa na arquitetura MPS.BR.

Process REQM – Requirements Management
Definition: The purpose of Requirements Management (REQM) is to manage requirements of the project's products and product components and to ensure alignment between those requirements and the project's plans and work products. ***Description:*** Requirements management processes manage all requirements received or generated by the project, including both technical and nontechnical requirements as well as requirements levied on the project by the organization. In particular, if the Requirements Development process area is implemented, its processes will generate product and product component requirements that will also be managed by the requirements management processes. Throughout the process areas, where the terms "product" and "product component" are used, their intended meanings also encompass services, service systems, and their components. When the Requirements Management, Requirements Development, and Technical Solution process areas are all implemented, their associated processes can be closely tied and be performed concurrently. (…)

This ***Process*** has the following ***Process Outcomes***:

- ***Process Outcome*** SP 1.1 – Understand Requirements
  Definition: Develop an understanding with the requirements providers on the meaning of the requirements. ***Description:*** As the project matures and requirements are derived, all activities or disciplines will receive requirements. To avoid requirements creep, criteria are established to designate appropriate channels or official sources from which to receive requirements. Those who receive requirements conduct analyses of them with the provider to ensure that a compatible, shared understanding is reached on the meaning of requirements. The result of these analyses and dialogs is a set of approved requirements.

- ***Process Outcome*** SP 1.2 – Obtain Commitment to Requirements
  Definition:Obtain commitment to requirements from project participants.
  Description: Refer to the Project Monitoring and Control process area for more information about monitoring commitments. The previous specific practice dealt with reaching an understanding with requirements providers. This specific practice deals with agreements and commitments among those who carry out activities necessary to implement requirements. Requirements evolve throughout the project. As requirements evolve, this specific practice ensures that project participants commit to the current and approved requirements and the resulting changes in project plans, activities, and work products.

- ***Process Outcome*** SP 1.3 – Manage Requirements Changes
  Definition: Manage changes to requirements as they evolve during the project.
  Description: Refer to the Configuration Management process area for more information about tracking and controlling changes. Requirements change for a variety of reasons. As needs change and as work proceeds, changes may have to be made to existing requirements. It is essential to manage these additions and changes efficiently and effectively. To effectively analyze the impact of changes, it is necessary that the source of each requirement is known and the rationale for the change is documented. The project may want to track appropriate measures of requirements volatility to judge whether new or revised approach to change control is necessary.

- ***Process Outcome*** SP 1.4 – Maintain Bidirectional Traceability of Requirements
  Definition: Maintain bidirectional traceability among requirements and work products. ***Description:*** The intent of this specific practice is to maintain the bidirectional traceability of requirements. (See the

definition of "bidirectional traceability" in the glossary.) When requirements are managed well, traceability can be established from a source requirement to its lower level requirements and from those lower level requirements back to their source requirements. (…)

- ***Process Outcome*** SP 1.5 – Ensure Alignment Between Project Work and Requirements
  Definition: Ensure that project plans and work products remain aligned with requirements.
  Description: This specific practice finds inconsistencies between requirements and project plans and work products and initiates corrective actions to resolve them.

"(...)" – Significa que o texto ou parte do modelo foi omitido para melhorar a legibilidade.

5.3 Questionário de avaliação

O instrumento de avaliação escolhido para coletar os dados para análise foi o questionário, pois os especialistas estão distribuídos em diversos locais e não estão diretamente disponíveis para entrevista. Uma parte do questionário para avaliação de um resultado específico é apresentado no Quadro 5.2. O questionário de avaliação utilizou algumas recomendações da Enciclopédia de Métodos de Pesquisa de Opinião (Lavrakas 2008). Uma das recomendações na elaboração do questionário é utilizar questões fechadas, pois aumentam a capacidade de comparar respostas. Além disso, os dados podem ser facilmente quantificados, o que poupa tempo de processamento de dados. As questões fechadas também são ideais para questionários auto-administrados, porque evitam uma maior subjetividade e volatilidade em relação às perguntas abertas. Esses questionários são projetados para serem respondidos sem a intervenção do pesquisador, como por exemplo, uma entrevista.

Desta forma o questionário de avaliação foi elaborado com duas questões, uma questão fechada com uma resposta tricotômica, que representa uma escala de atendimento entre 100% – Contempla Totalmente, 50% – Contempla Parcialmente e 0% – Não Contempla. E para as respostas Contempla Parcialmente e Não Contempla buscou-se saber por meio de uma justificativa o motivo do não atendimento. A segunda questão é uma questão aberta, onde o avaliador tem a liberdade de expressar a sua opinião sobre o resultado avaliado. Além disso, foi perguntado o nome do avaliador que é de preenchimento opcional e a sua experiência nos modelos de capacidade processo.

Quadro 5.2 – Questionário utilizado para avaliar a área de processo REQM do CMMI-DEV versão 1.3 na arquitetura MPS.BR.

Nome do Especialista (Opcional):
Você possui conhecimento e/ou experiência em qual(is) modelo(s)? ( ) CMMI-DEV versão 1.3 ( ) ASPICE versão 2.5 ( ) MR-MPS-SW versão agosto/2012
Quanto tempo de experiência no(s) modelo(s) você possui?
1) O conteúdo da área de processo REQM do CMMI-DEV versão 1.3 foi expresso corretamente do ponto de vista lógico na arquitetura MPS.BR? a. Contempla Totalmente. b. Contempla Parcialmente. c. Não Contempla. Caso tenha respondido b ou c, favor justificar.
2) Comentários gerais: a. Quais os pontos fortes apresentados no resultado? b. Quais os pontos fracos apresentados no resultado? c. Quais as suas sugestões de melhoria? d. De uma forma geral o resultado apresentado foi satisfatório?

O questionário completo com os resultados do PRO2PI-MMCA utilizado na avaliação encontra-se no Apêndice B.

5.4 Resultado e análise do Painel de Especialistas

Esta seção apresenta a análise dos dados coletados e consolidados na avaliação dos resultados do PRO2PI-MMCA por meio dos questionários utilizados como instrumento de avaliação no Painel de Especialistas. Com base nos critérios de seleção dos especialistas foram selecionados oito especialistas em modelos de capacidade de processo para participar do painel de especialistas. Os questionários com os resultados a serem avaliados foram enviados por e-mail aos especialistas. Os e-mails foram obtidos em artigos publicados. Dos oito e-mails enviados, seis avaliações foram recebidas.

Uma visão geral da composição do painel de especialistas é apresentada por meio da formação acadêmica e da experiência dos seis especialistas participantes. E logo após é apresentado à avaliação e a análise de cada resultado específico do PRO2PI-MMCA. As justificativas dos

especialistas para a primeira questão, quando as respostas foram “Contempla Parcialmente” e “Não Contempla” foram classificadas em dois grupos, “Avaliou a Formatação” e “Erro Arquitetônico”.

Avaliou a Formatação significa que o especialista durante a avaliação entendeu que a disposição ou a formatação textual dos elementos arquitetônicos, seguidos dos seus respectivos conteúdos do modelo é importante e deve ser atendido durante o processo de transformação de perfis de capacidade. Erro Arquitetônico significa que o especialista durante a avaliação encontrou um erro no atendimento e na representação das relações lógicas entre os elementos que compõem a arquitetura dos modelos. O foco da avaliação pelo painel de especialistas é encontrar erros arquitetônicos.

5.4.1 Formação Acadêmica dos Especialistas em Modelos de Capacidade de Processo

A formação acadêmica dos especialistas em modelos de capacidade de processo foi obtida por meio de consulta ao Currículo Lattes[6]. A Figura 5.3 apresenta a formação acadêmica dos seis especialistas em modelos. Com base no gráfico é possível concluir que existe uma alta formação acadêmica por parte dos especialistas, pois um terço dos especialistas são doutores.

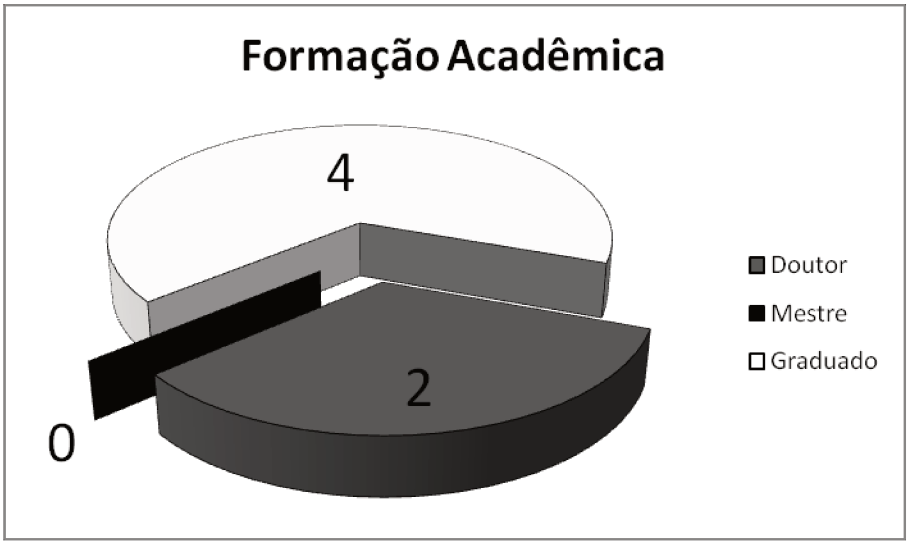


Figura 5.3 – Formação Acadêmica dos Especialistas em Modelos de Capacidade de Processo.

[6] Plataforma Lattes. http://lattes.cnpq.br/

5.4.2 Experiência dos Especialistas em Modelos de Capacidade de Processo

A Figura 5.4 apresenta a experiência dos seis especialistas em modelos de capacidade de processo.

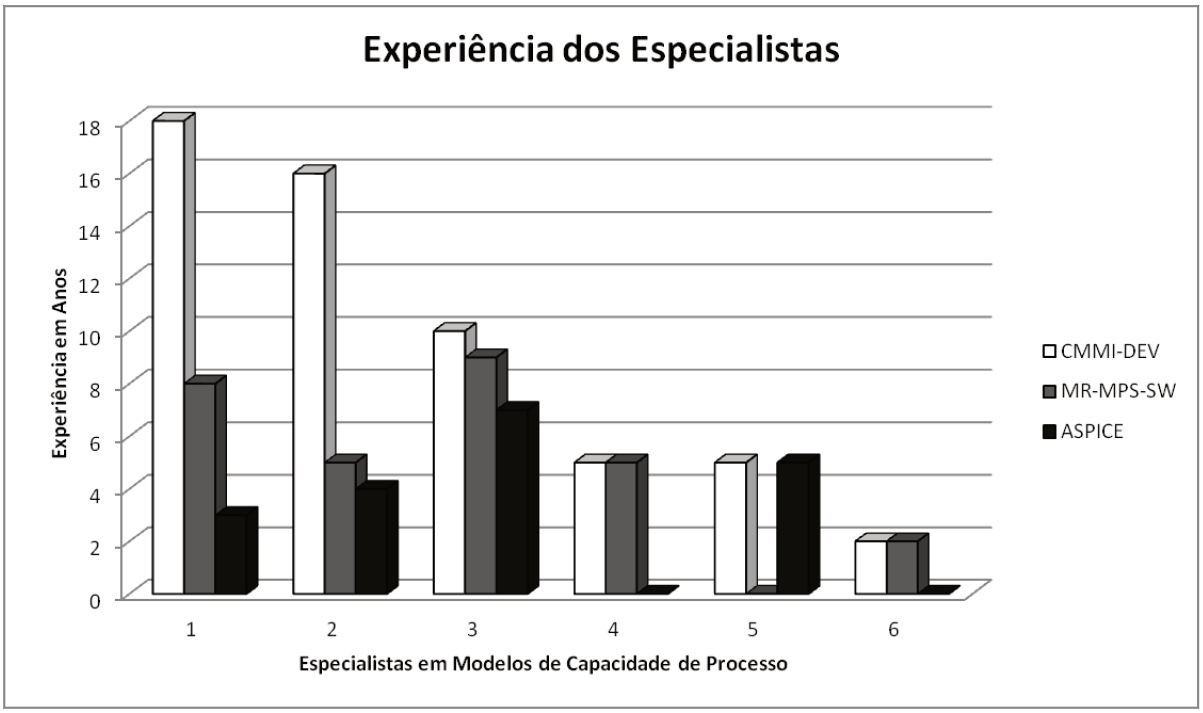


Figura 5.4 – Experiência dos Especialistas em Modelos de Capacidade de Processo.

A experiência em modelos considerou também versões anteriores dos modelos utilizados no trabalho. Com base no gráfico é possível concluir que a maior experiência dos especialistas é no modelo CMMI-DEV, seguido pelos modelos MR-MPS-SW e ASPICE. Além disso, nota-se que os especialistas possuem vasta experiência nos modelos utilizados no trabalho, pois nenhum especialista apresentou experiência menor do que dois anos. O especialista com maior experiência tem dezoito anos de atuação com o modelo CMMI-DEV, considerando neste caso versões anteriores do modelo CMMI-DEV e o modelo SW-CMM, que deu origem ao CMMI-DEV.

5.4.3 Área de processo "REQM – *Requirements Management*" do modelo CMMI-DEV versão 1.3 expressa na arquitetura SPICE

Quatro especialistas avaliaram o resultado do PRO2PI-MMCA: Área de processo "REQM – *Requirements Management*" do modelo CMMI-DEV versão 1.3 expressa na arquitetura SPICE. São apresentadas a avaliação e a análise do resultado.

Questão 1) O conteúdo da área de processo REQM do CMMI-DEV versão 1.3 foi expresso corretamente do ponto de vista lógico na arquitetura SPICE?

A Figura 5.5 apresenta as respostas para a questão 1. Um especialista considera que contempla parcialmente e três consideram que contempla totalmente.

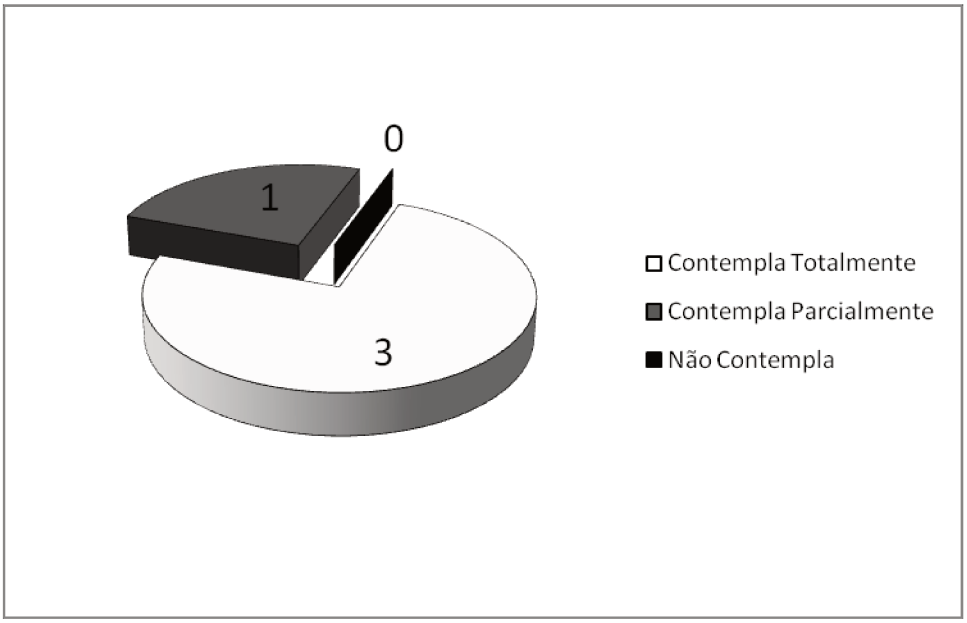


Figura 5.5 – Avaliação da área de processo REQM do CMMI-DEV versão 1.3 na arquitetura SPICE.

A Figura 5.6 apresenta a classificação da justificativa da avaliação do resultado como contempla parcialmente.

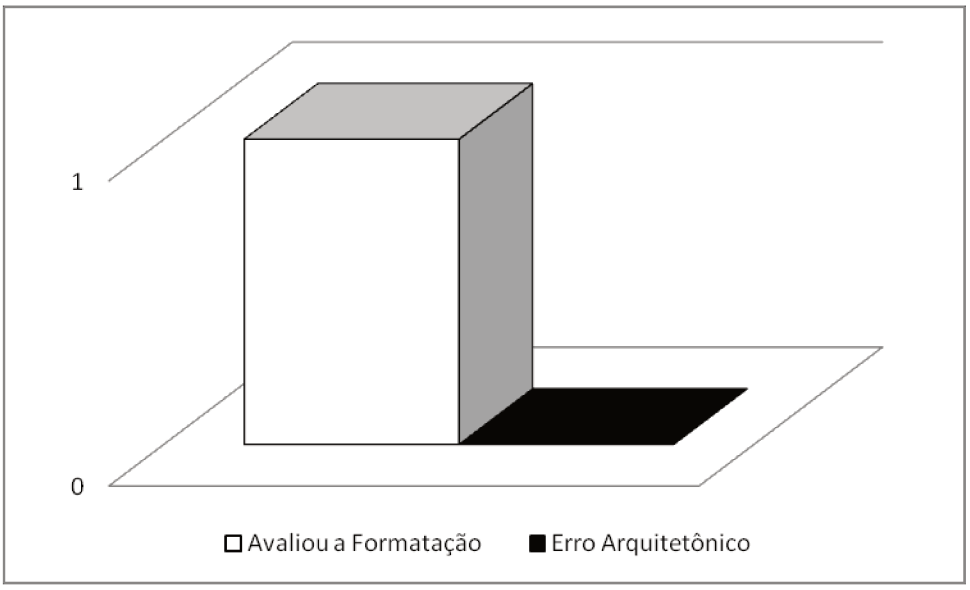


Figura 5.6 – Classificação da justificativa da avaliação da área de processo REQM do CMMI-DEV versão 1.3 na arquitetura SPICE.

