Universidade Estadual de Londrina

VANESSA MATIAS LEITE

UMA INSTÂNCIA DE UMA ARQUITETURA ORIENTADA A MODELO E SUAS IMPLICAÇÕES NA IMPLANTAÇÃO DOS PROCESSOS DO MPS.BR

Londrina
2017

VANESSA MATIAS LEITE

UMA INSTÂNCIA DE UMA ARQUITETURA ORIENTADA A MODELO E SUAS IMPLICAÇÕES NA IMPLANTAÇÃO DOS PROCESSOS DO MPS.BR

Dissertação apresentada ao Programa de Mestrado em Ciência da Computação da Universidade Estadual de Londrina para obtenção do título de Mestre em Ciência da Computação.

Orientador: Prof(a). Dr(a). Jandira Guenka Palma.

Londrina
2017

Vanessa Matias Leite

Uma instância de uma arquitetura orientada a modelo e suas implicações na implantação dos processos do MPS.BR/ Vanessa Matias Leite. – Londrina–PR, 2017-

157 p. : il. (algumas color.) ; 30 cm.

Orientador: Prof(a). Dr(a). Jandira Guenka Palma

– Universidade Estadual de Londrina, 2017.

1. Model Driven Architecture. 2. MPS.BR. I. Jandira Guenka Palma. II. Universidade Estadual de Londrina. Centro de Ciências Exatas. Programa de Pós-Graduação em Ciência da Computação III. Título

CDU 02:141:005.7

VANESSA MATIAS LEITE

UMA INSTÂNCIA DE UMA ARQUITETURA ORIENTADA A MODELO E SUAS IMPLICAÇÕES NA IMPLANTAÇÃO DOS PROCESSOS DO MPS.BR

Dissertação apresentada ao Programa de Mestrado em Ciência da Computação da Universidade Estadual de Londrina para obtenção do título de Mestre em Ciência da Computação.

BANCA EXAMINADORA

Orientador: Prof(a). Dr(a). Jandira Guenka Palma
Universidade Estadual de Londrina - UEL

Prof. Dr. Willian Massami Watanabe
Universidade Tecnológica Federal do Paraná - UTFPR

Prof. Dr. Adilson Bonifácio
Universidade Estadual de Londrina - UEL

Prof. Dr. Jacques Duílio Brancher
Universidade Estadual de Londrina - UEL

Londrina, 22 de Março de 2017.

AGRADECIMENTOS

Gostaria de agradecer a todos aqueles que estiveram comigo em mais essa fase da minha vida:

Primeiramente a Deus, que esteve comigo em todos os momentos em minha vida e no mestrado não foi diferente, dando-me força e fé para conseguir vencer os obstáculos e continuar trilhando o caminho certo.

A minha família, principalmente aos meus pais e ao meu irmão, que foram pacientes (nem sempre) e carinhosos e me deram todo o apoio necessário e estiveram comigo em mais essa jornada.

Ao meu noivo Eduardo Inocente que esteve comigo me apoiando e aconselhando-me nesta etapa, sem sua presença todo esse processo se tornaria mais difícil.

A minha orientadora Prof. Dr^{a}. Jandira Guenka Palma, por me acompanhar durante esses anos oferecendo todo suporte necessário e compartilhando comigo o seu conhecimento.

Gostaria de expressar minha gratidão também ao professor Dr. Adilson Bonifácio, por ter aceitado ser meu orientador, mesmo sem me conhecer, no começo do mestrado.

Aos meus amigos, em especial a Brenda e Priscila que estão comigo desde longa data e nos momentos difíceis me proporcionaram uma amizade fundamental e horas de diversões e risadas.

LEITE, V. M. **Uma instância de uma arquitetura orientada a modelo e suas implicações na implantação dos processos do MPS.BR**. 157 p. Dissertação de Mestrado (Mestrado em Ciência da Computação) – Universidade Estadual de Londrina, Londrina–PR, 2017.

RESUMO

O desenvolvimento de software ainda apresenta diversos desafios. São diversas etapas desde o levantamento de requisitos até a codificação. Para melhorar o desenvolvimento de software, as organizações buscam implementar um modelo de referência de qualidade, que tem por objetivo fornecer diretrizes e práticas para a melhoria do processo de software, implicando assim na qualidade do produto. O MPS.BR (Melhoria do Processo de Software Brasileiro) é o modelo de qualidade nacional, voltado para pequenas e médias empresas. Apesar de instruir boas práticas de qualidade, os modelos de referência não possuem a finalidade de definir a realização das diversas etapas de desenvolvimento. Assim, para produzir os resultados esperados dos modelos de qualidade, ainda precede-se de um método de desenvolvimento. Assim sendo, foi proposto o emprego do *Model Driven Architecture*- MDA. Neste trabalho buscou-se analisar o quão uma instância e os conceitos do MDA podem ter efeitos sobre os processos do MPS.BR. Para isso, primeiramente foi elaborada uma instância de MDA, para depois a realização das duas análises. A primeira análise é referente aos artefatos da instância e os conceitos do MDA sobre os processos do MPS.BR nível G. A segunda análise é sobre quais artefatos da instância de MDA podem ser utilizados como evidências de certos resultados esperados dos processos do MPS.BR. Para averiguar essas análises foi realizada uma aplicação de questionário com especialista que avaliaram o que foi proposto . Como resultado destas avaliações constatou-se que a instância criada e os conceitos do MDA auxiliam na implementação do MPS.BR.