Um especialista avaliou que contempla parcialmente, pois entendeu que ao realizar a transformação de modelos, que a relação existente na arquitetura CMMI entre os elementos *Specific Practice* e *Typical Work Product* não são apresentadas na arquitetura SPICE. E que o elemento *Specific Subpractice* da arquitetura CMMI está ausente na arquitetura SPICE.

O objetivo ao realizar a transformação de modelos, é o atendimento e a representação correta das relações lógicas entre os elementos que compõem a arquitetura de destino, no caso a arquitetura SPICE. A arquitetura SPICE relaciona os elementos *Process Outcome* e *Work Product* e esse relacionamento é atendido corretamente. O elemento *Specific Subpractice* da arquitetura CMMI não tem elemento correspondente na arquitetura SPICE, por isso o elemento está ausente.

Questão 2) Comentários gerais:

O Quadro 5.3 apresenta os comentários gerais obtidos na avaliação para a área de processo REQM do CMMI-DEV versão 1.3 na arquitetura SPICE.

Quadro 5.3 – Comentários gerais da área de processo REQM do CMMI-DEV versão 1.3 na arquitetura SPICE.

Pontos fortes apresentados no resultado
- Correto mapeamento dos aspectos.
Pontos fracos apresentados no resultado
- Ao listar todos os "work products" juntos, após as "base practices", o modelo não explicita a correlação "specific practices" versus "typical work products", presente no CMMI. - As "specific subpractices" do CMMI estão ausentes.
Sugestões de melhoria
- Se for pertinente, inserir elementos lógicos para associar "base practices" a "work products" e incluir "specific subpractices". - Apresentar um mapa de correspondência dos elementos do Modelo Origem X Modelo Destino. - Como os "work products" no CMMI são das "specific practices", seria interessante manter a identificação das "specific practices" (algo como SP 1.1 – 1).
De uma forma geral o resultado apresentado foi satisfatório?
- Sim.

As sugestões de melhoria serão analisadas e caso adequadas, serão implementadas em uma nova versão do software PRO2PI-MMCA.

5.4.4 Área de processo "REQM – *Requirements Management*" do modelo CMMI-DEV versão 1.3 expressa na arquitetura MPS.BR

Cinco especialistas avaliaram o resultado do PRO2PI-MMCA: Área de processo "REQM – *Requirements Management*" do modelo CMMI-DEV versão 1.3 expressa na arquitetura MPS.BR. São apresentadas a avaliação e a análise do resultado.

Questão 1) O conteúdo da área de processo REQM do CMMI-DEV versão 1.3 foi expresso corretamente do ponto de vista lógico na arquitetura MPS.BR? A Figura 5.7 apresenta as respostas para a questão 1. Três especialistas consideram que contempla parcialmente e dois consideram que contempla totalmente.

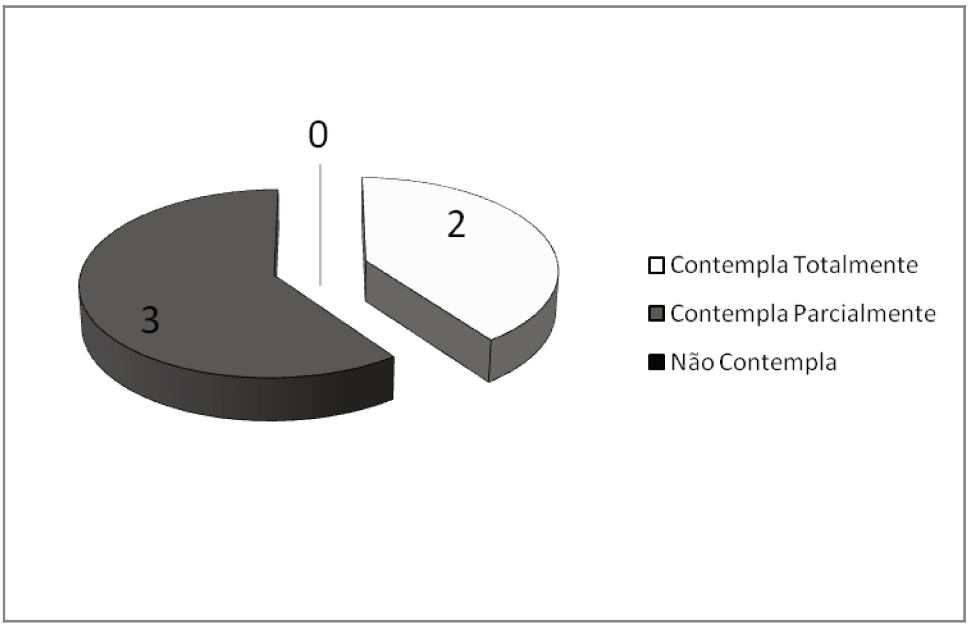


Figura 5.7 – Avaliação da área de processo REQM do CMMI-DEV versão 1.3 na arquitetura MPS.BR.

A Figura 5.8 apresenta a classificação das justificativas das avaliações do resultado como contempla parcialmente.

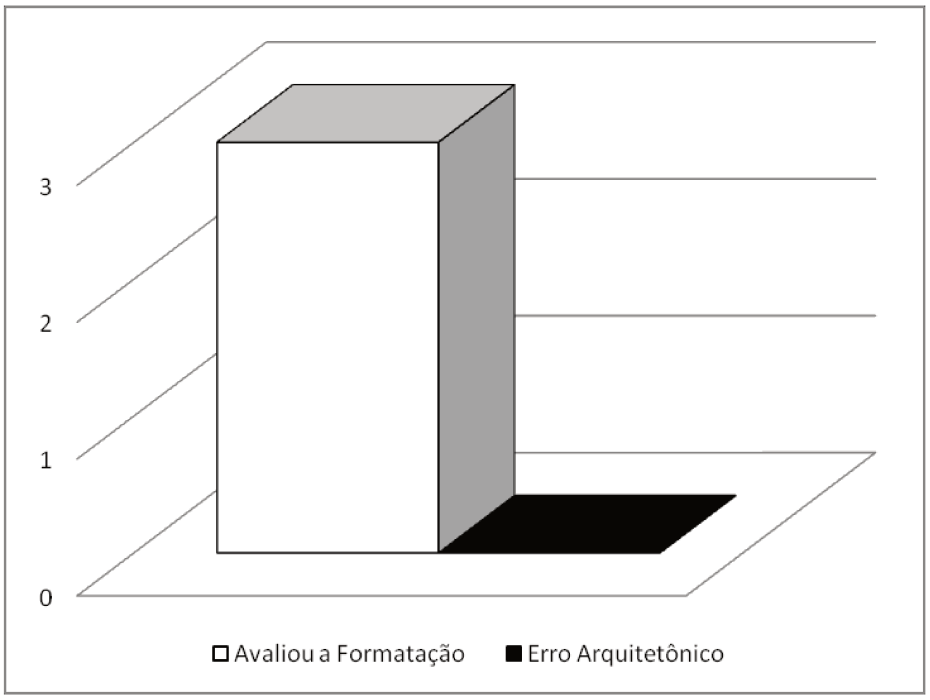


Figura 5.8 – Classificação das justificativas da avaliação da área de processo REQM do CMMI-DEV versão 1.3 na arquitetura MPS.BR.

Três especialistas avaliaram que contempla parcialmente, pois entenderam que ao realizar a transformação de modelos, que a correspondência entre o elemento *Specific Practice* da arquitetura CMMI com o elemento *Process Outcome* da arquitetura MPS.BR não é adequada e os campos *Definition* e *Description* não deveriam estar presentes no resultado.

Segundo o mapeamento existente no guia de implementação e avaliação do MR-MPS-SW versão agosto/2012 em conjunto com o CMMI-DEV versão 1.3 (SOFTEX 2012c), a correspondência entre o elemento *Specific Practice* da arquitetura CMMI com o elemento *Process Outcome* da arquitetura MPS.BR está correta e pode ser utilizada. Os campos *Definition* e *Description* estão presentes no resultado para permitir a rastreabilidade do resultado com os atributos do metamodelo PRO2PI-MMC, o que não influencia na representação da arquitetura MPS.BR.

Questão 2) Comentários gerais:

O Quadro 5.4 apresenta os comentários gerais obtidos na avaliação para a área de processo REQM do CMMI-DEV versão 1.3 na arquitetura MPS.BR.

Quadro 5.4 – Comentários gerais da área de processo REQM do CMMI-DEV versão 1.3 na arquitetura MPS.BR.

Pontos fortes apresentados no resultado
- A captura do propósito da área de processo/processo.
Pontos fracos apresentados no resultado
- A correspondência entre o elemento "specific practice" da arquitetura CMMI com o elemento "process outcome" da arquitetura MPS.BR.
Sugestões de melhoria
- Apresentar um mapa de correspondência dos elementos do Modelo Origem X Modelo Destino. - Trocar o nome do elemento "process outcome" para "resultado esperado" na arquitetura MPS.BR.
De uma forma geral o resultado apresentado foi satisfatório?
- Sim.

As sugestões de melhoria serão analisadas e caso adequadas, serão implementadas em uma nova versão do software PRO2PI-MMCA.

5.4.5 Processo "DRE – Desenvolvimento de Requisitos" do modelo MR-MPS-SW versão agosto/2012 expresso na arquitetura CMMI

Cinco especialistas avaliaram o resultado do PRO2PI-MMCA: Processo "DRE – Desenvolvimento de Requisitos" do modelo MR-MPS-SW versão agosto/2012 expresso na arquitetura CMMI. São apresentadas a avaliação e a análise do resultado.

Questão 1) O conteúdo do processo DRE do MR-MPS-SW versão agosto/2012 foi expresso corretamente do ponto de vista lógico na arquitetura CMMI? A Figura 5.9 apresenta as respostas para a questão 1. Dois especialistas consideram que contempla parcialmente, dois consideram que contempla totalmente e um considera que não contempla.

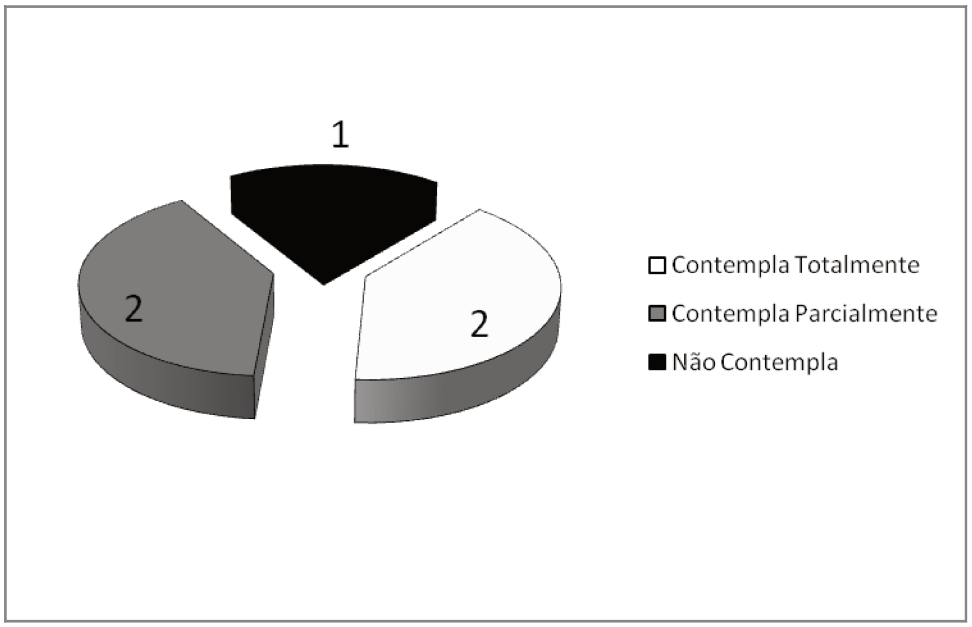


Figura 5.9 – Avaliação do processo DRE do MR-MPS-SW versão agosto/2012 na arquitetura CMMI.

A Figura 5.10 apresenta a classificação das justificativas da avaliação do resultado como contempla parcialmente e não contempla.

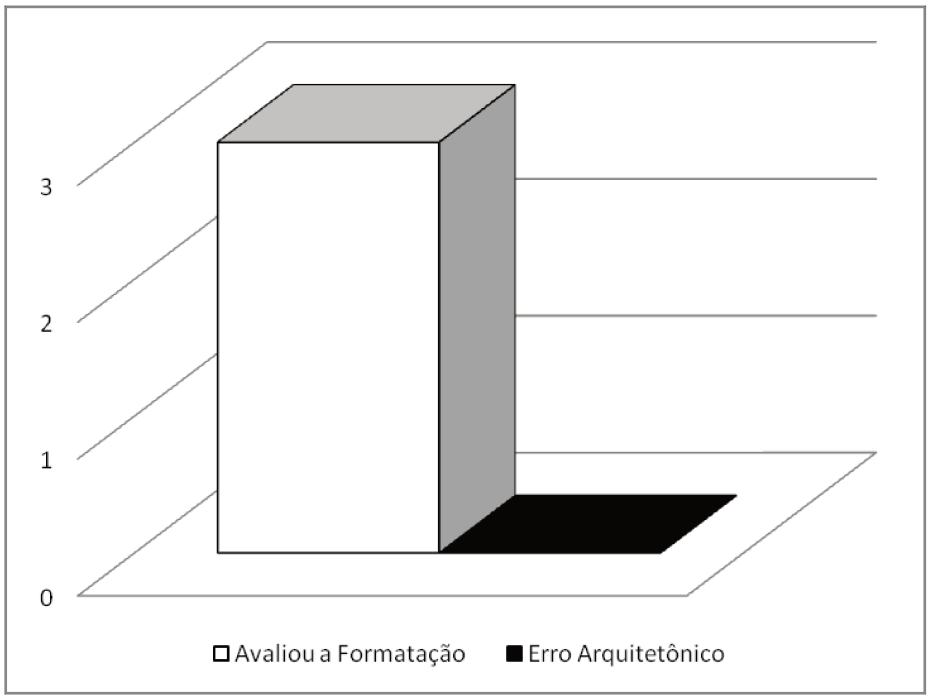


Figura 5.10 – Classificação das justificativas da avaliação do processo DRE do MR-MPS-SW versão agosto/2012 na arquitetura CMMI.

Dois especialistas avaliaram que contempla parcialmente, pois entenderam que ao realizar a transformação de modelos, que os elementos *Typical Work Product* e *Specific Subpractices* da arquitetura CMMI, que são importantes para o entendimento, não foram preenchidos com um conteúdo representativo. Um especialista avaliou que não contempla, pois os elementos *Specific Goal*, *Typical Work Product* e *Specific Subpractices* da arquitetura CMMI, não foram preenchidos com um conteúdo representativo.

Segundo o mapeamento existente no guia de implementação e avaliação do MR-MPS-SW versão agosto/2012 em conjunto com o CMMI-DEV versão 1.3 (SOFTEX 2012c), o elemento *Specific Goal* da arquitetura CMMI não possui elemento correspondente na arquitetura MPS.BR e é atendido indiretamente pelo elemento *Specific Practice* da arquitetura CMMI, que corresponde ao elemento *Process Outcome* da arquitetura MPS.BR. Os elementos *Typical Work Product* e *Specific Subpractices* da arquitetura CMMI são informativos e não possuem elementos correspondentes diretos na arquitetura MPS.BR, por isso, esses elementos são preenchidos com um conteúdo genérico, apenas para manter a integridade da arquitetura de destino.

Questão 2) Comentários gerais:

O Quadro 5.5 apresenta os comentários gerais obtidos na avaliação para o processo DRE do MR-MPS-SW versão agosto/2012 na arquitetura CMMI.

Quadro 5.5 – Comentários gerais do processo DRE do MR-MPS-SW versão agosto/2012 na arquitetura CMMI.

Pontos fortes apresentados no resultado
- Nenhum.
Pontos fracos apresentados no resultado
- Ausência de elementos na arquitetura CMMI.
Sugestões de melhoria
- Apresentar um mapa de correspondência dos elementos do Modelo Origem X Modelo Destino. - Quando, no modelo de origem, não existir elementos existentes no modelo de destino, ao invés de utilizar "Generic Identification, Generic Name, Generic Definition, etc" seria interessante colocar uma frase indicando a inexistência do elemento no modelo de origem. - Como o modelo MPS.BR tem menos elementos que o Modelo CMMI, ao representá-lo na arquitetura do CMMI vários campos ficaram sem preencher, isto é, apresentando a descrição "generic ...". Poderia colocar N/A – Não se Aplica em vez de "generic ...".
De uma forma geral o resultado apresentado foi satisfatório?
- Atende em grande parte.

As sugestões de melhoria serão analisadas e caso adequadas, serão implementadas em uma nova versão do software PRO2PI-MMCA.

5.4.6 Processo "DRE – Desenvolvimento de Requisitos" do modelo MR-MPS-SW versão agosto/2012 expresso na arquitetura SPICE

Três especialistas avaliaram o resultado do PRO2PI-MMCA: Processo "DRE – Desenvolvimento de Requisitos" do modelo MR-MPS-SW versão agosto/2012 expresso na arquitetura SPICE. São apresentadas a avaliação e a análise do resultado.

Questão 1) O conteúdo do processo DRE do MR-MPS-SW versão agosto/2012 foi expresso corretamente do ponto de vista lógico na arquitetura SPICE? A Figura 5.11 apresenta as respostas para a questão 1. Um especialista considera que contempla parcialmente e dois consideram que contempla totalmente.

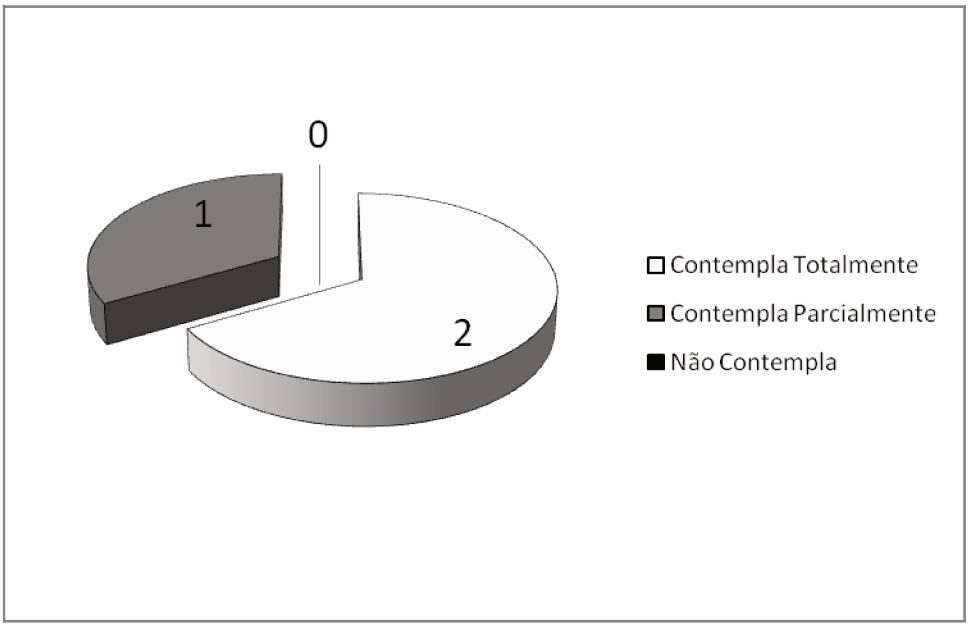


Figura 5.11 – Avaliação do processo DRE do MR-MPS-SW versão agosto/2012 na arquitetura SPICE.

A Figura 5.12 apresenta a classificação da justificativa da avaliação do resultado como contempla parcialmente.

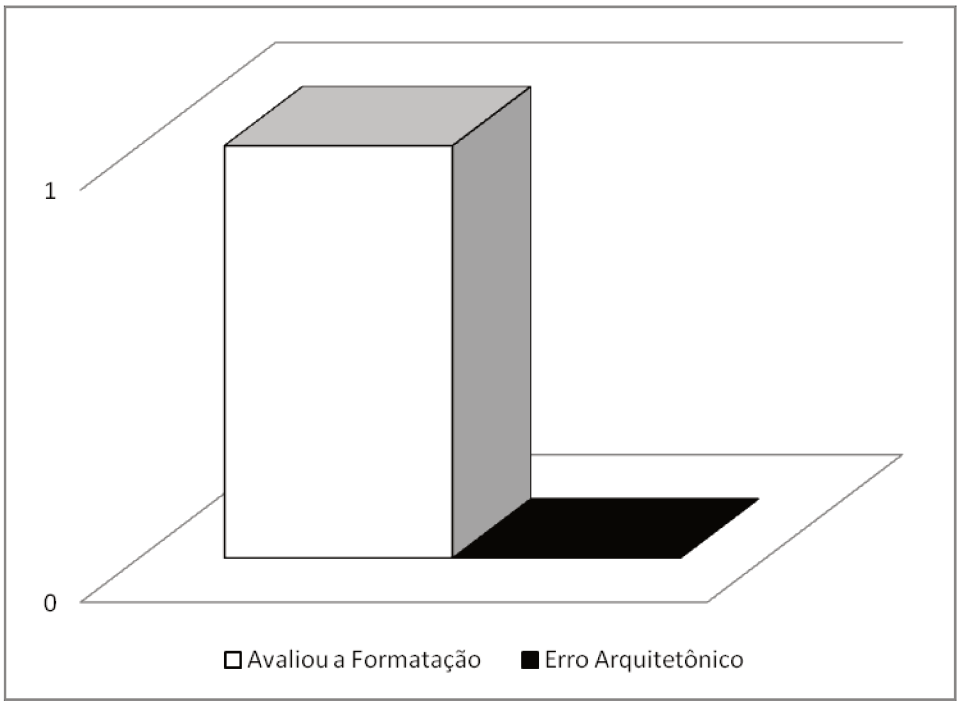


Figura 5.12 – Classificação da justificativa da avaliação do processo DRE do MR-MPS-SW versão agosto/2012 na arquitetura SPICE.

Um especialista avaliou que contempla parcialmente, pois entendeu que ao realizar a transformação de modelos, que os elementos *Process Outcome* e *Work Product* da arquitetura SPICE, que são importantes para o entendimento, não foram preenchidos com um conteúdo representativo.

Os elementos *Process Outcome* e *Work Product* da arquitetura SPICE não possuem elementos correspondentes diretos na arquitetura MPS.BR, por isso, esses elementos são preenchidos com um conteúdo genérico, apenas para manter a integridade da arquitetura de destino, o que não afeta a representação da arquitetura e o seu uso para implementação, uma vez que o elemento requerido *Base Practice* da arquitetura SPICE possui o elemento correspondente *Process Outcome* na arquitetura MPS.BR.

Questão 2) Comentários gerais:

O Quadro 5.6 apresenta os comentários gerais obtidos na avaliação para o processo DRE do MR-MPS-SW versão agosto/2012 na arquitetura SPICE.

Quadro 5.6 – Comentários gerais do processo DRE do MR-MPS-SW versão agosto/2012 na arquitetura SPICE.

Pontos fortes apresentados no resultado
- Nenhum.
Pontos fracos apresentados no resultado
- Ausência de elementos na arquitetura SPICE.
Sugestões de melhoria
- Apresentar um mapa de correspondência dos elementos do Modelo Origem X Modelo Destino. - Quando, no modelo de origem, não existir elementos existentes no modelo de destino, ao invés de utilizar "Generic Identification, Generic Name, Generic Definition, etc" seria interessante colocar uma frase indicando a inexistência do elemento no modelo de origem.
De uma forma geral o resultado apresentado foi satisfatório?
- Sim.

As sugestões de melhoria serão analisadas e caso adequadas, serão implementadas em uma nova versão do software PRO2PI-MMCA.

5.4.7 Processo "ENG.1 – *Requirements elicitation*" do modelo ASPICE versão 2.5 expresso na arquitetura CMMI

Três especialistas avaliaram o resultado do PRO2PI-MMCA: Processo "ENG.1 – *Requirements elicitation*" do modelo ASPICE versão 2.5 expresso na arquitetura CMMI. São apresentadas a avaliação e a análise do resultado.

Questão 1) O conteúdo do processo ENG.1 do ASPICE versão 2.5 foi expresso corretamente do ponto de vista lógico na arquitetura CMMI? A Figura 5.13 apresenta as respostas para a questão 1. Um especialista considera que contempla parcialmente e dois consideram que contempla totalmente.

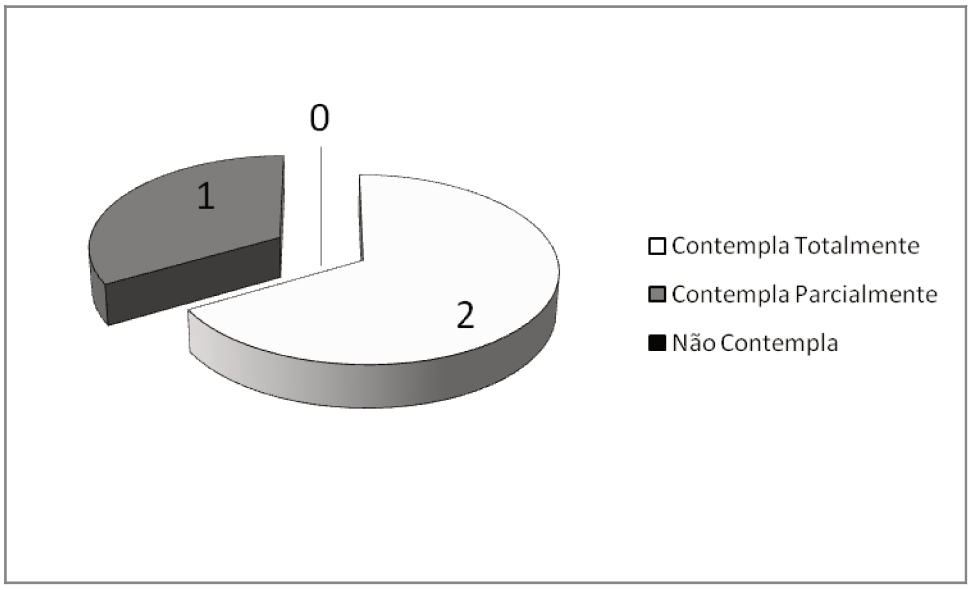


Figura 5.13 – Avaliação do processo ENG.1 do ASPICE versão 2.5 na arquitetura CMMI.

A Figura 5.14 apresenta a classificação da justificativa da avaliação do resultado como contempla parcialmente.

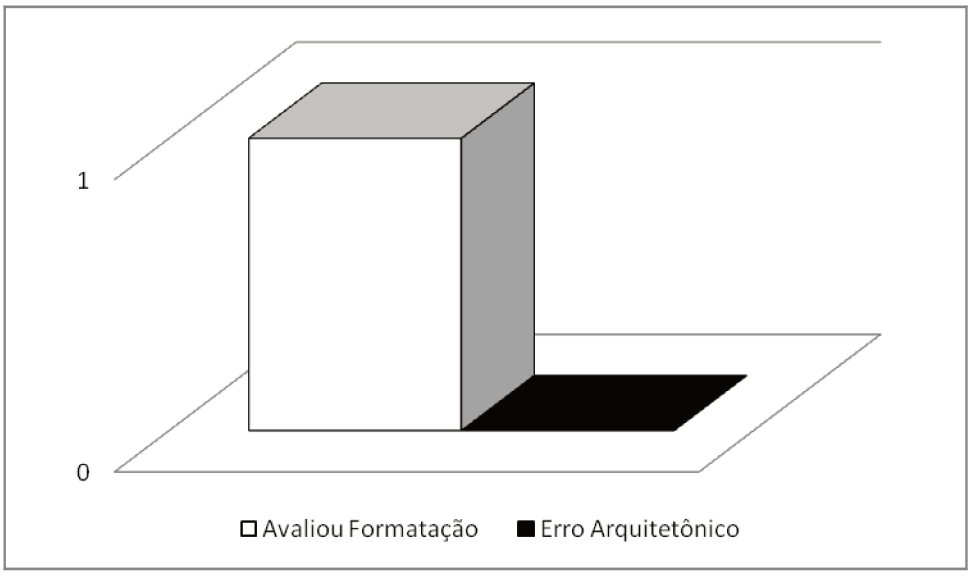


Figura 5.14 – Classificação da justificativa da avaliação do processo ENG.1 do ASPICE versão 2.5 na arquitetura CMMI.

Um especialista avaliou que contempla parcialmente, pois entendeu que ao realizar a transformação de modelos, que o elemento *Process Outcome* da arquitetura SPICE não teve um elemento correspondente na arquitetura CMMI. Porém, o elemento *Process Outcome* da arquitetura SPICE corresponde ao elemento *Specific Goal* da arquitetura CMMI. E a transformação faz uso dessa correspondência.

Questão 2) Comentários gerais:

O Quadro 5.7 apresenta os comentários gerais obtidos na avaliação para o processo ENG.1 do ASPICE versão 2.5 na arquitetura CMMI.

Quadro 5.7 – Comentários gerais do processo ENG.1 do ASPICE versão 2.5 na arquitetura CMMI.

Pontos fortes apresentados no resultado
- Nenhum.
Pontos fracos apresentados no resultado
- Nenhum.
Sugestões de melhoria
- Nenhuma.
De uma forma geral o resultado apresentado foi satisfatório?
- Sim.

5.4.8 Processo "ENG.1 – *Requirements elicitation*" do modelo ASPICE versão 2.5 expresso na arquitetura MPS.BR

Dois especialistas avaliaram o resultado do PRO2PI-MMCA: Processo "ENG.1 – *Requirements elicitation*" do modelo ASPICE versão 2.5 expresso na arquitetura MPS.BR. São apresentadas a avaliação e a análise do resultado.

Questão 1) O conteúdo do processo ENG.1 do ASPICE versão 2.5 foi expresso corretamente do ponto de vista lógico na arquitetura MPS.BR? A Figura 5.15 apresenta as respostas para a questão 1. Um especialista considera que contempla parcialmente e um considera que contempla totalmente.

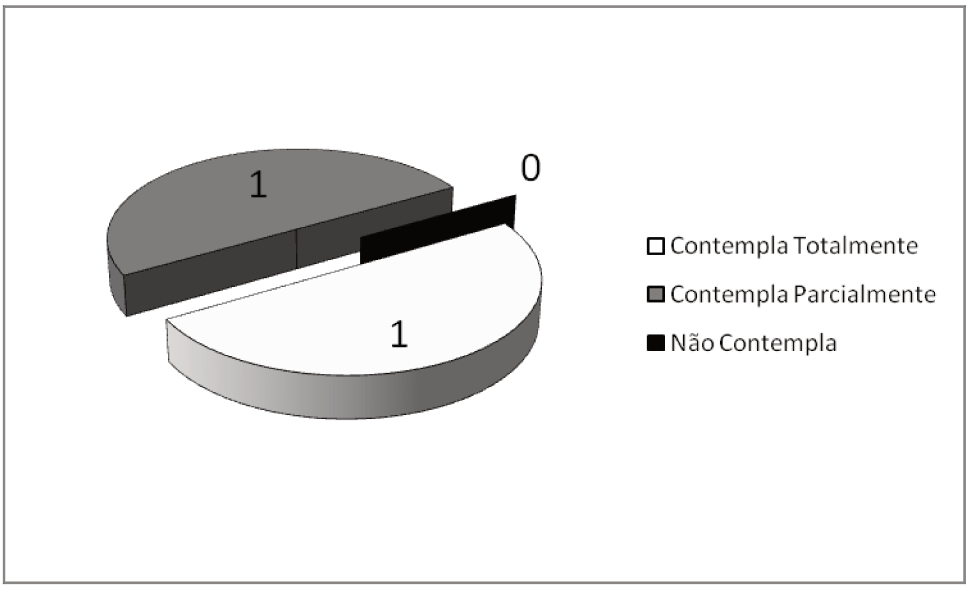


Figura 5.15 – Avaliação do processo ENG.1 do ASPICE versão 2.5 na arquitetura MPS.BR.

A Figura 5.16 apresenta a classificação da justificativa da avaliação do resultado como contempla parcialmente.

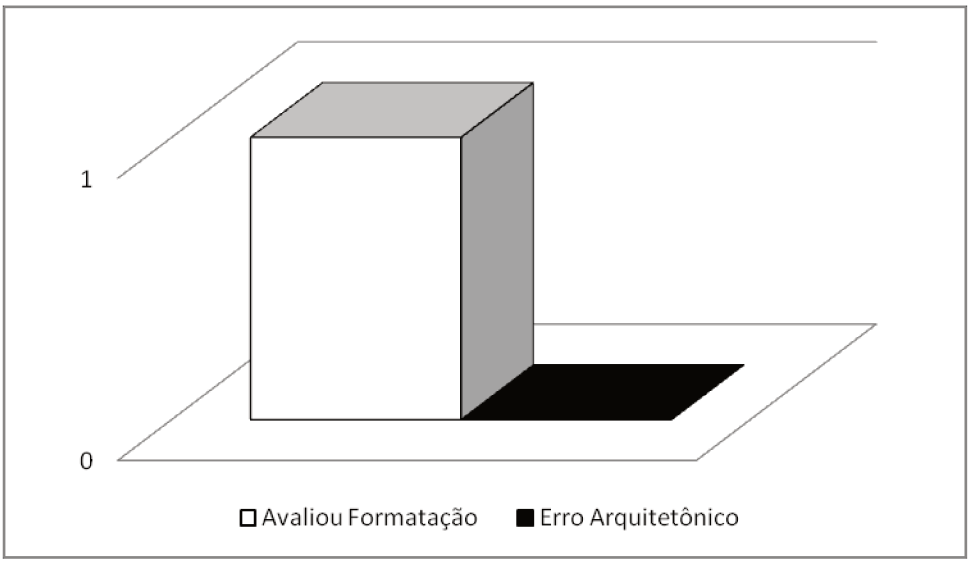


Figura 5.16 – Classificação da justificativa da avaliação do processo ENG.1 do ASPICE versão 2.5 na arquitetura MPS.BR.

Um especialista avaliou que contempla parcialmente, pois entendeu que ao realizar a transformação de modelos, que os elementos *Process Outcome* e *Work Product* da arquitetura SPICE não são contemplados na arquitetura MPS.BR. Os elementos *Process Outcome* e *Work Product* da arquitetura SPICE não possuem elementos correspondentes diretos na arquitetura MPS.BR, por isso, esses elementos não aparecem na arquitetura MPS.BR, o que não afeta a representação da arquitetura e o seu uso para implementação, uma vez que o elemento requerido *Base Practice* da arquitetura SPICE possui o elemento correspondente *Process Outcome* na arquitetura MPS.BR.

Questão 2) Comentários gerais:

O Quadro 5.8 apresenta os comentários gerais obtidos na avaliação para o processo ENG.1 do ASPICE versão 2.5 na arquitetura MPS.BR.

Quadro 5.8 – Comentários gerais do processo ENG.1 do ASPICE versão 2.5 na arquitetura MPS.BR.

Pontos fortes apresentados no resultado
- Nenhum.
Pontos fracos apresentados no resultado
- Nenhum.
Sugestões de melhoria
- Trocar o nome do elemento “process outcome” para “resultado esperado” na arquitetura MPS.BR.
De uma forma geral o resultado apresentado foi satisfatório?
- Sim.

A sugestão de melhoria será analisada e caso adequada, será implementada em uma nova versão do software PRO2PI-MMCA.

5.5 Ameaças à validade da avaliação

Foram identificadas três ameaças à validade da avaliação realizada. Estas ameaças estão relacionadas ao pequeno número de especialistas, às limitações da formatação dos perfis gerados e ao exercício com apenas três modelos. A avaliação do metamodelo foi realizada com um pequeno grupo de especialistas em modelos. Embora esta avaliação tenha sido suficiente para o objetivo do trabalho, é importante uma validação com um grupo maior de especialistas. As avaliações dos resultados demonstraram que é necessário criar um metamodelo para tratar a formatação dos perfis de capacidade de processo gerados. Ao realizar as transformações, nos casos em que a arquitetura de origem possui mais elementos em relação à arquitetura de destino, algumas informações são perdidas, pois nem todos os elementos da arquitetura de origem são mapeados e consequentemente transformados para os elementos da arquitetura de destino.