Palavras-chave: Model Driven Architecture, MPS.BR, Model Driven Development. Modelo de Referência

LEITE, V. M. **An instance of a model driven architecture and its implications for the implantation of MPS.BR processes**. 157 p. Master's Thesis (Master in Science in Computer Science) – State University of Londrina, Londrina–PR, 2017.

ABSTRACT

Software development still presents several challenges. There are several steps from requirements gathering to coding. To improve software development, organizations seek to implement a quality reference model, which aims to provide guidelines and practices for the improvement of the software process, implying product quality. The MPS.BR is the national quality model, aimed at small and medium business. Despite instructing good quality practices, the reference models do not have the purpose of defining the achievement of the various stages of development. Thus, to produce the expected results of quality models, a development method is still necessary. Therefore, the use of the Model Driven Architecture - MDA was proposed. In this work we aimed to analyze how an instance and the concepts of MDA can have effects on the processes of MPS.BR. For this, an MDA instance was first elaborated, to later carry out two analyzes. The first analysis is referring to the instance artifacts and concepts of MDA on the processes of MPS.BR level G. The second analysis is about which MDA instance artifacts can be used as evidence of certain expected results of MPS.BR processes. Specialists avaluated these analyzes via a questionnarie and as a result of these evaluations it was verified that the created instance and the concepts of the MDA assist in the implementation of the MPS.BR.

Keywords: Model Driven Architecture, MPS.BR, Model Driven Development. SPI

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AMP	Avaliação e Melhoria do Processo Organizacional
CIM	*Computation Independent Model*
CMMI	*Capability Maturity Model*
DFP	Definição do Processo Organizacional
DRE	Desenvolvimento de Requisitos
DRU	Desenvolvimento para Reutilização
GPR	Gerência de Processos
GRE	Gerência de Requisitos
ITP	Integração do Produto
MDA	*Model Driven Architecture*
MDD	*Model Driven Development*
MED	Medição
MVC	*Model View Controller*
MOF	*Meta Object Facility*
OMG	*Object Management Group*
PIM	*Platform Independent Model*
PCP	Projeto e Construção do Produto
PSM	*Platform Specific Model*
RE	Resultados Esperados
SCAMPI	*Standard CMMI Appraisal Method for Process Improvement*
SPI	*Software Process Improvement*
UML	*Unified Modeling Language*
VAL	Validação
VER	Verificação
XMI	XML Metadata Interchange

SUMÁRIO

1 INTRODUÇÃO

O desenvolvimento de software é uma área que ainda apresenta vários desafios. Do desdobramento do problema para o código existem várias etapas e esta é uma mão-de-obra intensiva. Os desafios consistem em aumentar a produtividade, facilitar e melhorar o progresso dos vários estágios de desenvolvimento. Outro desafio encontrado é que cada vez que a tecnologia muda ou evolui muito trabalho precisa ser refeito [1]. Assim, as empresas buscam solucionar os problemas por meio da implantação de modelos de qualidade ou por meios de novas metodologias de desenvolvimento de software.

Muitas empresas buscam melhorias em seus processos de software com base em modelos de qualidade. Estes, também comumente conhecidos como modelos de referência ou SPI (*Software Process Improvement*- Melhoria do Processo de Software), trazem diretrizes e práticas para melhorar o processo de uma organização, levando a uma padronização do fluxo de desenvolvimento, que reflete em processos e produtos de maior qualidade.

Os SPI's são modelos que visam instituir melhorias aos processos das organizações, sendo que cada modelo de referência tem suas próprias características. Cabe a cada organização analisar seus pontos fortes e fracos e optar por um modelo que se aproxime mais de sua realidade e que agregará maiores benefícios à empresa. Alguns SPI's presentes no mercado de hoje são o CMMI (*Capability Maturity Model Integration*), o ISO/IEC 12207, o ISO/IEC 15504, o MPS.BR.

O CMMI é um modelo de maturidade internacional voltado para o desenvolvimento de produtos e serviços de software [4]. O ISO/IEC 12207 visa à definição, avaliação e melhoria dos processos de software [5]. Já o ISO/IEC 15504, também conhecido como SPICE, visa à melhoria nos processos do projeto e não da empresa como um todo [6]. O MPS.BR é o modelo brasileiro, direcionado principalmente para empresas de pequeno e médio porte e tem como principal objetivo a melhoria dos processos na empresa[7]. Com o emprego dos modelos de referência, as empresas relatam que há benefícios para o processo da organização e para a qualidade dos seus produtos, implicando assim em um aumento na satisfação dos clientes [8], [9], [10], [11] e [12].

Apesar dos benefícios que os modelos de referência podem oferecer, estes também podem acarretar certas dificuldades para as empresas. A dificuldade para adaptação ao novo modelo e até mesmo a resistência dos colaboradores podem ser problemas para implementação e manutenção de um modelo de qualidade. Dentre algumas das dificuldades levantadas por organizações que adotam o MPS.BR, podem ser citadas: (*i*) documentação excessiva [13] e [9]; (*ii*) não aumento da produtividade [13] e [14]; (*iii*) subjetividade do

modelo de referência [13]; (*iv*) muita burocratização [15]; (*v*) falta de ferramentas [16], [13] e [17].