Uma limitação adicional à avaliação do trabalho é que o metamodelo foi exercitado com apenas alguns elementos de três modelos de capacidade. Embora estes três modelos sejam representativos e suficientes para o objetivo deste trabalho, é importante exercitar o metamodelo com outros elementos destes modelos e também com elementos de outros modelos. Além disso, o resultado deste trabalho está relacionado com a segunda fase de uma estratégia de três fases.

5.6 Discussão

A maior parte dos seis resultados foram avaliados como “Contempla Totalmente”. Nas avaliações em que os resultados foram avaliados como “Contempla Parcialmente” ou em um único caso em que um resultado foi avaliado como “Não Contempla”, o não atendimento do perfil de capacidade gerado, não se deu por erro arquitetônico. O não atendimento total do perfil de capacidade estava sempre relacionado a questões de formatação, ou seja, disposição dos elementos, não concordância da nomenclatura utilizada e apresentação de atributos do metamodelo.

Os pontos fracos e as sugestões de melhoria apontados, também indicam pontos relacionados à formatação. A ideia é analisar todos os pontos fracos e as sugestões de melhorias e caso sejam julgadas adequadas, implementar em uma próxima versão do software PRO2PI-MMCA.

Ficou evidente a necessidade da criação de um metamodelo para tratar a questão da apresentação (formatação) dos perfis de capacidade de processo no formato original da arquitetura.

Como ponto forte do trabalho, os especialistas apontaram o correto mapeamento entre os elementos das arquiteturas CMMI, SPICE e MPS.BR. Foi possível observar que os elementos principais dos modelos de capacidade, utilizados para implementar os processos sempre estavam presentes nos perfis de capacidade gerados. Em contra partida, alguns elementos informativos em algumas transformações foram perdidos, como por exemplo, os produtos de trabalho do CMMI-DEV não aparecem na arquitetura MPS.BR. Isto não comprometeu o resultado do trabalho.

A avaliação e análise dos seis Perfis de Capacidade de Processo gerados pelo PRO2PI-MMCA foi satisfatória, pois mostrou que o Metamodelo de Perfis de Capacidade de Processo implementado foi capaz de representar e transformar elementos de Múltiplos Modelos de Capacidade de Processo de Software, mantendo os elementos necessários para implementar os processos de capacidade para alcançar a Melhoria de Processo de Software em organizações intensivas em software. Esta transformação é uma base para a integração de elementos de múltiplos modelos. A apresentação dos elementos com uma mesma arquitetura viabiliza o seu entendimento.

As três ameaças identificadas à validade da avaliação realizada estão controladas para o objetivo deste trabalho que é mostrar a viabilidade do uso do metamodelo. O resultado ainda não apoia completamente a integração de elementos de múltiplos modelos, que deve ser tratada em trabalhos futuros, mas sim a representação e transformação destes elementos, o que é suficiente para mostrar a viabilidade do uso do metamodelo.

"A melhor maneira de prever o futuro é criá-lo."

(Peter Drucker)

6 Conclusões

As abordagens existentes para múltiplos modelos tendem a limitar a utilização plena dos mesmos, pois estão baseadas no paradigma da atual Melhoria de Processo de Software. Desta forma, é oportuno utilizar um novo paradigma para lidar com múltiplos modelos em melhoria de processo software. O uso de um Metamodelo de Perfis de Capacidade de Processo baseado no paradigma da Engenharia Dirigida por Modelos apresentou-se como uma nova forma de se trabalhar com múltiplos modelos.

O desenvolvimento de um software como prova de conceito, foi fundamental para validar o metamodelo proposto. Este software permitiu gerar os resultados que foram avaliados pelos especialistas em modelos. O software também serviu de laboratório para testar e exercitar o metamodelo, revelando as suas limitações. A avaliação dos especialistas apontou o correto mapeamento entre os elementos arquitetônicos dos modelos envolvidos no trabalho. Os especialistas em modelos também indicaram melhorias que podem ser feitas no software.

Este capítulo apresenta as conclusões da dissertação por meio de uma análise da realização das etapas do projeto, resultados adicionais, trabalhos futuros e conclusão.

6.1 Análise das etapas do trabalho

O trabalho foi orientado por um objetivo que foi desdobrado em uma cadeia de seis etapas. A Etapa 1 foi estudar a metodologia PRO2PI e suas referências conceituais. Esta etapa foi importante para a realização do trabalho, pois o trabalho tratou da evolução de um dos componentes metodológicos da metodologia PRO2PI. A conclusão do trabalho com a realização das demais etapas é a evidência do cumprimento desta etapa, pois não seria possível realizar o trabalho sem este entendimento.

A Etapa 2 foi identificar novas referências conceituais para a realização do trabalho e fazer uma revisão sistemática da literatura para entender como desenvolver e utilizar metamodelos

como um meio na resolução de problemas no paradigma da MDE. Esta etapa foi importante para atualizar as referências conceituais e confirmar uma visão sobre o desenvolvimento de metamodelos. As principais referências identificadas foram relacionadas à transformação de modelos além de novas referências sobre MDE em geral. A revisão sistemática da literatura foi realizada e confirmou a visão de que os ambientes para MDE ainda não estão maduros o suficiente. Desta forma, optou-se por implementar o metamodelo em paradigmas convencionais de programação com o cuidado de simular a implementação em MDE.

A Etapa 3 foi evoluir o componente metodológico Geraes do PRO2PI de sua versão 2008, que representava os conceitos básicos unificados de perfis e de modelos de capacidade de processo, para um Metamodelo de Perfis de Capacidade de Processo. Esta etapa foi importante porque a versão 2008 tratou apenas dos conceitos. Estes conceitos formam o elemento principal do metamodelo, mas não são suficientes. Os conceitos foram revistos e foram complementados com a inclusão de dois componentes: um relacionado com as arquiteturas de modelos e perfis de capacidade e outro relacionado com as regras adicionais de transformação entre modelos. Este três componentes definem então o metamodelo para representar e transformar perfis de capacidade de processo.

A Etapa 4 foi desenvolver uma ferramenta (software), como prova de conceito, para implementar o Metamodelo de Perfis de Capacidade de Processo com o propósito de representar múltiplos modelos nos termos do metamodelo e para transformar elementos para perfis de capacidade de processo, a partir da integração dos elementos destes múltiplos modelos. Esta etapa foi importante para validar o metamodelo definido. A implementação foi realizada e o software foi exercitado.

A Etapa 5 foi aplicar a ferramenta de geração de perfis de capacidade de processo para melhoria de processo de software, após povoar o banco de dados do software com partes de modelos representativos. Esta etapa foi importante para exercitar a implementação e consequentemente validar a prova de conceito. Foram incluídos no software exemplos de arquiteturas relevantes, regras adicionais de transformação entres as arquiteturas, e elementos de três modelos relevantes de cada arquitetura. Realizou-se também transformações de elementos dos modelos entre as arquiteturas.

A Etapa 6 foi analisar os resultados obtidos e consolidar a documentação do projeto de pesquisa. Esta etapa completa a cadeia dessas seis etapas. O resultado das utilizações da imple-

mentação foram analisados e validados por um conjunto de especialistas seguindo uma adaptação da técnica de painel de especialistas. Os resultados desta análise e validação foram satisfatórios.

Com a realização desta cadeia de seis etapas, o objetivo foi considerado como atendido.

6.2 Resultados adicionais do trabalho

Em complementação à realização das etapas do trabalho, resultados adicionais podem ser destacados. Entre eles estão a disponibilização de um software do metamodelo, a definição das regras de transformação entre as arquiteturas CMMI, SPICE e MPS.BR, além da comparação com outras abordagens.

O software desenvolvido, apesar de ser um protótipo com o objetivo de ser uma prova de conceito, pode ser utilizado para ajudar no entendimento de elementos de múltiplos modelos e para transformar estes elementos com diferentes arquiteturas para uma única arquitetura. Geralmente esta única arquitetura é uma arquitetura com a qual uma determinada equipe tem mais conhecimento, mais experiência e mais ativos já desenvolvidos.

O PRO2PI-MMC descrito neste trabalho permite uma frequência de atualização alta, pois, como uma arquitetura é modelada no sistema, as alterações de uma arquitetura, a inclusão de um novo modelo ou de um novo elemento de um modelo pode ser realizada a qualquer momento de um ciclo de melhoria.

Os princípios básicos dos modelos contidos no metamodelo ajudaram a integrar os elementos de múltiplos modelos, respeitando as características inerentes a cada modelo. O uso de um metamodelo como um modelo de uma linguagem de modelos, promoveu a utilização mais ampla de múltiplos modelos.

6.3 Trabalhos futuros

Conforme descrito na Seção 3.1 Estratégia de Desenvolvimento do Trabalho, esta dissertação está relacionada com a segunda fase de uma estratégia de três fases para a pesquisa e desenvolvimento de um Metamodelo de Perfis de Capacidade de Processo como um componente metodológico da metodologia PRO2PI. A primeira fase foi a pesquisa e desenvolvimento dos

conceitos básicos. A segunda fase foi a especificação e implementação de um metamodelo da arquitetura de perfis e modelos de capacidade. A terceira fase será uma evolução do metamodelo para incluir o conteúdo dos perfis e modelos de capacidade de processo e então permitir a integração dos elementos.

As sugestões de trabalhos futuros estão organizadas nos termos da conclusão da segunda fase e da realização da terceira fase. Em relação à segunda fase as principais sugestões de trabalhos futuros são:

- melhorar a implementação do metamodelo para apresentar os modelos na representação gráfica original pois na versão atual do metamodelo a apresentação de cada modelo é feita com os elementos da arquitetura, mas não na representação gráfica original; e
- exercitar o metamodelo com outros modelos e outras arquiteturas.

Em relação à terceira fase as principais sugestões de trabalhos futuros são:

- realizar esta fase para completar o metamodelo; e
- exercitar o metamodelo em projetos de melhoria de processo com múltiplos modelos.

Na terceira fase a evolução do metamodelo sugere a inclusão de um metamodelo de tipo ontológico, pois o objetivo é modelar o conteúdo dos modelos, visto que o metamodelo desenvolvido nesta segunda fase é um metamodelo do tipo linguístico, que trata da modelagem da arquitetura dos modelos. Porém, conforme reconhecido por vários autores (Atkinson and Kühne 2002), (Atkinson and Kühne 2003), (Atkinson and Kühne 2005) e (Henderson-Sellers 2007), um metamodelo deve ser apenas de um tipo. A sugestão para este desafio é manter o metamodelo como do tipo linguístico com a modelagem do conteúdo dos modelos por meio da modelagem do relacionamento entre elementos dos modelos. Neste caso, por exemplo, se um determinado elemento de uma área de processo de um modelo for considerado por um especialista como equivalente a outro elemento de outra área de processo de outro modelo, este relacionamento será modelado e utilizado para uma integração destas duas áreas de processos.

Outro trabalho futuro é o estudo dos resultados de uma pesquisa de doutorado defendida no final de janeiro de 2013 para verificar oportunidades de melhoria (Kelemen 2013). Esta tese de doutorado cita vários artigos sobre a metodologia PRO2PI.

6.4 Conclusão

O “problema” da utilização de múltiplos modelos foi “criado” pelo sucesso da MPS baseado em um modelo previamente definido. As abordagens atuais para múltiplos modelos buscam, em diferentes formas, transformar um conjunto definido de modelos em um novo modelo previamente definido, sempre baseado no conhecimento de especialistas sem um apoio metodológico e sem utilização de informações modeladas sobre os modelos. Desta forma tentam “resolver o problema no mesmo nível de abstração” com a atual MPS.

O metamodelo apresentado nesta dissertação é a base para apoiar a evolução da MPS com a metodologia PRO2PI. Esta metodologia trata a utilização integrada de elementos de múltiplos modelos em outro nível de abstração, seguindo a orientação de Albert Einstein: “Os problemas significativos que enfrentamos não podem ser resolvidos no mesmo nível de pensamento em que estávamos quando os criamos”. Este outro nível de abstração é atingido com a utilização de metamodelos.

Referências

Armitage, James W. 1993. “Process Guide for the DSSA - Process Life Cycle.” Special Report - CMU/SEI-93-SR-21, Software Engineering Institute, Carnegie Mellon University, Pittsburgh.

Armour, Philip G. 2004. *The Laws of Software Process - A New Model for the Production and Management of Software*. 1st ed. Washington: AUERBACH.

Atkinson, Colin, Matthias Gutheil, and Bastian Kennel. 2009. “A Flexible Infrastructure for Multilevel Language Engineering.” *IEEE TRANSACTIONS ON SOFTWARE ENGINEERING*, December, pp. 742-755.

Atkinson, Colin and Thomas Kühne. 2002. “Rearchitecting the UML Infrastructure.” *ACM Transactions on Modeling and Computer Simulation* 12(4):290-321.

Atkinson, Colin and Thomas Kühne. 2003. “Model-Driven Development: A Metamodeling Foundation.” *IEEE Computer Society*, Sep-Oct, pp. 36-41.

Atkinson, Colin and Thomas Kühne. 2005. “Concepts for Comparing Modeling Tool Architectures.” *Proceedings of the 8th International Conference MoDELS/UML*, pp. 398-413.

Banhesse, Edgar L., Clenio F. Salviano, and Mario Jino. 2012a. “Integrando Elementos de Múltiplos Modelos com um Metamodelo de Perfis de Capacidade de Processo.” *XI Simpósio Brasileiro de Qualidade de Software - SBQS*, Junho 11-15, pp. 128-142.

Beecham, Sarah, Tracy Hall, Carol Britton, Michaela Cottee, and Austen Rainer. 2005. “Using an Expert Panel to Validate a Requirements Process Improvement Model.” *The Journal of Systems and Software* 76(3):251-275.

Bézivin, Jean. 2003. “MDA™: From Hype to Hope, and Reality.” ATLAS Group (INRIA & LINA), University of Nantes, San Francisco.

Bézivin, Jean. 2005. “On the Unification Power of Models.” University of Nantes, Nantes, France.

Bézivin, Jean. 2006. “Model Driven Engineering: An Emerging Technical Space.” *Generative and Transformational Techniques in Software Engineering* 4143:36-64.

Bézivin, Jean and Olivier Gerbé. 2001. “Towards a Precise Definition of the OMG/MDA Framework.” *Proceedings 16th Annual International Conference on Automated Software Engineering*, November 26-29, pp. 273-280.

Brereton, Pearl, Barbara A. Kitchenham, David Budgen, Mark Turner, and Mohamed Khalil. 2006. “Lessons from applying the systematic literature review process within the software engineering domain.” *JSS - The Journal of Systems and Software*, July 10, pp. 571-583.

Card, D. N. 2004. “Research Directions in Software Process Improvement.” *28th IEEE International Computer Software and Applications Conference*, September 27-30, pp. 238-239.

Chen, Feng, Hongji Yang, Bing Qiao, and William C.-C. Chu. 2006. “A Formal Model Driven Approach to Dependable Software Evolution.” *Proceedings of the 30th Annual International Computer Software and Applications Conference (COMPSAC'06)*.

Cruz, António M. R. d. S. R. d. 2010. “Automatic Generation of User Interfaces from Rigorous Domain and Use Case Models.” Tese de Doutorado, Departamento de Engenharia Informática, Faculdade de Engenharia da Universidade do Porto, Porto.

Cruz, António M. R. d. and João P. Faria. 2008. “Automatic Generation of Interactive Prototypes for Domain Model Validation.” *ICSOFT*.

Cruz, António M. R. d. and João P. Faria. 2009. “Automatic Generation of User Interface Models and Prototypes from Domain and Use Case Models.” *ICSOFT*, May, pp. 35-60.

Cruz, António M. R. d. and João P. Faria. 2010. “A Metamodel-based Approach For Automatic User Interface Generation.” *MODELS - ACM-IEEE 13th International Conference on Model Driven Engineering Languages and Systems: Part I*, October 3-8, pp. 256-270.

Enterprise SPICE. 2010. “Enterprise SPICE - An Integrated Model for Enterprise-wide - Assessment and Improvement.” Technical Report - Issue 1, The Enterprise SPICE Project Team, SPICE.

Favre, Jean-Marie. 2004a. “Foundations of Model (Driven) (Reverse) Engineering: Models - Episode I: Stories of The Fidus Papyrus and of The Solarus.” Université Joseph Fourier, Grenoble, France.

Favre, Jean-Marie. 2004b. “CacOphoNy: Metamodel-Driven Software Architecture Reconstruction.” *Proceedings of the 11th Working Conference on Reverse Engineering (WCRE’04)*.

Favre, Jean-Marie. 2005. “Foundations of Meta-Pyramids: Languages vs. Metamodels - Episode II: Story of Thotus the Baboon.” University of Grenoble, Grenoble, France.

Femmer, Henning, Nora Broy, Marin Zec, Asa MacWilliams, and Roland Eckl. 2011. “Dynamic Software Visualization with BusyBorg - A Proof Of Concept.” *35th IEEE Annual Computer Software and Applications Conference*, pp. 492-497.

Gamma, E., R. Helm, R. Johnson, and J. Vlissides. 1994. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley.

Golding, David. 2008. *Beginning CakePHP From Novice to Professional*. Berkeley, California: Apress.

Gutierrez, J.J., M.J. Escalona, M. Mejías, I. Ramos, and J. Torres. 2009. “An approach for Model-Driven test generation.” *Third International Conference on Research Challenges in Information Science*, April 22-24, pp. 303-312.

Hauck, Jean C. R. 2011. “Um Método de Aquisição de Conhecimento para Customização de Modelos de Capacidade/Maturidade de Processos de Software.” Universidade Federal de Santa Catarina - Departamento de Engenharia do Conhecimento, Florianópolis.

Henderson-Sellers, Brian. 2007. “On the Challenges of Correctly Using Metamodels in Software Engineering.” *New Trends in Software Methodologies, Tools and Techniques - Proceedings of the Sixth SoMeT* 161:3-35.

Henderson-Sellers, Brian. 2010. “Bridging metamodels and ontologies in software engineering.” *The Journal of Systems and Software*, October, pp. 301-313.

Hyder, E. B., K. M. Heston, and M. C. Paulk. 2004. “The eSourcing Capability Model for Service Providers (eSCM-SP) Part 1: Model Overview.” ISRI - Institute for Software Research International, School of Computer Science, Carnegie Mellon University, Pittsburgh, Pennsylvania, USA.

ISO/IEC. 2003. “ISO/IEC 15504-2: Information Technology - Process Assessment - Part 2 - Performing an Assessment.” ISO, Geneve. International Organization for Standardization/International Electrotechnical Comission.

ISO/IEC. 2005. “ISO/IEC 15504-5: Information Technology - Process Assessment - Part 5 - An exemplar Process Assessment Model.” ISO, Geneve. International Organization for Standardization/International Electrotechnical Comission.

ISO/IEC. 2006. “ISO/IEC 15504-5: Information Technology - Process Assessment - Part 5 - An exemplar Process Assessment Model.” ISO, Geneve. International Organization for Standardization/International Electrotechnical Commission.

ISO/IEC. 2008. “ISO/IEC 12207 Systems and software engineering – Software life cycle processes.” ISO, Geneve. International Organization for Standardization/International Electrotechnical Comission.

Jouault, Fréedéric and Jean Bézivin. 2006. “KM3: a DSL for Metamodel Specification.” *Springer - In proc. of 8th FMOODS, LNCS 4037*, pp. 171-185.

Kelemen, Zádor D. 2013. “Process Based Unification for Multi-model Software Process Improvement.” Doctorate Thesis, Eindhoven University, Eindhoven.

Kitchenham, Barbara A. 2007. “Guidelines for performing Systematic Literature Reviews in Software Engineering.” Keele University and University of Durham, Keele and Durham.

Kleppe, Anneke G., Jos Warmer, and Wim Bast. 2003. *MDA explained: the model driven architecture - practice and promise*. 1st ed. Boston: Pearson Education.

Kurtev, Ivan, Jean Bézivin, Frédéric Jouault, and Patrick Valduriez. 2006. “Model-based DSL Frameworks.” *OOPSLA*, October 22-26, pp. 602-615.

Lavrakas, J. P. 2008. *Encyclopedia of Survey Research Methods*. 2nd ed. United States of America, California: SAGE.

Mendes, Fabiana F. 2010. “Melhoria de Processos de Tecnologia da Informação Multi-Modelo.” Universidade Federal de Goiás - Instituto de Informática, Goiânia.

Mens, Tom, Krzysztof Czarnecki, and Pieter V. Gorp. 2005. “A Taxonomy of Model Transformations.” *Dagstuhl Seminar Proceedings*, pp. 10.

Mohagheghi, Parastoo and Vegard Dehlen. 2008. “A Metamodel for Specifying Quality Models in Model-Driven Engineering.” *In Proceedings of the Nordic Workshop on Model Driven Engineering - Engineering Research Institute - University of Iceland*, August 20-22, pp. 51-65.

Mohagheghi, Parastoo, Vegard Dehlen, and Tor Neple. 2009. “Definitions and approaches to model quality in model-based software development - A review of literature.” *Information and Software Technology*, April 21, pp. 1646-1669.

Monperrus, Martin, Antoine Beugnard, and J. Champeau. 2009. “A Definition of "Abstraction Level" for Metamodels.” *16th Annual IEEE International Conference and Workshop on the Engineering of Computer Based Systems*, pp. 315-320.

Morana, Giovanni and Rao Mikkilineni. 2011. “Scaling and Self-repair of Linux Based Services Using a Novel Distributed Computing Model Exploiting Parallelism.” *20th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises*, pp. 98-103.

Oktaba, Hanna, Claudia A. Esquivel, A. S. Ramos, A. M. Martínez, G. Q. Osorio, Mara R. López, Francisco L. L. Hinojo, María E. R. López, María J. O. M., Yolanda F. Ordóñez, and Miguel Á. F. Lemus. 2005. “Modelo de Processos para a Indústria de Software - Por Níveis de Capacidade de Processo.” MoProSoft.

Passerone, Roberto, Imene B. Hafaiedh, Susanne Graf, Albert Benveniste, Daniela Cancila, Arnaud Cuccuru, Sébastien Gérard, Francois Terrier, Werner Damm, Alberto Ferrari, Leonardo Mangeruca, Bernhard Josko, Thomas Peikenkamp, and Alberto S. 2009. “Metamodels in Europe: Languages, Tools, and Applications.” *IEEE Design & Test of Computers*, June, pp. 38-53.

Petticrew, Mark and Helen Roberts. 2006. *Systematic Reviews in the Social Sciences - A PRACTICAL GUIDE*. Malden: Blackwell Publishing.

Pries-Heje, Jan and Jørn Johansen. 2010. “Software Process Improvement Manifesto.” *eurospi.net* (Version A.1.2.2010):1-17.

Rio, Américo and Fernando B. e. Abreu. 2010. “Websites Quality - Does It Depend on the Application Domain?” *Seventh International Conference on the Quality of Information and Communications Technology*, September 29, pp. 493-498.

Rocha, Ana R., Andrés Rubinstein, Ana L. Magalhães, Anne E. Katsurayama, Arley Duque, Carlos B. Palestino, Crhistian Souza, Cristina Cerdeiral, Leandro Teixeira, Nelson S. d. Paiva, and Leonardo Barros. 2009. "Avaliação Conjunta CMMI Nível 3 e MPS Nível C: Lições Aprendidas e Recomendações." *WAMPS - Workshop Anual do MPS*, Outubro 19-22, pp. 52-61.

Rossini, Alessandro, Adrian Rutle, Khalid A. Mughal, Yngve Lamo, and Uwe Wolter. 2011. "A Formal Approach to Data Validation Constraints in MDE." *In TTSS 2011: 5th International Workshop on Harnessing Theories for Tool Support in Software*, September, pp. 65-76.

Salviano, Clenio F. 2006. "Uma proposta orientada a perfis de capacidade de processo para evolução da melhoria de processo de software." Tese de Doutorado, FEEC, UNICAMP, Campinas.

Salviano, Clenio F. 2009. "A Multi-Model Process Improvement Methodology Driven by Capability Profiles." *COMPSAC - The 33rd Annual IEEE International Computer Software and Applications Conference*, July 20-24, pp. 636-637. In proceedings of COMPSAC 2009.

Salviano, Clenio F. 2011. "A Modeling View of Process Improvement." *SPICE - Software Process Improvement and Capability dEtermination*, May 31, pp. 1-12.

Salviano, C. F. and A. M. Figueiredo. 2008. "Unified Basic Concepts for Process Capability Models." *SEKE*, July 1-3, pp. 173-178.

Salviano, Clenio F., Márcia R. M. Martinez, Edgar L. Banhesse, Angela Enelize, Alessandra Zoucas, and Marcello Thiry. 2010. "A Method for Tridimensional Process Assessment using Modeling Theory." *IEEE - Computer Society - International Conference on the Quality of Information and Communications Technology*, pp. 430-435.

Sarathy, Vijay, Purnendu Narayan, and Rao Mikkilineni. 2010. "Next generation Cloud Computing Architecture Enabling real-time dynamism for shared distributed physical infrastructure." *Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises*, pp. 48-53.

SEI. 2006. "CMMI for Development, Version 1.2." Technical Report, Software Engineering Institute, Carnegie Mellon University, Bedford, Hanscom AFB.

SEI. 2007. "CMMI for Acquisition, Version 1.2." Technical Report, Software Engineering Institute, Carnegie Mellon University, Bedford, Hanscom AFB.

SEI. 2009. "CMMI for Services, Version 1.2." Technical Report, Software Engineering Institute, Carnegie Mellon University, Bedford, Hanscom AFB.

SEI. 2010. "CMMI for Development, Version 1.3." Technical Report, Software Engineering Institute, Carnegie Mellon University, Bedford, Hanscom AFB.

Seidewitz, Ed. 2003. "What Models Mean." *IEEE Computer Society*, pp. 26-32.

Shiliang, Wu and Zhong Qin. 2010. “A Meta-model for Developing Business-model Driven Management Information Systems.” *International Conference on Logistics Systems and Intelligent Management*, January 9-10, pp. 282-286.

Silva, Edna L. d. and Estera M. Menezes. 2005. “Metodologia da Pesquisa e Elaboração de Dissertação.” Universidade Federal de Santa Catarina - UFSC, Florianópolis.

Siviy, Jeannine M., M. L. Penn, and Robert W. Stoddard. 2008. *CMMI and Six Sigma*. Boston, MA: Addison-Wesley - Pearson Education.

SOFTEX. 2009. “MR-MPS - Modelo de Referência para Melhoria de Processo de Software - Guia Geral.” Relatório Técnico, SOFTEX - Associação para Promoção da Excelência do Software Brasileiro, Campinas.

SOFTEX. 2012a. “MR-MPS-SW - Modelo de Referência de Melhoria de Processo do Software Brasileiro para Software - Guia Geral.” Relatório Técnico, SOFTEX - Associação para Promoção da Excelência do Software Brasileiro, Campinas, SP. Associação para Promoção da Excelência do Software Brasileiro.

SOFTEX. 2012b. “MR-MPS-SV - Modelo de Referência de Melhoria de Processo do Software Brasileiro para Serviços - Guia Geral.” Relatório Técnico, SOFTEX - Associação para Promoção da Excelência do Software Brasileiro, Campinas, SP.

SOFTEX. 2012c. “Guia de Implementação - Parte 11: Implementação e Avaliação do MR-MPS-SW:2012 em Conjunto com o CMMI-DEV v1.3.” Relatório Técnico, SOFTEX - Associação para Promoção da Excelência do Software Brasileiro, Campinas.

SPICE. 2010. “Automotive SPICE Process Assessment Model, Version 2.5.” The Procurement Forum, SPICE, Brussels.

Thiry, Marcello, Alessandra Zoucas, and Leonardo Tristão. 2010. “Mapping Process Capability Models to Support Integrated Software Process Assessments.” *CLEI ELECTRONIC JOURNAL* 13:Paper 4.

Unterkalmsteiner, Michael, Tony Gorschek, A. K. M. M. Islam, Chow K. Cheng, Rahadian B. Permadi, and Robert Feldt. 2012. “Evaluation and Measurement of Software Process Improvement - A Systematic Literature Review.” *IEEE Transactions On Software Engineering*, pp. 1-29.

Wittle, Jon, John Hutchinson, and Mark Rouncefield. 2013. “The State of Practice in Model-Driven Engineering.” *IEEE Software*, April 23, pp. 1-13.

Zhang, Tian, Fréedéric Jouault, Jean Bézivin, and Jianhua Zhao. 2008. “A MDE Based Approach for Bridging Formal Models.” *2nd IFIP/IEEE International Symposium on Theoretical Aspects of Software Engineering*, pp. 113-116.

Apêndices

Apêndice A

Perfil de Capacidade de Processo de Múltiplos Modelos

Este apêndice apresenta um perfil de capacidade de processo de múltiplos modelos. O perfil foi gerado a partir da ferramenta PRO2PI-MMCA e é composto pela área de processo PI – *Product Integration* no nível de capacidade 3 – Definido do modelo CMMI-DEV versão 1.3, pelo processo ENG.3 – *System architectural design* no nível de capacidade 2 – Gerenciado do modelo ASPICE versão 2.5 e pelo processo DRE – Desenvolvimento de Requisitos no nível de capacidade 2 – Gerenciado do modelo MR-MPS-SW versão de agosto/2012. O perfil de capacidade foi transformado para a arquitetura CMMI.

Process Area PI – Product Integration
Definition: The purpose of Product Integration (PI) is to assemble the product from the product components, ensure that the product, as integrated, behaves properly (i.e., possesses the required functionality and quality attributes), and deliver the product.
Description: This process area addresses the integration of product components into more complex product components or into complete products. The scope of this process area is to achieve complete product integration through progressive assembly of product components, in one stage or in incremental stages, according to a defined integration strategy and Throughout the process areas, where the terms "product" and procedures. "product component" are used, their intended meanings also encompass services, service systems, and their components. (...)

This ***Process Area*** has the following ***Specific Goals***:

Specific Goal SG 1 – Prepare for Product Integration
Definition: Preparation for product integration is conducted.
Description: Preparing for the integration of product components involves establishing an integration strategy, establishing the environment for performing the integration, and establishing integration procedures and criteria. Preparation for integration starts early in the project.

This ***Specific Goal*** has the following ***Specific Practices***:

- ***Specific Practice*** SP 1.1 – Establish an Integration Strategy
  Definition: Establish and maintain a product integration strategy.
  Description: The product integration strategy describes the approach for receiving, assembling, and evaluating the product components that comprise the product. A product integration strategy addresses items such as the following:
  - Making product components available for integration (e.g., in what sequence)
  - Assembling and evaluating as a single build or as a progression of incremental builds
  - Including and testing features in each iteration when using iterative development
  - Managing interfaces
  - Using models, prototypes, and simulations to assist in evaluating an assembly, including its interfaces
  - Establishing the product integration environment
  - Defining procedures and criteria
  - Making available the appropriate test tools and equipment
  - Managing product hierarchy, architecture, and complexity

- Recording results of evaluations
- Handling exceptions (...)

This ***Specific Practice*** has the following ***Specific Subpractices***:

- ***Specific Subpractice*** 1 – Identify the product components to be integrated.
  Definition:
  Description:

- ***Specific Subpractice*** 2 – Identify the verifications to be performed during the integration of the product components.
  Definition:
  Description: This identification includes verifications to be performed on interfaces.

- ***Specific Subpractice*** 3 – Identify alternative product component integration strategies.
  Definition:
  Description: Developing an integration strategy can involve specifying and evaluating several alternative integration strategies or sequences.

- ***Specific Subpractice*** 4 – Select the best integration strategy.
  Definition:
  Description: The availability of the following will need to be aligned or harmonized with the integration strategy: product components; the integration environment; test tools and equipment; procedures and criteria; relevant stakeholders; and staff who possess the appropriate skills.

- ***Specific Subpractice*** 5 – Periodically review the product integration strategy and revise as needed.
  Definition:
  Description: Assess the product integration strategy to ensure that variations in production and delivery schedules have not had an adverse impact on the integration sequence or compromised the factors on which earlier decisions were made.

- ***Specific Subpractice*** 6 – Record the rationale for decisions made and deferred.
  Definition:
  Description:

  This ***Specific Practice*** has the following ***Typical Work Products***:

- ***Typical Work Product*** 1 – Product integration strategy
  Definition:
  Description:

- ***Typical Work Product*** 2 – Rationale for selecting or rejecting alternative product integration strategies
  Definition:
  Description:

- ***Specific Practice*** SP 1.2 – Establish the Product Integration Environment
  Definition: Establish and maintain the environment needed to support the integration of the product components.
  Description: The environment for product integration can either be acquired or developed. To establish an environment, requirements for the purchase or development of equipment, software, or other resources will need to be developed. These requirements are gathered when implementing the processes associated with the Requirements Development process area. (...)

  This ***Specific Practice*** has the following ***Specific Subpractices***:

  - ***Specific Subpractice*** 1 – Identify the requirements for the product integration environment.
    Definition:
    Description:

  - ***Specific Subpractice*** 2 – Identify verification procedures and criteria for the product integration

environment.
Definition:
Description:

- ***Specific Subpractice*** 3 – Decide whether to make or buy the needed product integration environment.
  Definition:
  Description: Refer to the Supplier Agreement Management process area for more information about managing the acquisition of products and services from suppliers.

- ***Specific Subpractice*** 4 – Develop an integration environment if a suitable environment cannot be acquired.
  Definition:
  Description: For unprecedented, complex projects, the product integration environment can be a major development. As such, it would involve project planning, requirements development, technical solutions, verification, validation, and risk management.

- ***Specific Subpractice*** 5 – Maintain the product integration environment throughout the project.
  Definition:
  Description:

- ***Specific Subpractice*** 6 – Dispose of those portions of the environment that are no longer useful.
  Definition:
  Description:

  This ***Specific Practice*** has the following ***Typical Work Products***:

- ***Typical Work Product*** 1 – Verified environment for product integration
  Definition:
  Description:

- ***Typical Work Product*** 2 – Support documentation for the product integration environment
  Definition:
  Description:

- ***Specific Practice*** SP 1.3 – Establish Product Integration Procedures and Criteria
  Definition: Establish and maintain procedures and criteria for integration of the product components.
  Description: Procedures for the integration of the product components can include such things as the number of incremental iterations to be performed and details of the expected tests and other evaluations to be carried out at each stage. Criteria can indicate the readiness of a product component for integration or its acceptability. (...)

Capability Level CL 3 – Capability Level 3 – Defined
Definition: A capability level 3 process is characterized as a defined process. A defined process is a managed process that is tailored from the organization's set of standard processes according to the organization's tailoring guidelines; has a maintained process description; and contributes process related experiences to the organizational process assets.
Description: A critical distinction between capability levels 2 and 3 is the scope of standards, process descriptions, and procedures. At capability level 2, the standards, process descriptions, and procedures can be quite different in each specific instance of the process (e.g., on a particular project). At capability level 3, the standards, process descriptions, and procedures for a project are tailored from the organization's set of standard processes to suit a particular project or organizational unit and therefore are more consistent, except for the differences allowed by the tailoring guidelines. Another critical distinction is that at capability level 3 processes are typically described more rigorously than at capability level 2. A defined process clearly states the purpose, inputs, entry criteria, activities, roles, measures, verification steps, outputs, and exit criteria. At capability level 3, processes are managed more proactively using an understanding of the interrelationships of the process activities and detailed measures of the process and its work products.