Para solucionar ou mitigar essas dificuldades, propõe-se utilizar um modelo de desenvolvimento de software, o qual, na sua essência, possui fatores que possam auxiliar a diminuir determinadas dificuldades encontradas pelas organizações na implementação e manutenção do MPS.BR. O modelo de desenvolvimento sugerido é o *Model Driven Development*- MDD, este apresenta vantagens como[1], [18], [19] e [20]: produtividade, manutenção, documentação, corretude e reutilização. O MDD possuí uma especificação, o *Model Driven Architecture* – MDA. O MDA apresenta um formalismo maior que o MDD, e isto auxiliaria na dificuldade da subjetividade.

Portanto, este trabalho propõe-se a utilizar o *Model Driven Architecture* - MDA, de forma que este, além de direcionar o processo de desenvolvimento de software, possa com seus artefatos atender as evidências do MPS.BR, assim como minimizar algumas dificuldades relatadas e ainda manter os seus benefícios.

O MDA é uma implementação do MDD mantida pela OMG- *Object Management Group*, e utiliza modelos como artefatos de desenvolvimento primário, partindo assim de um nível maior de abstração [21].

De acordo com a OMG, o MDA é um conjunto de padrões utilizados de maneira conjunta, sendo que se destacam os padrões UML– *Unified Model Language* e MOF - *Meta-Object Facility*, ambos também definidos pela OMG [22]. No MDA, inicialmente são construídos modelos/diagramas, os quais passam por transformações até atingir o código ou um modelo específico. Esses níveis de abstrações possuem o propósito de descrever um sistema de software sob uma perspectiva particular. Estas abstrações especificadas pela OMG são o CIM (*Computation Independent Model*),o PIM(*Platform Independent Model*) e o PSM(*Platform Specific Model*). No CIM se tem uma visão do sistema independente de computação, não mostrando detalhes da estrutura do sistema [22]. Já o PIM foca na operação do sistema, no entanto, não se preocupa com detalhes da implementação em uma plataforma específica [22]. Por último, o PSM é uma visão do sistema que possuí características, elementos e informações da tecnologia que será empregada [22].

Portanto, para analisar se o MDA como método de desenvolvimento auxilia na implementação do MPS.BR foram elaboradas duas hipóteses. Estas hipóteses são:

H1: O processo de implementação do MPS.BR nível G, com a adoção do MDA como método de desenvolvimento, auxilia na implementação do processo MPS.BR.

H2: A adoção do MDA como metodologia de desenvolvimento, contribui para antecipar o alcance dos resultados esperados de determinados processos do MPS.BR.

1.1 Objetivos

O objetivo consiste em desenvolver e analisar uma instância de MDA para auxiliar a implementação dos processos do MPS.BR

Para efetuar a análise serão realizadas as seguintes etapas:

- Elaboração de uma instância MDA para uma arquitetura *Model-View-Controller*-MVC:
  - Estabelecer regras e diretrizes para a construção dos modelos CIM;
  - Estabelecer regras e diretrizes para o mapeamento dos modelos de CIM para PIM;
- Aplicação das regras e conceitos MDA em um problema real;
- Análise de processos MPS.BR nível G, com os artefatos gerados pela adoção da instância MDA.
- Identificar artefatos e conceitos MDA que podem ser empregados para demonstrar as evidências de qualidade requisitadas pelo MPS.BR.
- Averiguar com especialistas de implantação de MPS.BR se os artefatos gerados na proposta deste trabalho atende aos quesitos do MPS.BR.

1.2 Metodologia

A metodologia utilizada para o desenvolvimento deste trabalho está descrita a seguir:

Neste trabalho será elaborada uma instância de MDA para a compreensão dos artefatos gerados e, assim, realizar o relacionamento com o modelo de referência MPS.BR. Para isso, primeiro serão estabelecidos os diagramas que irão compor o CIM, sendo adotados os diagramas de caso de uso, cenário de caso de uso e a interface de tela. Após os diagramas definidos, será criada uma sintaxe BNF- *Backus-Naur Form*. Essa sintaxe visa delimitar o tamanho e a composição das frases que integram os cenários dos casos de uso.