This ***Capability Level*** has the following ***Generic Goal***:

Generic Goal GG 3 – Institutionalize a Defined Process
Definition: The process is institutionalized as a defined process.
Description:

This ***Generic Goal*** has the following ***Generic Practices***:

- ***Generic Practice*** GP 3.1 – Establish a Defined Process
  Definition: Establish and maintain the description of a defined process.
  Description: The purpose of this generic practice is to establish and maintain a d from the organization's set of description of the process that is tailored standard processes to address the needs of a specific instantiation. The organization should have standard processes that cover the process area, as well as have guidelines for tailoring these standard processes to meet the needs of a project or organizational function. (...)

  This ***Generic Practice*** has the following ***Generic Subpractices***:

  - ***Generic Subpractice*** 1 – Select from the organization's set of standard processes those processes that cover the process area and best meet the needs of the project or organizational function.
    Definition:
    Description:

  - ***Generic Subpractice*** 2 – Establish the defined process by tailoring the selected processes according to the organization's tailoring guidelines.
    Definition:
    Description:

  - ***Generic Subpractice*** 3 – Ensure that the organization's process objectives are appropriately addressed in the defined process.
    Definition:
    Description:

  - ***Generic Subpractice*** 4 – Document the defined process and the records of the tailoring.
    Definition:
    Description:

  - ***Generic Subpractice*** 5 – Revise the description of the defined process as necessary.
    Definition:
    Description:

- ***Generic Practice*** GP 3.2 – Collect Process Related Experiences
  Definition: Collect process related experiences derived from planning and performing the process to support the future use and improvement of the organization's processes and process assets.
  Description: The purpose of this generic practice is to collect process related experiences, including information and artifacts derived from planning and performing the process. Examples of process related experiences include work products, measures, measurement results, lessons learned, and process improvement suggestions. The information and artifacts are collected so that they can be included in the organizational process assets and made available to those who are (or who will be) planning and performing the same or similar processes. The information and artifacts are stored in the organization's measurement repository and the organization's process asset library. Examples of relevant information include the effort expended for the various activities, defects injected or removed in a particular activity, and lessons learned. Refer to the Integrated Project Management process area for more information about contributing to organizational process assets. Refer to the Organizational Process Definition process area for more information about establishing organizational process assets.

  This ***Generic Practice*** has the following ***Generic Subpractices***:

  - ***Generic Subpractice*** 1 – Store process and product measures in the organization's measurement repository.
    Definition: The process and product measures are primarily those measures that are defined in the common set of measures for the organization's set of standard processes.
    Description:

  - ***Generic Subpractice*** 2 – Submit documentation for inclusion in the organization's process asset library.
    Definition:
    Description:

- ***Generic Subpractice*** 3 – Document lessons learned from the process for inclusion in the organization's process asset library.
  Definition:
  Description:

- ***Generic Subpractice*** 4 – Propose improvements to the organizational process assets.
  Definition:
  Description:

Process Area ENG.3 – System architectural design
Definition: The purpose of the System architectural design process is to identify which system requirements are to be allocated to which elements of the system.
Description:

This ***Process Area*** has the following ***Specific Goals***:

Specific Goal 1 – a system architectural design is defined that identifies the elements of the system and meets the defined systems requirements
Definition:
Description:

This ***Specific Goal*** has the following ***Specific Practices***:

- ***Specific Practice*** ENG.3.BP1 – Define system architectural design.
  Definition: Establish the system architectural design that identifies the elements of the system with respect to the functional and non-functional system requirements.
  Description: NOTE 1: The system might be decomposed into several subsystems on different system levels, if necessary.

  This ***Specific Practice*** has the following ***Specific Subpractice***:

  - ***Specific Subpractice*** Generic Identification – Generic Name
    Definition: Generic Definition
    Description: Generic Description

  This ***Specific Practice*** has the following ***Typical Work Products***:

  - ***Typical Work Product*** 04-06 – System architectural design
    Definition:
    Description:
    - Provides an overview of all system design
    - Describes the interrelationship between system elements
    - Describes the relationship between the system elements and the software
    - Specifies the design for each required system element, consideration is given to things like:
    -- memory/capacity requirements
    -- hardware interfaces requirements
    -- user interfaces requirements
    -- external system interface requirements
    -- performance requirements
    (...)

  - ***Typical Work Product*** 13-22 – Traceability record
    Definition:
    Description:
    - All requirements (customer and internal) are to be traced
    - Identifies a mapping of requirement to life cycle work products
    - Provides the linkage of requirements to work product decomposition (i.e., requirement design code test deliverables, etc.)
    - Provides forward and backwards mapping of requirements to associated work products throughout all phases of the life cycle
    - NOTE: this may be included as a function of another defined work product (example: A CASE tool for design decomposition may have a mapping ability as part of its features)

- ***Typical Work Product*** 17-50 – Verification criteria
  Definition:
  Description:
  - Each requirement is verifiable or can be assessed
  - Verification criteria define the qualitative and quantitative criteria for verification of a requirement.
  - Verification criteria demonstrate that a requirement can be verified within agreed constraints.
  (Additional Requirement to 17-00 Requirements specification)

- ***Specific Practice*** ENG.3.BP4 – Develop verification criteria.
  Definition: Define the verification criteria for each element of the system concerning the functional and non-functional system requirements based on the system architectural design.
  Description:

  This ***Specific Practice*** has the following ***Specific Subpractice***:

  - ***Specific Subpractice*** Generic Identification – Generic Name
    Definition: Generic Definition
    Description: Generic Description

    This ***Specific Practice*** has the following ***Typical Work Products***:

  - ***Typical Work Product*** 04-06 – System architectural design
    Definition:
    Description:
    - Provides an overview of all system design
    - Describes the interrelationship between system elements
    - Describes the relationship between the system elements and the software
    - Specifies the design for each required system element, consideration is given to things like:
    -- memory/capacity requirements
    -- hardware interfaces requirements
    (...)

  - ***Typical Work Product*** 13-22 – Traceability record
    Definition:
    Description:
    - All requirements (customer and internal) are to be traced
    - Identifies a mapping of requirement to life cycle work products
    - Provides the linkage of requirements to work product decomposition (i.e., requirement design code test deliverables, etc.)
    - Provides forward and backwards mapping of requirements to associated work products throughout all phases of the life cycle
    - NOTE: this may be included as a function of another defined work product (example: A CASE tool for design decomposition may have a mapping ability as part of its features)

  - ***Typical Work Product*** 17-50 – Verification criteria
    Definition:
    Description:
    - Each requirement is verifiable or can be assessed
    - Verification criteria define the qualitative and quantitative criteria for verification of a requirement.
    - Verification criteria demonstrate that a requirement can be verified within agreed constraints.
    (Additional Requirement to 17-00 Requirements specification)

Specific Goal 2 – the system requirements are allocated to the elements of the system
Definition:
Description:

This ***Specific Goal*** has the following ***Specific Practice***:

- ***Specific Practice*** ENG.3.BP2 – Allocate System Requirements.
  Definition: Allocate all system requirements to the elements of the system architectural design.
  Description:

  This ***Specific Practice*** has the following ***Specific Subpractice***:

  - ***Specific Subpractice*** Generic Identification – Generic Name
    Definition: Generic Definition
    Description: Generic Description

    This ***Specific Practice*** has the following ***Typical Work Product***:

  - ***Typical Work Product*** 04-06 – System architectural design
    Definition:
    Description:
    - Provides an overview of all system design
    - Describes the interrelationship between system elements
    - Describes the relationship between the system elements and the software
    - Specifies the design for each required system element, consideration is given to things like:
    -- memory/capacity requirements
    -- hardware interfaces requirements
    -- user interfaces requirements
    -- external system interface requirements
    -- performance requirements
    -- commands structures
    -- security/data protection characteristics
    -- system parameter settings
    -- manual operations
    -- reusable components
    - Mapping of requirements to system elements
    - Description of the operation modes of the system components (startup, shutdown, sleep mode, diagnosis mode, etc.)
    - Description of the dependencies among the system components regarding the operation modes
    - Description of the dynamic behaviour of the system and the system components
    (...)

Capability Level CL 2 – Capability Level 2 – Managed process
Definition: The previously described Performed process is now implemented in a managed fashion (planned, monitored and adjusted) and its work products are appropriately established, controlled and maintained.
Description:

This ***Capability Level*** has the following ***Generic Goals***:

Generic Goal PA 2.1 – Performance management attribute

Definition: The performance management attribute is a measure of the extent to which the performance of the process is managed.
Description:

This ***Generic Goal*** has the following ***Generic Practices***:

- ***Generic Practice*** GP 2.1.1 – Identify the objectives for the performance of the process.
  Definition:
  Description: NOTE 1: Performance objectives may include – (1) quality of the artifacts produced, (2) process cycle time or frequency (3) resource usage and (4) boundaries of the process. Performance objectives are identified based on process requirements. The scope of the process performance is defined. Assumptions and constraints are considered when identifying the performance objectives. NOTE 2: At minimum, project performance objectives for resources, effort and schedule should be stated.

  This ***Generic Practice*** has the following ***Generic Subpractices***:

  - ***Generic Subpractice*** a – objectives for the performance of the process are identified

Definition:
Description:

- ***Generic Subpractice*** b – performance of the process is planned and monitored
  Definition:
  Description:

- ***Generic Subpractice*** c – performance of the process is adjusted to meet plans
  Definition:
  Description:

- ***Generic Subpractice*** d – responsibilities and authorities for performing the process are defined, assigned and communicated
  Definition:
  Description:

- ***Generic Subpractice*** e – resources and information necessary for performing the process are identified, made available, allocated and used
  Definition:
  Description:

- ***Generic Subpractice*** f – interfaces between the involved parties are managed to ensure both effective communication and also clear assignment of responsibility
  Definition:
  Description:

- ***Generic Practice*** GP 2.1.2 – Plan and monitor the performance of the process to fulfill the identified objectives.
  Definition: Plan(s) for the performance of the process are developed. The process performance cycle is defined.
  Description: Key milestones for the performance of the process are established. Estimates for process performance attributes are determined and maintained. Process activities and tasks are defined. Schedule is defined and aligned with the approach to performing the process. Process work product reviews are planned. The process is performed according to the plan(s). Process performance is monitored to ensure planned results are achieved.

  This ***Generic Practice*** has the following ***Generic Subpractices***:

  - ***Generic Subpractice*** a – objectives for the performance of the process are identified
    Definition:
    Description:

  - ***Generic Subpractice*** b – performance of the process is planned and monitored
    Definition:
    Description:

  - ***Generic Subpractice*** c – performance of the process is adjusted to meet plans
    Definition:
    Description:

  - ***Generic Subpractice*** d – responsibilities and authorities for performing the process are defined, assigned and communicated
    Definition:
    Description:

  - ***Generic Subpractice*** e – resources and information necessary for performing the process are identified, made available, allocated and used
    Definition:
    Description:

  - ***Generic Subpractice*** f – interfaces between the involved parties are managed to ensure both effective communication and also clear assignment of responsibility

Definition:
Description:

- ***Generic Practice*** GP 2.1.3 – Adjust the performance of the process.
  Definition: Process performance issues are identified.
  Description: Appropriate actions are taken when planned results and objectives are not achieved. The plan(s) are adjusted, as necessary. Rescheduling is performed as necessary.

  This ***Generic Practice*** has the following ***Generic Subpractices***:

  - ***Generic Subpractice*** a – objectives for the performance of the process are identified
    Definition:
    Description:

  - ***Generic Subpractice*** b – performance of the process is planned and monitored
    Definition:
    Description:

  - ***Generic Subpractice*** c – performance of the process is adjusted to meet plans
    Definition:
    Description:

  - ***Generic Subpractice*** d – responsibilities and authorities for performing the process are defined, assigned and communicated
    Definition:
    Description:

  - ***Generic Subpractice*** e – resources and information necessary for performing the process are identified, made available, allocated and used
    Definition:
    Description:

  - ***Generic Subpractice*** f – interfaces between the involved parties are managed to ensure both effective communication and also clear assignment of responsibility
    Definition:
    Description:

- ***Generic Practice*** GP 2.1.4 – Define responsibilities and authorities for performing the process.
  Definition: Responsibilities, commitments and authorities to perform the process are defined, assigned and communicated. Responsibilities and authorities to verify process work products are defined and assigned. The needs for process performance experience, knowledge and skills are defined.
  Description:

  This ***Generic Practice*** has the following ***Generic Subpractices***:

  - ***Generic Subpractice*** a – objectives for the performance of the process are identified
    Definition:
    Description:

  - ***Generic Subpractice*** b – performance of the process is planned and monitored
    Definition:
    Description:

  - ***Generic Subpractice*** c – performance of the process is adjusted to meet plans
    Definition:
    Description:

  - ***Generic Subpractice*** d – responsibilities and authorities for performing the process are defined, assigned and communicated
    Definition:
    Description:

  - ***Generic Subpractice*** e – resources and information necessary for performing the process are

identified, made available, allocated and used
Definition:
Description:

- ***Generic Subpractice*** f – interfaces between the involved parties are managed to ensure both effective communication and also clear assignment of responsibility
Definition:
Description:

- ***Generic Practice*** GP 2.1.5 – Identify and make available resources to perform the process according to plan.
Definition: The human and infrastructure resources necessary for performing the process are identified made available, allocated and used.
Description: The information necessary to perform the process is identified and made available. The necessary infrastructure and facilities are identified and made available. (...)

Process Area DRE – Desenvolvimento de Requisitos
Definition: O propósito do processo Desenvolvimento de Requisitos é definir os requisitos do cliente, do produto e dos componentes do produto.
Description

This ***Process Area*** has the following ***Specific Goal***:

Specific Goal Generic Identification – Generic Name
Definition: Generic Definition
Description: Generic Description
This ***Specific Goal*** has the following ***Specific Practices***:

- ***Specific Practice*** DRE 1 – As necessidades, expectativas e restrições do cliente, tanto do produto quanto de suas interfaces, são identificadas
Definition:
Description:

  This ***Specific Practice*** has the following ***Specific Subpractice***:

  - ***Specific Subpractice*** Generic Identification – Generic Name
Definition: Generic Definition
Description: Generic Description

    This ***Specific Practice*** has the following ***Typical Work Product***:

  - ***Typical Work Product*** Generic Identification – Generic Name
Definition: Generic Definition
Description: Generic Description

- ***Specific Practice*** DRE 2 – Um conjunto definido de requisitos do cliente é especificado e priorizado a partir das necessidades, expectativas e restrições identificadas
Definition:
Description:

  This ***Specific Practice*** has the following ***Specific Subpractice***:

  - ***Specific Subpractice*** Generic Identification – Generic Name
Definition: Generic Definition
Description: Generic Description

    This ***Specific Practice*** has the following ***Typical Work Product***:

  - ***Typical Work Product*** Generic Identification – Generic Name
Definition: Generic Definition
Description: Generic Description

- ***Specific Practice*** DRE 3 – Um conjunto de requisitos funcionais e não funcionais, do produto e dos componentes do produto que descrevem a solução do problema a ser resolvido, é definido e mantido a partir dos requisitos do cliente
  Definition:
  Description:

  This ***Specific Practice*** has the following ***Specific Subpractice***:

  - ***Specific Subpractice*** Generic Identification – Generic Name
    Definition: Generic Definition
    Description: Generic Description

    This ***Specific Practice*** has the following ***Typical Work Product***:

  - ***Typical Work Product*** Generic Identification – Generic Name
    Definition: Generic Definition
    Description: Generic Description

- ***Specific Practice*** DRE 4 – Os requisitos funcionais e não funcionais de cada componente do produto são refinados, elaborados e alocados;Interfaces internas e externas do produto e de cada componente do produto são definidas
  Definition:
  Description:

    This ***Specific Practice*** has the following ***Specific Subpractice***:

  - ***Specific Subpractice*** Generic Identification – Generic Name
    Definition: Generic Definition
    Description: Generic Description

    This ***Specific Practice*** has the following ***Typical Work Product***:

  - ***Typical Work Product*** Generic Identification – Generic Name
    Definition: Generic Definition
    Description: Generic Description

- ***Specific Practice*** DRE 5 – Conceitos operacionais e cenários são desenvolvidos
  Definition:
  Description:

  This ***Specific Practice*** has the following ***Specific Subpractice***:

  - ***Specific Subpractice*** Generic Identification – Generic Name
    Definition: Generic Definition
    Description: Generic Description

    This ***Specific Practice*** has the following ***Typical Work Product***:

  - ***Typical Work Product*** Generic Identification – Generic Name
    Definition: Generic Definition
    Description: Generic Description

- ***Specific Practice*** DRE 6 – Os requisitos são analisados, usando critérios definidos, para balancear as necessidades dos interessados com as restrições existentes
  Definition:
  Description:

  This ***Specific Practice*** has the following ***Specific Subpractice***:

  - ***Specific Subpractice*** Generic Identification – Generic Name

Definition: Generic Definition
Description: Generic Description

This ***Specific Practice*** has the following ***Typical Work Product***:

- ***Typical Work Product*** Generic Identification – Generic Name
  Definition: Generic Definition
  Description: Generic Description

- ***Specific Practice*** DRE 7 – Os requisitos são validados
  Definition:
  Description:

  This ***Specific Practice*** has the following ***Specific Subpractice***:

  - ***Specific Subpractice*** Generic Identification – Generic Name
    Definition: Generic Definition
    Description: Generic Description

    This ***Specific Practice*** has the following ***Typical Work Product***:

  - ***Typical Work Product*** Generic Identification – Generic Name
    Definition: Generic Definition
    Description: Generic Description

Capability Level NC 2 – Processo gerenciado
Definition:
Description:

This ***Capability Level*** has the following ***Generic Goals***:

Generic Goal AP 1.1 – O processo é executado
Definition: Este atributo evidencia o quanto o processo atinge o seu propósito.
Description:

This ***Generic Goal*** has the following ***Generic Practice***:

- ***Generic Practice*** RAP 1 – O processo atinge seus resultados definidos.
  Definition:
  Description:

  This ***Generic Practice*** has the following ***Generic Subpractice***:

  - ***Generic Subpractice*** Generic Identification – Generic Name
    Definition: Generic Definition
    Description: Generic Description

Generic Goal AP 2.1 – O processo é gerenciado
Definition: Este atributo evidencia o quanto a execução do processo é gerenciada.
Description:

This ***Generic Goal*** has the following ***Generic Practices***:

- ***Generic Practice*** RAP 2 – Existe uma política organizacional estabelecida e mantida para o processo.
  Definition:
  Description:

  This ***Generic Practice*** has the following ***Generic Subpractice***:

  - ***Generic Subpractice*** Generic Identification – Generic Name
    Definition: Generic Definition

Description: Generic Description

- ***Generic Practice*** RAP 3 – A execução do processo é planejada.
  Definition:
  Description:

  This ***Generic Practice*** has the following ***Generic Subpractice***:

  - ***Generic Subpractice*** Generic Identification – Generic Name
    Definition: Generic Definition
    Description: Generic Description

- ***Generic Practice*** RAP 4 – A execução do processo é monitorada e ajustes são realizados. Medidas são planejadas e coletadas para monitoração da execução do processo e ajustes são realizados.
  Definition:
  Description:

  This ***Generic Practice*** has the following ***Generic Subpractice***:

  - ***Generic Subpractice*** Generic Identification – Generic Name
    Definition: Generic Definition
    Description: Generic Description

- ***Generic Practice*** RAP 5 – As informações e os recursos necessários para a execução do processo são identificados e disponibilizados.
  Definition:
  Description:

  This ***Generic Practice*** has the following ***Generic Subpractice***:

  - ***Generic Subpractice*** Generic Identification – Generic Name
    Definition: Generic Definition
    Description: Generic Description

- ***Generic Practice*** RAP 6 – As responsabilidades e a autoridade para executar o processo são definidas, atribuídas e comunicadas.
  Definition:
  Description:

  This ***Generic Practice*** has the following ***Generic Subpractice***:

  - ***Generic Subpractice*** Generic Identification – Generic Name
    Definition: Generic Definition
    Description: Generic Description

- ***Generic Practice*** RAP 7 – As pessoas que executam o processo são competentes em termos de formação, treinamento e experiência.
  Definition:
  Description:

  This ***Generic Practice*** has the following ***Generic Subpractice***:

  - ***Generic Subpractice*** Generic Identification – Generic Name
    Definition: Generic Definition
    Description: Generic Description

- ***Generic Practice*** RAP 8 – A comunicação entre as partes interessadas no processo é planejada e executada de forma a garantir o seu envolvimento.
  Definition:
  Description:

This ***Generic Practice*** has the following ***Generic Subpractice***:

- ***Generic Subpractice*** Generic Identification – Generic Name
  Definition: Generic Definition
  Description: Generic Description

- ***Generic Practice*** RAP 9 – Os resultados do processo são revistos com a gerência de alto nível para fornecer visibilidade sobre a sua situação na organização.
  Definition:
  Description:

  This ***Generic Practice*** has the following ***Generic Subpractice***:

  - ***Generic Subpractice*** Generic Identification – Generic Name
    Definition: Generic Definition
    Description: Generic Description

- ***Generic Practice*** RAP 10 – O processo planejado para o projeto é executado. A aderência dos processos executados às descrições de processo, padrões e procedimentos é avaliada objetivamente e são tratadas as não conformidades.
  Definition:
  Description:

  This ***Generic Practice*** has the following ***Generic Subpractice***:

  - ***Generic Subpractice*** Generic Identification – Generic Name
    Definition: Generic Definition
    Description: Generic Description

Generic Goal AP 2.2 – Os produtos de trabalho do processo são gerenciados
Definition: Este atributo evidencia o quanto os produtos de trabalho produzidos pelo processo são gerenciados apropriadamente.
Description:

This ***Generic Goal*** has the following ***Generic Practices***:

- ***Generic Practice*** RAP 11 – Os requisitos dos produtos de trabalho do processo são identificados.
  Definition:
  Description:

  This ***Generic Practice*** has the following ***Generic Subpractice***:

  - ***Generic Subpractice*** Generic Identification – Generic Name
    Definition: Generic Definition
    Description: Generic Description

- ***Generic Practice*** RAP 12 – Requisitos para documentação e controle dos produtos de trabalho são estabelecidos.
  Definition:
  Description:

  This ***Generic Practice*** has the following ***Generic Subpractice***:
  - ***Generic Subpractice*** Generic Identification – Generic Name
    Definition: Generic Definition
    Description: Generic Description

- ***Generic Practice*** RAP 13 – Os produtos de trabalho são colocados em níveis apropriados de controle.
  Definition:
  Description:

  This ***Generic Practice*** has the following ***Generic Subpractice***:

- ***Generic Subpractice*** Generic Identification – Generic Name
  Definition: Generic Definition
  Description: Generic Description

- ***Generic Practice*** RAP 14 – Os produtos de trabalho são avaliados objetivamente com relação aos padrões, procedimentos e requisitos aplicáveis e são tratadas as não conformidades.
  Definition:
  Description:

  This ***Generic Practice*** has the following ***Generic Subpractice***:

  - ***Generic Subpractice*** Generic Identification – Generic Name
    Definition: Generic Definition
    Description: Generic Description

"(...)" – Significa que o texto ou parte do modelo foi omitido para melhorar a legibilidade.

Apêndice B

Questionário e resultados avaliados

Avaliação dos Perfis de Capacidade de Processo do PRO2PI-MMCA

Introdução

Meu nome é Edgar Lopes Banhesse, sou aluno do programa de mestrado da FEEC – Faculdade de Engenharia Elétrica e de Computação da UNICAMP – Universidade Estadual de Campinas. Meu projeto de pesquisa do mestrado, realizado em cooperação com a Divisão de Melhoria de Processo de Software (DMPS) do Centro de Tecnologia da Informação Renato Archer (CTI), trata da utilização de um Metamodelo de Perfis de Capacidade de Processo para Melhoria de Processo de Software a partir de Múltiplos Modelos. Esse metamodelo é denominado PRO2PI-MMC. Estou procurando por especialistas em modelos de capacidade de processo que possam responder esse questionário, com o intuito de avaliar os resultados do software PRO2PI-MMCA que implementa o PRO2PI-MMC. Por isso, gostaria muito de contar com 15 minutos de sua atenção para preencher este questionário, que me ajudará muito com meu projeto de mestrado.

O software utiliza o PRO2PI-MMC para, por exemplo, apresentar o conteúdo de um processo ou de uma área de processo de um modelo específico na arquitetura de outro modelo. Segundo o CMMI, a arquitetura de um modelo é o conjunto de estruturas necessárias para inferir sobre um modelo. Estas estruturas são compostas de elementos, das relações entre os elementos e das propriedades de ambos. Alguns dos principais elementos de um modelo CMMI incluem áreas de processo, metas específicas, práticas específicas, subpráticas específicas e produtos de trabalho. Uma relação da arquitetura CMMI, por exemplo, é uma meta específica com uma ou mais práticas específicas.

O objetivo da avaliação ao responder as questões é verificar o atendimento das relações lógicas entre os elementos que compõem a arquitetura dos modelos, seguido pelo respectivo conteúdo. Não é objetivo avaliar a formatação, ou seja, como os elementos, seguidos dos respectivos conteúdos, são apresentados graficamente. Para ilustrar essa orientação é apresentado na Figura B1 uma meta específica do CMMI-DEV versão 1.3, a "SG 1 – *Manage Requirements*" da área de processo "REQM – *Requirements Management*" na formatação utilizada pelo modelo CMMI e depois a mesma meta específica gerada pelo software PRO2PI-MMCA. A formatação é diferente, mas o conteúdo apresentado no elemento arquitetônico é o mesmo. O questionário é composto por três seções: introdução; orientação, que descreve quais resultados do PRO2PI-MMCA são apresentados para avaliação; e avaliação, que contém os resultados e as questões de avaliação dos resultados do PRO2PI-MMCA. Agradeço muito por sua colaboração e participação nessa avaliação.

SG 1 **Manage Requirements**

Requirements are managed and inconsistencies with project plans and work products are identified.

The project maintains a current and approved set of requirements over the life of the project by doing the following:

- Managing all changes to requirements
- Maintaining relationships among requirements, project plans, and work products
- Ensuring alignment among requirements, project plans, and work products
- Taking corrective action

Refer to the Requirements Development process area for more information about analyzing and validating requirements.

Refer to the Develop Alternative Solutions and Selection Criteria specific practice in the Technical Solution process area for more information about determining the feasibility of the requirements.

Refer to the Project Monitoring and Control process area for more information about managing corrective action to closure.

Specific Goal SG 1 - Manage Requirements
Definition: Requirements are managed and inconsistencies with project plans and work products are identified.
Description: The project maintains a current and approved set of requirements over the life of the project by doing the following:
- Managing all changes to requirements
- Maintaining relationships among requirements, project plans, and work products
- Ensuring alignment among requirements, project plans, and work products
- Taking corrective action

Refer to the Requirements Development process area for more information about analyzing and validating requirements.
Refer to the Develop Alternative Solutions and Selection Criteria specific practice in the Technical Solution process area for more information about determining the feasibility of the requirements.
Refer to the Project Monitoring and Control process area for more information about managing corrective action to closure.

Figura B1 – SG 1 – *Manage Requirements* do CMMI-DEV versão 1.3 x PRO2PI-MMCA.

Orientação

A próxima seção contém seis resultados do PRO2PI-MMCA, que são:

1) A área de processo "REQM – *Requirements Management*" do modelo CMMI-DEV versão 1.3 expressa na arquitetura SPICE;
2) A área de processo "REQM – *Requirements Management*" do modelo CMMI-DEV versão 1.3 expressa na arquitetura MPS.BR;
3) O processo "DRE – Desenvolvimento de Requisitos" do modelo MR-MPS-SW versão agosto/2012 expresso na arquitetura CMMI;
4) O processo "DRE – Desenvolvimento de Requisitos" do modelo MR-MPS-SW versão agosto/2012 expresso na arquitetura SPICE;
5) O processo "ENG.1 – *Requirements elicitation*" do modelo ASPICE versão 2.5 expresso na arquitetura CMMI; e
6) O processo "ENG.1 – *Requirements elicitation*" do modelo ASPICE versão 2.5 expresso na arquitetura MPS.BR.

Nos resultados gerados pelo PRO2PI-MMCA, optou-se por manter a integridade da arquitetura do modelo de destino, desta forma os elementos que ficariam sem conteúdo foram preenchidos com a palavra "*Generic*" seguida de "*identification*, *name*, *definition* e *description*". O símbolo "(...)", significa que o texto ou parte do modelo foi omitido para melhorar a legibilidade e a avaliação dos especialistas. O especialista em modelos deverá analisar um ou mais resultados do PRO2PI-MMCA e responder as suas respectivas questões, apenas se o mesmo tiver conhecimento e experiência prática nos modelos envolvidos no resultado específico avaliado.

Avaliação

Nome do Especialista (Opcional):

Você possui conhecimento e/ou experiência em qual(is) modelo(s)?

( ) CMMI-DEV versão 1.3 ( ) ASPICE versão 2.5 ( ) MR-MPS-SW versão agosto/2012

Quanto tempo de experiência no(s) modelos você possui?

Objetivo: avaliar se o conteúdo representativo de um determinado modelo de capacidade de processo foi expresso corretamente do ponto de vista lógico na arquitetura de outro modelo de capacidade de processo.

1) Área de processo "REQM – *Requirements Management*" do modelo CMMI-DEV versão 1.3 expressa na arquitetura SPICE

Process REQM – Requirements Management

Definition: The purpose of Requirements Management (REQM) is to manage requirements of the project's products and product components and to ensure alignment between those requirements and the project's plans and work products.

Description: Requirements management processes manage all requirements received or generated by the project, including both technical and nontechnical requirements as well as requirements levied on the project by the organization. In particular, if the Requirements Development process area is implemented, its processes will generate product and product component requirements that will also be managed by the requirements management processes. Throughout the process areas, where the terms "product" and "product component" are used, their intended meanings also encompass services, service systems, and their components. When the Requirements Management, Requirements Development, and Technical Solution process areas are all implemented, their associated processes can be closely tied and be performed concurrently. (…)

This ***Process*** has the following ***Process Outcome***:

Process Outcome SG 1 – Manage Requirements

Definition: Requirements are managed and inconsistencies with project plans and work products are identified.

Description: The project maintains a current and approved set of requirements over the life of the project by doing the following:

-Managing all changes to requirements

-Maintaining relationships among requirements, project plans, and work products

-Ensuring alignment among requirements, project plans, and work products

-Taking corrective action

Refer to the Requirements Development process area for more information about analyzing and validating requirements.

Refer to the Develop Alternative Solutions and Selection Criteria specific practice in the Technical Solution process area for more information about determining the feasibility of the requirements.

Refer to the Project Monitoring and Control process area for more information about managing corrective action to closure.

This ***Process Outcome*** has the following ***Base Practices***:

Base Practice SP 1.1 – Understand Requirements

Definition: Develop an understanding with the requirements providers on the meaning of the requirements.

Description: As the project matures and requirements are derived, all activities or disciplines will receive requirements. To avoid requirements creep, criteria are established to designate appropriate channels or official sources from which to receive requirements. Those who receive requirements conduct analyses of them with the provider to ensure that a compatible, shared understanding is reached on the meaning of requirements. The result of these analyses and dialogs is a set of approved requirements.

Base Practice SP 1.2 – Obtain Commitment to Requirements

Definition: Obtain commitment to requirements from project participants.

Description: Refer to the Project Monitoring and Control process area for more information about monitoring commitments. The previous specific practice dealt with reaching an understanding with requirements providers. This specific practice deals with agreements and commitments among those who carry out activities necessary to implement requirements. Requirements evolve throughout the project. As requirements evolve, this specific practice ensures that project participants commit to the current and approved requirements and the resulting changes in project plans, activities, and work products.

Base Practice SP 1.3 – Manage Requirements Changes

Definition: Manage changes to requirements as they evolve during the project.

Description: Refer to the Configuration Management process area for more information about tracking and controlling changes. Requirements change for a variety of reasons. As needs change and as work proceeds, changes may have to be made to existing requirements. It is essential to manage these additions and changes efficiently and effectively. To effectively analyze the impact of changes, it is necessary that the source of each requirement is known and the rationale for the change is documented. The project may want to track appropriate measures of requirements volatility to judge whether new or revised approach to change control is necessary.

Base Practice SP 1.4 – Maintain Bidirectional Traceability of Requirements

Definition: Maintain bidirectional traceability among requirements and work products.

Description: The intent of this specific practice is to maintain the bidirectional traceability of requirements. (See the definition of "bidirectional traceability" in the glossary.) When requirements are managed well, traceability can be established from a source requirement to its lower level requirements and from those lower level requirements back to their source requirements. (…)

Base Practice SP 1.5 – Ensure Alignment Between Project Work and Requirements

Definition: Ensure that project plans and work products remain aligned with requirements.

Description: This specific practice finds inconsistencies between requirements and project plans and work products and initiates corrective actions to resolve them.

This ***Process Outcome*** has the following ***Work Products***:

Work Product 1 – Lists of criteria for distinguishing appropriate requirements providers

Definition:

Description:

Work Product 2 – Criteria for evaluation and acceptance of requirements

Definition:

Description:

Work Product 3 – Results of analyses against criteria

Definition:

Description:

Work Product 4 – A set of approved requirements

Definition:

Description:

Work Product 1 – Requirements impact assessments

Definition:

Description:

Work Product 2 – Documented commitments to requirements and requirements changes

Definition:

Description:

Work Product 1 – Requirements change requests

Definition:

Description:

Work Product 2 – Requirements change impact reports

Definition:

Description:

Work Product 3 – Requirements status

Definition:

Description:

Work Product 4 – Requirements database

Definition:

Description:

Work Product 1 – Requirements traceability matrix

Definition:

Description:

Work Product 2 – Requirements tracking system
Definition:
Description:

Work Product 1 – Documentation of inconsistencies between requirements and project plans and work products, including sources and conditions
Definition:
Description:

Work Product 2 – Corrective actions
Definition:
Description:

Questões

1) O conteúdo da área de processo REQM do CMMI-DEV versão 1.3 foi expresso corretamente do ponto de vista lógico na arquitetura SPICE?
    a. Contempla Totalmente.
    b. Contempla Parcialmente.
    c. Não Contempla.

    Caso tenha respondido b ou c, favor justificar.

2) Comentários gerais:
    a. Quais os pontos fortes apresentados no resultado?
    b. Quais os pontos fracos apresentados no resultado?
    c. Quais as suas sugestões de melhoria?
    d. De uma forma geral o resultado apresentado foi satisfatório?

2) Área de processo "REQM – *Requirements Management*" do modelo CMMI-DEV versão 1.3 expressa na arquitetura MPS.BR

Process REQM – Requirements Management
Definition: The purpose of Requirements Management (REQM) is to manage requirements of the project's products and product components and to ensure alignment between those requirements and the project's plans and work products.
Description: Requirements management processes manage all requirements received or generated by the project, including both technical and nontechnical requirements as well as requirements levied on the project by the organization. In particular, if the Requirements Development process area is implemented, its processes will generate product and product component requirements that will also be managed by the requirements management processes. Throughout the process areas, where the terms "product" and "product component" are used, their intended meanings also encompass services, service systems, and their components. When the Requirements Management, Requirements Development, and Technical Solution process areas are all implemented, their associated processes can be closely tied and be performed concurrently. (…)

This ***Process*** has the following ***Process Outcomes***:

- ***Process Outcome*** SP 1.1 – Understand Requirements
  Definition: Develop an understanding with the requirements providers on the meaning of the requirements.
  Description: As the project matures and requirements are derived, all activities or disciplines will receive requirements. To avoid requirements creep, criteria are established to designate appropriate channels or official sources from which to receive requirements. Those who receive requirements conduct analyses of them with the provider to ensure that a compatible, shared understanding is reached on the meaning of requirements. The result of these analyses and dialogs is a set of approved requirements.