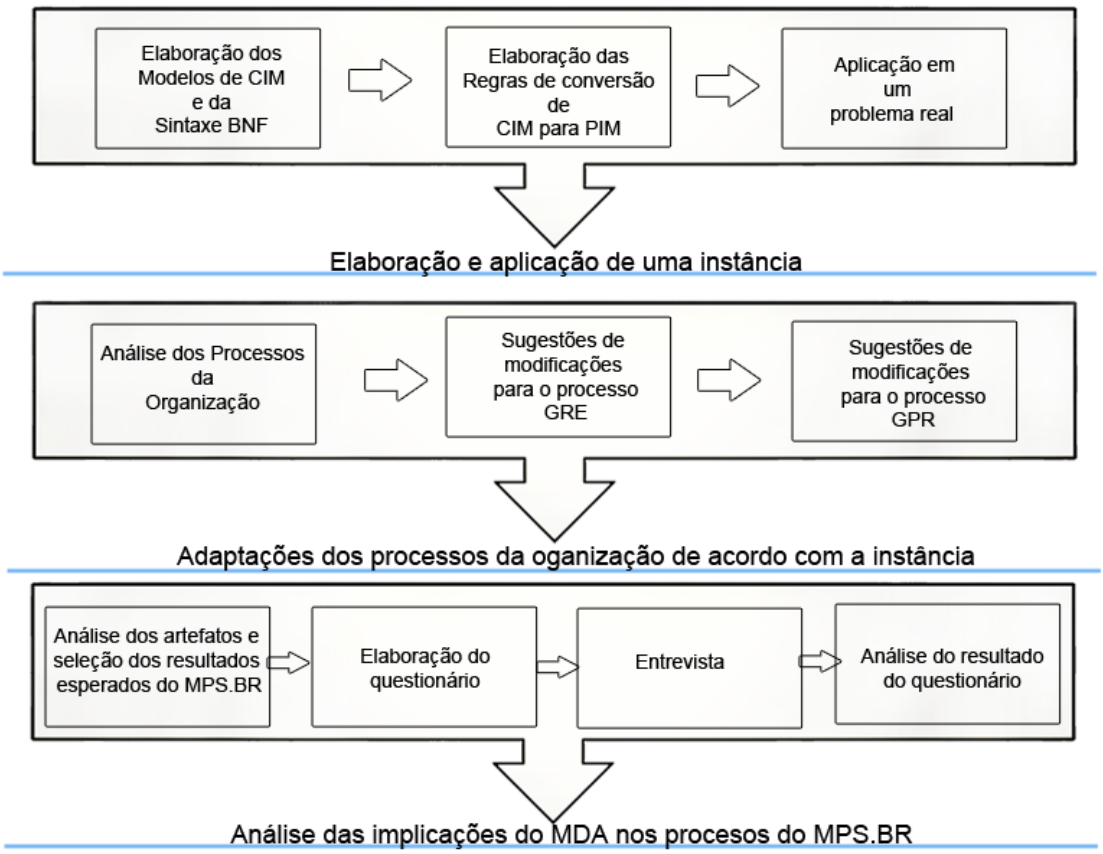
Com os diagramas de CIM definidos e com o BNF estabelecido, serão criadas um conjunto de regras de mapeamento voltados para a arquitetura MVC, as quais possibilitam a extração de informação para à conversão dos modelos CIM para os diagramas de classe e sequência que compõem o PIM. Para realizar uma aplicação desta instância é apresentado um problema real, em que serão aplicados os diagramas e regras de conversões de CIM para PIM.

Uma análise de um processo nível G do MPS.BR inserindo os conceitos da instância elaborada, juntamente com conceitos do MDA também serão realizados neste trabalho. Primeiro analisar-se-á todos os processos do nível G de uma empresa já certificada MPS.BR. Os processos analisados são referentes à Gerência de Projetos (GPR) e à Gerência de Requisitos (GRE). Após essa análise, serão sugeridas modificações em determinados pontos dos processos, os quais seriam necessários os conceitos do MDA para a adoção do mesmo. Essas sugestões serão embasadas nos artefatos gerados pela instância criada e nos conceitos do MDA.

Com o propósito de avaliar se o MDA poderá trazer benefícios para o MPS.BR serão realizadas entrevistas com profissionais da área de qualidade de software. Para estas entrevistas, é elaborado um questionário divido em duas partes. Na primeira parte são apresentadas questões referentes aos processos que sofreram as modificações para a incorporação da instância elaborada. Já a segunda parte do questionário, apresenta questões sobre se os artefatos da instância elaborada e conceitos do MDA podem antecipar quesitos de qualidade do MPS.BR.

A metodologia descrita anteriormente está apresentada na Figura 1. Esta apresenta todos os passos desenvolvidos para a elaboração do trabalho proposto.

Figura 1 – Metodologia. Fonte: do autor.



1.3 Organização/Estrutura do Trabalho

Neste trabalho buscou-se analisar o modelo de desenvolvimento MDA com o modelo de qualidade MPS.BR. Esta análise foi realizada a partir de avaliações feitas em processos prontos do modelo de referência brasileiro cedidos por uma organização, sendo sugeridos conceitos do MDA em determinados pontos no processo. Outra abordagem realizada por este trabalho é analisar, a partir dos artefatos gerados pelo MDA, o quão o MDA como metodologia de desenvolvimento, auxilia na implementação do MPS.BR. O restante deste documento está organizado da seguinte forma:

- No Capítulo 2 é realizada uma revisão bibliográfica sobre os assuntos bases desta dissertação, são esses: Melhoria do processo de Software, *Model Driven Development* (MDD), *Model driven Architecture* (MDA) e MVC. Neste capítulo também são apresentados os trabalhos relacionados sobre os assuntos referentes ao tema e ao problema central desta dissertação, para identificar o atual estado da arte.
- No Capítulo 3 é desenvolvida uma instância de MDA, no qual é elaborada uma sintaxe para o desenvolvimento dos diagramas da fase CIM, além da elaboração de regras para conversão das fases CIM para PIM. Também é apresentado um exemplo, o qual é aplicado as premissas do MDA, as fases CIM e PIM.
- No capítulo 4 são analisados os processos referentes ao nível G do MPS.BR. Assim serão sugeridas modificações em pontos nos processos para que eles atendam a instância elaborada e os conceitos do MDA.
- No Capítulo 5, é apresentado o resultado do questionário aplicado, juntamente com as análises das hipóteses.
- No Capítulo 6 são apresentadas as considerações finais, contribuições e perspectivas de trabalhos futuros.