- ***Process Outcome*** SP 1.2 – Obtain Commitment to Requirements
  Definition:Obtain commitment to requirements from project participants.
  Description: Refer to the Project Monitoring and Control process area for more information about monitoring commitments. The previous specific practice dealt with reaching an understanding with requirements providers. This specific practice deals with agreements and commitments among those who carry out activities necessary to implement requirements. Requirements evolve throughout the project. As requirements evolve, this specific practice ensures that project participants commit to the current and approved requirements and the resulting changes in project plans, activities, and work products.

- ***Process Outcome*** SP 1.3 – Manage Requirements Changes
  Definition: Manage changes to requirements as they evolve during the project.
  Description: Refer to the Configuration Management process area for more information about tracking and controlling changes. Requirements change for a variety of reasons. As needs change and as work proceeds, changes may have to be made to existing requirements. It is essential to manage these additions and changes efficiently and effectively. To effectively analyze the impact of changes, it is necessary that the source of each requirement is known and the rationale for the change is documented. The project may want to track appropriate measures of requirements volatility to judge whether new or revised approach to change control is necessary.

- ***Process Outcome*** SP 1.4 – Maintain Bidirectional Traceability of Requirements
  Definition: Maintain bidirectional traceability among requirements and work products.
  Description: The intent of this specific practice is to maintain the bidirectional traceability of requirements. (See the definition of "bidirectional traceability" in the glossary.) When requirements are managed well, traceability can be established from a source requirement to its lower level requirements and from those lower level requirements back to their source requirements. (…)

- ***Process Outcome*** SP 1.5 – Ensure Alignment Between Project Work and Requirements
  Definition: Ensure that project plans and work products remain aligned with requirements.
  Description: This specific practice finds inconsistencies between requirements and project plans and work products and initiates corrective actions to resolve them.

Questões

1) O conteúdo da área de processo REQM do CMMI-DEV versão 1.3 foi expresso corretamente do ponto de vista lógico na arquitetura MPS.BR?
    a. Contempla Totalmente.
    b. Contempla Parcialmente.
    c. Não Contempla.

    Caso tenha respondido b ou c, favor justificar.

2) Comentários gerais:
    a. Quais os pontos fortes apresentados no resultado?
    b. Quais os pontos fracos apresentados no resultado?
    c. Quais as suas sugestões de melhoria?
    d. De uma forma geral o resultado apresentado foi satisfatório?

3) O processo "DRE – Desenvolvimento de Requisitos" do modelo MR-MPS-SW versão agosto/2012 expresso na arquitetura CMMI

Process Area DRE – Desenvolvimento de Requisitos
Definition: O propósito do processo Desenvolvimento de Requisitos é definir os requisitos do cliente, do produto e dos componentes do produto.
Description

This ***Process Area*** has the following ***Specific Goal***:

Specific Goal Generic Identification – Generic Name
Definition: Generic Definition
Description: Generic Description
This ***Specific Goal*** has the following ***Specific Practices***:

- ***Specific Practice*** DRE 1 – As necessidades, expectativas e restrições do cliente, tanto do produto quanto de suas interfaces, são identificadas
  Definition:
  Description:

  This ***Specific Practice*** has the following ***Specific Subpractice***:

  - ***Specific Subpractice*** Generic Identification – Generic Name
    Definition: Generic Definition
    Description: Generic Description

This ***Specific Practice*** has the following ***Typical Work Product***:

- ***Typical Work Product*** Generic Identification – Generic Name
  Definition: Generic Definition
  Description: Generic Description

- ***Specific Practice*** DRE 2 – Um conjunto definido de requisitos do cliente é especificado e priorizado a partir das necessidades, expectativas e restrições identificadas
  Definition:
  Description:

  This ***Specific Practice*** has the following ***Specific Subpractice***:

  - ***Specific Subpractice*** Generic Identification – Generic Name
    Definition: Generic Definition
    Description: Generic Description

    This ***Specific Practice*** has the following ***Typical Work Product***:

  - ***Typical Work Product*** Generic Identification – Generic Name
    Definition: Generic Definition
    Description: Generic Description

- ***Specific Practice*** DRE 3 – Um conjunto de requisitos funcionais e não funcionais, do produto e dos componentes do produto que descrevem a solução do problema a ser resolvido, é definido e mantido a partir dos requisitos do cliente
  Definition:
  Description:

  This ***Specific Practice*** has the following ***Specific Subpractice***:

  - ***Specific Subpractice*** Generic Identification – Generic Name
    Definition: Generic Definition
    Description: Generic Description

    This ***Specific Practice*** has the following ***Typical Work Product***:

  - ***Typical Work Product*** Generic Identification – Generic Name
    Definition: Generic Definition
    Description: Generic Description

- ***Specific Practice*** DRE 4 – Os requisitos funcionais e não funcionais de cada componente do produto são refinados, elaborados e alocados;Interfaces internas e externas do produto e de cada componente do produto são definidas
  Definition:
  Description:

  This ***Specific Practice*** has the following ***Specific Subpractice***:

  - ***Specific Subpractice*** Generic Identification – Generic Name
    Definition: Generic Definition
    Description: Generic Description

    This ***Specific Practice*** has the following ***Typical Work Product***:

  - ***Typical Work Product*** Generic Identification – Generic Name
    Definition: Generic Definition
    Description: Generic Description

- ***Specific Practice*** DRE 5 – Conceitos operacionais e cenários são desenvolvidos

Definition:
Description:

This ***Specific Practice*** has the following ***Specific Subpractice***:

- ***Specific Subpractice*** Generic Identification – Generic Name
  Definition: Generic Definition
  Description: Generic Description

  This ***Specific Practice*** has the following ***Typical Work Product***:

- ***Typical Work Product*** Generic Identification – Generic Name
  Definition: Generic Definition
  Description: Generic Description

- ***Specific Practice*** DRE 6 – Os requisitos são analisados, usando critérios definidos, para balancear as necessidades dos interessados com as restrições existentes
  Definition:
  Description:

  This ***Specific Practice*** has the following ***Specific Subpractice***:

  - ***Specific Subpractice*** Generic Identification – Generic Name
    Definition: Generic Definition
    Description: Generic Description

    This ***Specific Practice*** has the following ***Typical Work Product***:

  - ***Typical Work Product*** Generic Identification – Generic Name
    Definition: Generic Definition
    Description: Generic Description

- ***Specific Practice*** DRE 7 – Os requisitos são validados
  Definition:
  Description:

  This ***Specific Practice*** has the following ***Specific Subpractice***:

  - ***Specific Subpractice*** Generic Identification – Generic Name
    Definition: Generic Definition
    Description: Generic Description

    This ***Specific Practice*** has the following ***Typical Work Product***:

  - ***Typical Work Product*** Generic Identification – Generic Name
    Definition: Generic Definition
    Description: Generic Description

Questões

1) O conteúdo do processo DRE do MR-MPS-SW versão agosto/2012 foi expresso corretamente do ponto de vista lógico na arquitetura CMMI?
   a. Contempla Totalmente.
   b. Contempla Parcialmente.

c. Não Contempla.

Caso tenha respondido b ou c, favor justificar.

2) Comentários gerais:
   a. Quais os pontos fortes apresentados no resultado?
   b. Quais os pontos fracos apresentados no resultado?
   c. Quais as suas sugestões de melhoria?
   d. De uma forma geral o resultado apresentado foi satisfatório?

4) O processo "DRE – Desenvolvimento de Requisitos" do modelo MR-MPS-SW versão agosto/2012 expresso na arquitetura SPICE

Process DRE – Desenvolvimento de Requisitos
Definition: O propósito do processo Desenvolvimento de Requisitos é definir os requisitos do cliente, do produto e dos componentes do produto.
Description:

This ***Process*** has the following ***Process Outcome***:

- ***Process Outcome*** Generic Identification – Generic Name
  Definition: Generic Definition
  Description: Generic Description

  This ***Process Outcome*** has the following ***Base Practices***:

  - ***Base Practice*** DRE 1 – As necessidades, expectativas e restrições do cliente, tanto do produto quanto de suas interfaces, são identificadas
    Definition:
    Description:
  - ***Base Practice*** DRE 2 – Um conjunto definido de requisitos do cliente é especificado e priorizado a partir das necessidades, expectativas e restrições identificadas
    Definition:
    Description:
  - ***Base Practice*** DRE 3 – Um conjunto de requisitos funcionais e não funcionais, do produto e dos componentes do produto que descrevem a solução do problema a ser resolvido, é definido e mantido a partir dos requisitos do cliente
    Definition:
    Description:
  - ***Base Practice*** DRE 4 – Os requisitos funcionais e não funcionais de cada componente do produto são refinados, elaborados e alocados;Interfaces internas e externas do produto e de cada componente do produto são definidas
    Definition:
    Description:
  - ***Base Practice*** DRE 5 – Conceitos operacionais e cenários são desenvolvidos

Definition:
Description:

- ***Base Practice*** DRE 6 – Os requisitos são analisados, usando critérios definidos, para balancear as necessidades dos interessados com as restrições existentes
  Definition:
  Description:

- ***Base Practice*** DRE 7 – Os requisitos são validados
  Definition:
  Description:

This ***Process Outcome*** has the following ***Work Product***:

- ***Work Product*** Generic Identification – Generic Name
  Definition: Generic Definition
  Description: Generic Description

Questões

1) O conteúdo do processo DRE do MR-MPS-SW versão agosto/2012 foi expresso corretamente do ponto de vista lógico na arquitetura SPICE?
   a. Contempla Totalmente.
   b. Contempla Parcialmente.
   c. Não Contempla.

   Caso tenha respondido b ou c, favor justificar.

2) Comentários gerais:
   a. Quais os pontos fortes apresentados no resultado?
   b. Quais os pontos fracos apresentados no resultado?
   c. Quais as suas sugestões de melhoria?
   d. De uma forma geral o resultado apresentado foi satisfatório?

5) O processo "ENG.1 – *Requirements elicitation*" do modelo ASPICE versão 2.5 expresso na arquitetura CMMI

Process Area ENG.1 – Requirements elicitation
Definition: The purpose of the Requirements elicitation process is to gather, process, and track evolving customer needs and requirements throughout the life of the product and/or service so as to establish a requirements baseline that serves as the basis for defining the needed work products.

Description:
This ***Process Area*** has the following ***Specific Goals***:

Specific Goal 1 – continuing communication with the customer is established
Definition:
Description:

This ***Specific Goal*** has the following ***Specific Practice***:

- ***Specific Practice*** ENG.1.BP1 – Obtain customer requirements and requests.
  Definition: Obtain and define customer requirements and requests through direct solicitation of customer input and through review of customer business proposals (where relevant), target operating and hardware environment, and other documents bearing on customer requirements.
  Description: NOTE 1: The information needed to keep traceability for each customer requirement has to be gathered and documented.

  This ***Specific Practice*** has the following ***Specific Subpractice***:

  - ***Specific Subpractice*** Generic Identification – Generic Name
    Definition: Generic Definition
    Description: Generic Description

    This ***Specific Practice*** has the following ***Typical Work Products***:

  - ***Typical Work Product*** 13-04 – Communication record
    Definition:
    Description:
    - All forms of interpersonal communication, record including:
    -- letters
    -- faxes
    -- e-mails
    -- voice recordings
    -- telexes

  - ***Typical Work Product*** 17-03 – Customer Requirements
    Definition:
    Description:
    - Purpose/objectives defined requirements
    - Includes issues/requirements from (contract) reviews
    - Identifies any:
    -- time schedule/constraints
    -- required feature and functional characteristics
    -- necessary performance considerations/constraints
    -- necessary internal/external interface considerations/constraints
    -- required system characteristics/constraints
    -- human engineering considerations/constraints
    -- security considerations/constraints
    -- environmental considerations/constraints
    -- operational considerations/constraints
    -- maintenance considerations/constraints
    -- installation considerations/constraints
    -- support considerations/constraints
    -- design constraints
    -- safety/reliability considerations/constraints
    -- quality requirements/expectations

Specific Goal 2 – agreed customer requirements are defined and baselined
Definition:
Description:

This ***Specific Goal*** has the following ***Specific Practices***:

- ***Specific Practice*** ENG.1.BP2 – Understand customer expectations.
  Definition: Ensure that both supplier and customer understand each requirement in the same way.
  Description: NOTE 2: Review with customers their requirements and requests to better understand their needs and expectations. Refer to the process SUP.4 Joint Review.

  This ***Specific Practice*** has the following ***Specific Subpractice***:

  - ***Specific Subpractice*** Generic Identification – Generic Name
    Definition: Generic Definition
    Description: Generic Description

    This ***Specific Practice*** has the following ***Typical Work Products***:

  - ***Typical Work Product*** 15-01 – Analysis report
    Definition:
    Description:
    - What was analyzed
    - Who did the analysis
    - The analysis criteria used:
    -- selection criteria or prioritization scheme used
    -- decision criteria
    -- quality criteria
    - Records the results:
    -- what was decided/selected
    -- reason for the selection
    -- assumptions made
    -- potential risks
    - Aspects of correctness to analyze include:
    -- completeness
    -- understandability
    -- testability
    -- verifiability
    -- feasibility
    -- validity
    -- consistency
    -- adequacy of content

  - ***Typical Work Product*** 17-03 – Customer Requirements
    Definition:
    Description:
    - Purpose/objectives defined requirements
    - Includes issues/requirements from (contract) reviews
    - Identifies any:
    -- time schedule/constraints
    -- required feature and functional characteristics
    -- necessary performance considerations/constraints
    -- necessary internal/external interface considerations/constraints
    -- required system characteristics/constraints
    -- human engineering considerations/constraints
    -- security considerations/constraints
    -- environmental considerations/constraints
    -- operational considerations/constraints
    -- maintenance considerations/constraints
    -- installation considerations/constraints
    -- support considerations/constraints
    -- design constraints
    -- safety/reliability considerations/constraints
    -- quality requirements/expectations

- ***Specific Practice*** ENG.1.BP3 – Agree on requirements.
  Definition: Obtain an explicit agreement from all relevant parties to work to these requirements.
  Description:

This ***Specific Practice*** has the following ***Specific Subpractice***:

- ***Specific Subpractice*** Generic Identification – Generic Name
  Definition: Generic Definition
  Description: Generic Description

  This ***Specific Practice*** has the following ***Typical Work Products***:

- ***Typical Work Product*** 15-01 – Analysis report
  Definition:
  Description:
  - What was analyzed
  - Who did the analysis
  - The analysis criteria used:
  -- selection criteria or prioritization scheme used
  -- decision criteria
  -- quality criteria
  - Records the results:
  -- what was decided/selected
  -- reason for the selection
  -- assumptions made
  -- potential risks
  - Aspects of correctness to analyze include:
  -- completeness
  -- understandability
  -- testability
  (…)

Questões

1) O conteúdo do processo ENG.1 do ASPICE versão 2.5 foi expresso corretamente do ponto de vista lógico na arquitetura CMMI?
   a. Contempla Totalmente.
   b. Contempla Parcialmente.
   c. Não Contempla.

   Caso tenha respondido b ou c, favor justificar.

2) Comentários gerais:
   a. Quais os pontos fortes apresentados no resultado?
   b. Quais os pontos fracos apresentados no resultado?
   c. Quais as suas sugestões de melhoria?
   d. De uma forma geral o resultado apresentado foi satisfatório?

6) O processo "ENG.1 – *Requirements elicitation*" do modelo ASPICE versão 2.5 expresso na arquitetura MPS.BR

Process ENG.1 – Requirements elicitation
Definition: The purpose of the Requirements elicitation process is to gather, process, and track evolving customer needs and requirements throughout the life of the product and/or service so as to establish a requirements baseline that serves as the basis for defining the needed work products.
Description:

This ***Process*** has the following ***Process Outcomes***:

- ***Process Outcome*** ENG.1.BP1 – Obtain customer requirements and requests.
  Definition: Obtain and define customer requirements and requests through direct solicitation of customer input and through review of customer business proposals (where relevant), target operating and hardware environment, and other documents bearing on customer requirements.
  Description: NOTE 1: The information needed to keep traceability for each customer requirement has to be gathered and documented.

- ***Process Outcome*** ENG.1.BP2 – Understand customer expectations.
  Definition: Ensure that both supplier and customer understand each requirement in the same way.
  Description: NOTE 2: Review with customers their requirements and requests to better understand their needs and expectations. Refer to the process SUP.4 Joint Review.

- ***Process Outcome*** ENG.1.BP3 – Agree on requirements.
  Definition: Obtain an explicit agreement from all relevant parties to work to these requirements.
  Description:

- ***Process Outcome*** ENG.1.BP4 – Establish customer requirements baseline.
  Definition: Formalize the customer's requirements and establish as a baseline for project use and monitoring against customer needs.
  Description: The supplier should determine the requirements not stated by the customer but necessary for specified and intended use and include them in the baseline.

- ***Process Outcome*** ENG.1.BP5 – Manage customer requirements changes.
  Definition: Manage all changes made to the customer requirements against the customer requirements baseline to ensure enhancements resulting from changing technology and customer needs are identified and that those who are affected by the changes are able to assess the impact and risks and initiate appropriate change control and mitigation actions.
  Description: NOTE 3: Requirements change may arise from different sources as for instance changing technology and customer needs, legal constraints.

- ***Process Outcome*** ENG.1.BP6 – Establish customer-supplier query communication mechanism.
  Definition: Provide a means by which the customer can be aware of the status and disposition of their requirements changes and the supplier can have the ability to communicate necessary information, including data, in a customer-specified language and format.
  Description: NOTE 4: This may include joint meetings with the customer or formal communication to review the status for their requirements and requests; Refer to the process SUP.4 Joint Review. NOTE 5: The formats of the information communicated by the supplier may include computer-aided design data and electronic data exchange.

Questões

1) O conteúdo do processo ENG.1 do ASPICE versão 2.5 foi expresso corretamente do ponto de vista lógico na arquitetura MPS.BR?
    a. Contempla Totalmente.
    b. Contempla Parcialmente.
    c. Não Contempla.

    Caso tenha respondido b ou c, favor justificar.

2) Comentários gerais:
    a. Quais os pontos fortes apresentados no resultado?
    b. Quais os pontos fracos apresentados no resultado?
    c. Quais as suas sugestões de melhoria?
    d. De uma forma geral o resultado apresentado foi satisfatório?