2 REFERENCIAL TEÓRICO

Neste capítulo serão expostos assuntos que são base para a construção e entendimento desta dissertação. O primeiro conteúdo abordado será a melhoria do processo de software, que aborda a qualidade do processo de desenvolvimento de software, sendo que irá tratar dos modelos de referências, dando ênfase ao modelo brasileiro, o MPS.BR. O segundo conteúdo abordado é o desenvolvimento orientado a modelo (MDD) com seus conceitos, principais vantagens e desvantagens. Em seguida será apresentada uma abordagem do MDD, chamada MDA, na qual serão descritas suas características, especificações, etapas e linguagens. Na sequência, será abordada a arquitetura Model-View-Controller (MVC). Por último, neste capítulo serão apresentados os trabalhos relacionados sobre os assuntos referentes ao tema e ao problema central desta dissertação.

2.1 Melhoria do Processo de Software (SPI- *Software Process Improvement*)

O desenvolvimento de software vai muito além de apenas criar ferramentas e linguagens de programação eficazes. O desenvolvimento de software é algo complexo, que exige um esforço coletivo, sendo um processo dependente de diversas pessoas e procedimentos de uma organização. Para garantir a qualidade de um software, uma padronização de procedimentos deve ser proposta, ou seja, instaurar processos que podem ser definidos como conjuntos coerentes de políticas, estruturas organizacionais, tecnologias, procedimentos e artefatos necessários para idealização, elaboração, implantação e manutenção de um produto de software [23].

De acordo com o trabalho de Unterkalmsteiner [24], a qualidade do software está diretamente ligada à qualidade do processo. Portanto, diversos SPI foram criados com o intuito de trazer uma padronização para os processos. O objetivo do SPI é aumentar a qualidade dos produtos, além de reduzir tempo e custo de desenvolvimento. ISO/IEC 12207, ISO/IEC 15504, CMMI e MPS.BR são exemplos de alguns SPI presentes no mercado.

O ISO/IEC 12207 é um padrão internacional que define processos e atividades para o desenvolvimento de software, sendo que está associado com todo o processo de ciclo de vida, desde a sua concepção até o produto final [25]. No entanto, o ISO/IEC 12207 não determina como as atividades e tarefas devem ser realizadas. Dessa forma, sua estrutura é adaptável [5].

O ISO/IEC 15504, também conhecido como SPICE, foi criado originalmente como complemento do ISO/IEC 12207 e tem como objetivo orientar a avaliação e a auto avaliação da capacidade de empresas em processos e, a partir dessa avaliação, permitir a melho-

ria dos processos [6]. Esse modelo de qualidade de software estabelece um " *framework*" que é utilizado tanto para a criação do método de avaliação como para a melhoria dos processos de software [26].

Capability Maturity Model Integration(CMMI) é uma abordagem que visa a melhoria dos processos da organização através do seu modelo de maturidade. Seu objetivo principal é reduzir custos de implementação das melhorias nos processos, eliminando inconsistências, além de determinar diretrizes para auxiliar as organizações em diversos estágios de um projeto de software [27] . Estruturado em 5 níveis de maturidade, o CMMI possui 22 áreas de processos, sendo compatível com outros padrões e modelos de qualidade, como CMM, ISO/IEC 12207 e ISO/IEC 15504 [4].

O modelo de referência brasileiro, o MPS.BR foi criado para a realidade do país, sendo voltado principalmente para pequenas e médias empresas. Visto que o MPS.BR é o modelo empregado para a elaboração desta dissertação, este será abordado na próxima subseção.

2.1.1 MPS.BR

O Programa de Melhoria de Processo do Software Brasileiro, também conhecido pela sigla MPS.BR, visa o aperfeiçoamento da capacidade de desenvolvimento de software nas empresas brasileiras. Ele foi desenvolvido pela SOFTEX em 2003, com parceria entre governo federal e academia. O modelo brasileiro é independente, porém é compatível com as Normas ISO 12207 e 15504, bem como o CMMI.

O MPS.BR é um programa de melhoria que tem por objetivo aperfeiçoar a capacidade de desenvolvimento de software, de serviços e de práticas de recursos humanos. Para isso o MPS.BR oferece três tipos de modelos de referência: o de desenvolvimento de software – MPS-SW, o de serviços- MPS-SV e o de recursos humanos- MPS-RH. Neste trabalho será utilizado o modelo de desenvolvimento de software e, para se referir a este modelo, será utilizado a sigla MPS.BR.

O guia do MPS.BR é baseado em níveis de maturidade, sendo que o nível de maturidade é um grau de melhoria para um predeterminado conjunto de processos, os quais todos os resultados esperados e atributos dos processos são satisfeitos[3].

O MPS.BR é composto por 19 processos que são distribuídos e implementados ao longo dos seus sete níveis de maturidade, os quais são denominados pelas letras de G a A, sendo que o G é o nível inicial e progride até o A. Cada nível herda os processos de níveis anteriores. Assim, o nível A possuirá todos os processos proposto pelo MPS.BR. A Tabela 1 apresenta os níveis de maturidade com os seus respectivos processos.

Tabela 1 – Processos do MPS.BR [3]

Nível	Processos
A	
B	Gerência de Projetos – GPR (evolução)
C	Desenvolvimento para Reutilização- DRU Gerência de Decisões- GDE Gerência de Risco- GRI
D	Desenvolvimento de Requisitos- DRE Integração do Produto- ITP Projeto e Construção do Produto- PCP Validação- VAL Verificação- VER
E	Gerência de Projetos – GPR (evolução) Avaliação e Melhoria do Processo Organizacional- AMP Definição do Processo Organizacional- DFP Gerência de Recursos Humanos- GRH Gerência de Reutilização- GRU
F	Medição- MED Garantia de Qualidade -GQA Gerência de Configuração- GCO Aquisição AQU Gerência de Portfólio de Projeto- GPP
G	Gerência de Requisitos- GRE Gerência de Projetos- GPR

Nota-se que na Tabela 1 os níveis A e B não possuem processos específicos. Esses níveis buscam a melhoria contínua dos processos já instaurados.

Cada processo contém resultados esperados, os quais são a especificação do que deve ser executado para o cumprimento do processo determinado. Estes resultados podem ser comprovados por um produto de trabalho produzido ou uma mudança significativa de estado ao se executar o processo. Neste trabalho, os resultados esperados serão analisados para verificar se a instância de MDA auxilia nos processos do MPS.BR.

2.1.1.1 Dificuldades Encontradas na Implantação e Manutenção do MPS.BR

Nesta seção serão apresentadas as dificuldades mencionadas na literatura relacionadas com a implantação e manutenção do MPS.BR. Essas dificuldades são relatadas em trabalhos e mostram obstáculos encontradas na incorporação de devidos processos do modelo de referência e a manutenção dos mesmos, com o decorrer do tempo. Serão exibidas duas tabelas: na Tabela 2 serão apresentadas as dificuldades identificadas de acordo com os processos do MPS.BR e na Tabela 3 as dificuldades relatadas são sobre os impasses encontrados como um todo na organização, com a implantação dos novos processos do MPS.BR.

Tabela 2 – Dificuldades da Implantação e Manutenção do MPS.BR

Dificuldades do MPS.BR	Processo e/ou Nível
Gerência de Configuração precisa de maior correção e crítica às ferramentas usadas [13]	GCO-Nível F
Métrica precisa de maior correção e crítica às ferramentas usadas [13]	MED-Nível F
Documentação Excessiva [13] [9]	Nível G
Muita subjetividade do modelo [13]	Nível G
O modelo não pensa muita na produtividade e deveria propor soluções que favoreçam isso [13]	Nível G
Muito "o que deve ser feito", porém não tem "como deve ser feito" [13]	Nível G
Somente com o que estava no Guia não foi possível implementar o processo, houve a necessidade de um equipe de consultoria [13]	Nível G
Dificuldade em ter um método de reutilização que não interfira no cotidiano e de fácil integração com outros processos [28]	GRU- Nível E
Difícil Identificação de Métricas úteis à monitoração e controle de processo [28] [29]	GRU-Nível E
Dificuldade em ter e manter uma base de dados históricos -GPR4 e GPR 9 [16]	GPR- Nível G
Dificuldade em fazer a rastreabilidade bidirecional dos requisitos - Falta de ferramenta para esta tarefa - GRE3 [16]	GRE- Nível G
Dificuldade no gerenciamento das mudanças nos requisitos – GRE5 [16]	GRE- Nível G
Comunicação e *feedback* do andamento do MPS.BR [15]	-
MPS.BR como burocratização (obstáculo ao "trabalho real") [15]	-
Dificuldade de definir uma estratégia não intrusiva, que não tivesse impacto no cotidiano dos colaboradores e que facilitasse a integração com outros processos [29]	GRU- Nível E
Busca por ativos reutilizáveis [29]	GRU- Nível E

A Tabela 2 apresentou dificuldades com relação ao MPS.BR e sua implementação e manutenção relatadas por organizações. Muitas dessas dificuldades não estão ligadas diretamente com os processos do MPS.BR, mas sim com o modelo e sua subjetividade.

Tabela 3 – Dificuldades Relatadas no Âmbito da Empresa com a Implantação do MPS.BR

Dificuldades Relatadas do MPS.BR
Resistência por parte dos colaboradores, gerentes e analistas de requisitos e todos da organização [13], [17], [15], [30] e [31]
Obrigatoriedade de treinamento de todos os colaboradores da empresa durante o processo de implantação [13]
Com a pressão do dia-a-dia muita coisa acaba não sendo executada como deveria [13]
Cultura da empresa é um obstáculo (Mudança Cultural)(Políticas Organizacionais) [32], [17], [33] e [30]
Falta de investimento prejudica a implantação [32] e [30]
Falta de recursos humanos prejudica o processo de implantação [32] e [31]
Pouco conhecimento de Engenharia de software [17]
Dificuldade na Introdução de novas tecnologias de desenvolvimento simultâneo à adoção do processo padrão [17]
Alta rotatividade de equipe [17], [9] e [15]
Ausência de ferramentas de apoio à execução dos processos [17]
Gerência de um programa de melhoria [33]
Entendimento sobre a melhoria de processos e conceitos relacionados (requisitos, separar processos ...) [34] e [9]
Experiência e Qualificação [15]
Pouco envolvimento da equipe [15]

A Tabela 3 apresentou a dificuldade com relação a organização e sua administração da implementação e manutenção do MPS.BR. Pode-se extrair desta tabela que as maiores dificuldades encontradas são: resistência à mudanças por parte dos colaboradores; cultura da empresa e alta rotatividade da equipe.

2.2 *Model Driven Development*- MDD

O *Model Driven Development* (MDD) não utiliza modelos como documentação. Para o MDD, os modelos se tornam artefatos essenciais do processo de desenvolvimento, pois através das transformação executadas sobre eles é que se obtém o código [18].

Portanto, o MDD visa encontrar abstrações de domínios específicos e torná-las acessíveis através da modelagem formal. Com isso, possibilita um crescimento para a automação da produção de software. Isso implica diretamente no aumento da produtividade, melhorando também a qualidade do sistema desenvolvido [35] e [18].

A abordagem de desenvolvimento MDD possui diversas vantagens. Kleppe, Warmer e Bast [1], Stahl e Völter [18], Mernik, Heering e Sloane [19] e Bahnot et al. [20] destacam as seguintes vantagens do desenvolvimento orientado a modelo :

- Produtividade: a produtividade tende a aumentar, pois em vez de dedicar tempo no

desenvolvimento de código, este será dispendido na elaboração de modelos de mais alto nível, os quais um único modelo pode gerar uma grande quantidade de códigos. As tarefas repetitivas são implementadas através da transformações automáticas.

- Portabilidade: um modelo de alto nível pode ser transformado em código para diferentes plataformas.

- Manutenção e Documentação: no MDD qualquer modificação é realizada nos modelos, mantendo assim a compatibilidade com o código e a documentação sempre atualizada.

- Reutilização: para a reutilização no MDD, os modelos devem passar pelas as adaptações necessárias e gerar o código automaticamente.

- Corretude: os erros conceituais são identificados em um nível mais alto de abstração.

No entanto, o MDD também traz algumas dificuldades a serem superadas. Ambler [36], Hailpern e Tarr [37] e Thomas [38] citam as seguintes desvantagens do desenvolvimento orientado a modelo:

- Rigidez: a rigidez referente ao MDD ocorre porque boa parte do código gerado não é de alcance do desenvolvedor.

- Complexidade: ferramentas de modelagens, geradores de código e as transformações agregam uma maior complexidade ao processo de desenvolvimento. Outro fator que aumenta a complexidade é a quantidade de modelos relacionados, pois quanto mais modelos relacionados, maior a complexidade de ligação entre os artefatos.

- Desempenho: a utilização de geradores automatizados tendem a produzir muito código desnecessário. Isso pode influenciar negativamente o desempenho do código.

- Curva de Aprendizagem: para a construção de modelos, utilização de ferramentas e, entre outras atividades do MDD, exige-se profissionais com essas habilidades. Apesar dessas técnicas não serem de difícil aprendizagem, podem requerer algum treinamento.

Uma abordagem do MDD, que é definida pela OMG, é o *Model Driven Architecture* (MDA). O MDA define um conceito especial de modelos, que distinguem aqueles modelos que levam em conta os detalhes do hardware e software subjacente (plataforma) e aqueles que não [37].

2.3 *Model Driven Architecture*- MDA

O *Model Driven Architecture*- MDA foi lançado pela OMG em 2001 e é uma especificação para apoiar o *Model Driven Development* - MDD, sendo que seu principal objetivo é a utilização de modelos em todo o processo para o desenvolvimento de software. O MDA tem como característica as transformações entre modelos. A automatização da transformação de modelos proporciona uma série de vantagens, citando a redução de custos, tempo, riscos de produção e manutenção de um sistema [22]. No entanto, nem todos os modelos são adequados para as transformações automáticas. Um modelo precisa ser suficientemente completo e preciso para que as informações sobre o sistema em desenvolvimento sejam bem especificadas pelos modelos. De acordo com a OMG, existem vários procedimentos para agregar valor aos modelos [22]:

- Modelos como meios de comunicação: os modelos e a modelagem podem facilitar a comunicação e o entendimento, sendo que a padronização do MDA auxilia na determinação de termos bem definidos, ícones e notação, os quais amparam o entendimento comum de uma área e no fornecimento da base para modelos. Outro ponto a ser considerado é que os modelos tornam-se parte da memória corporativa da empresa, pois capturam a essência da empresa bem como os requisitos, processos, informações e serviços.

- Derivação através de Transformações Automatizadas: a automação reduz tempo e o custo da realização de um projeto e também de manutenção, sendo que com a automação, mudanças e manutenção tem uma garantia maior de consistência de todos os artefatos que foram derivados. A derivação automática de artefatos de software envolve um modelo de entrada ("*source model*"), a transformação e um modelo final ("*Target Artifact*"), sendo que o modelo final pode ser outro modelo.

- Análise de modelos: a análise feita sobre modelos pode auxiliar em tomadas de decisões, monitoramento e avaliações sobre a qualidade. Alguma das análises possíveis sobre modelos são: métricas, estatísticas e validação..

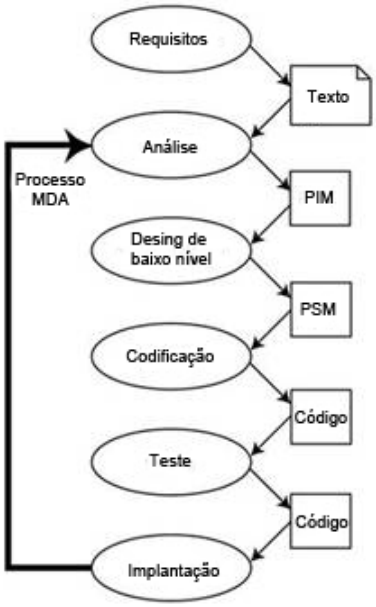
- Simulação e execução de modelos: os modelos podem ser submetidos a mecanismos de simulação e execução, permitindo assim um melhor entendimento sobre as funcionalidades de um sistema, sendo também uma forma de validação dos próprios modelos.

- Obtenção de informações de modelos: modelos bem definidos podem ser usados para a derivação de outras informações, como por exemplo, a partir das informações extraídas de um modelo é possível obter: documentação, especificações de aquisição, idéias derivadas, *playbooks* de processos ou até mesmo sistemas de software.

Para realizar um processo de MDA não é necessário hardware específico. Porém, existe a necessidade de softwares relevantes. Existem diversos softwares disponíveis no mercado que abordam esta proposta, mas nem todos podem ser considerados eficientes, sendo que algumas proposta são apenas parciais, não abordando assim o processo completo de desenvolvimento do projeto [39].

O ciclo de vida de desenvolvimento do MDA está apresentado Figura 2. Nota-se que não difere-se muito de um ciclo de vida de desenvolvimento de software comum, sendo que uma grande diferença encontra-se na natureza dos artefatos, que neste caso são criados durante o processo de desenvolvimento [1].

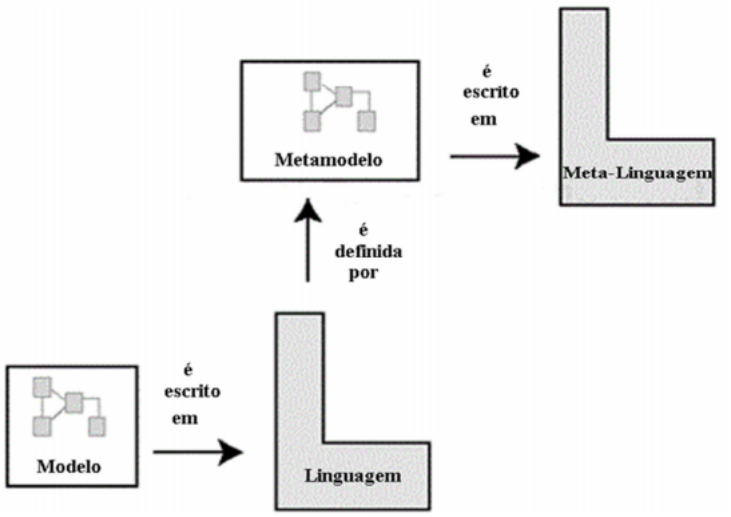
Figura 2 – Ciclo de vida de desenvolvimento de um MDA. Fonte [1]



Um modelo para o contexto do MDA é um conjunto de informações que representam aspectos de um sistema baseado em um conjunto específico de interesses. Um modelo deve possuir informações sobre um sistema dentro de um escopo estabelecido, regras de integridade que se aplicam a esse sistema e o significado dos termos usados [22].

Há certa circularidade para modelos e linguagens de modelagem. Uma linguagem de modelagem pode ser expressa como modelo, isso é denotado por metamodelo. O metamodelo é um modelo que define uma linguagem de modelagem e também é expresso usando uma linguagem de modelagem[22]. Essa linguagem de modelagem, também chamada de metalinguagem, é descrita por uma metalinguagem. A Figura 3 apresenta esta relação de modelo, linguagem e metamodelo.

Figura 3 – Relação entre Modelos e Metamodelos. Fonte [1]



2.3.1 Etapas do desenvolvimento do MDA

No desenvolvimento de um MDA, o desenvolvimento de modelos e as transformações entre eles são pontos primordiais. No MDA são estabelecidas três categorias de modelos. São eles:

Modelo Independente de Computação- CIM

O Modelo Independente de Computação (*Computation Independent Model- CIM*), também chamado de modelo de negócios [40], apresenta uma visão do sistema. Com isso, auxilia no entendimento do que é esperado que o sistema execute, sendo que este tipo de modelo é independente do conhecimento em computação, tendo um alto nível de abstração[41].

Modelo Independente de Plataforma- PIM

O Modelo Independente de Plataforma (*Plataform Independent Model - PIM*) descreve um sistema sem qualquer conhecimento da plataforma final de execução. O PIM possui um maior nível de abstração do que o PSM, no entanto, menor que o CIM, e representa uma especificação formal da estrutura e do funcionamento do sistema. Como oculta as características de uma plataforma em particular, possibilita o reaproveitamento em diferentes plataformas[41] e [42].

Modelo Específico de Plataforma- PSM

O Modelo Específico de Plataforma (*Platform Specific Model - PSM*) é criado a partir do PIM e, neste modelo, são considerados detalhes da plataforma específica, sendo a base para a transformação do modelo para código [41] e [42].

Com os modelos apresentados anteriormente é possível executar transformações, que podem ser compreendidas como um conjunto de técnicas e regras aplicadas em um

modelo, resultando assim em um outro modelo com as características desejadas. As transformações consideradas pelo MDA são [41]:

- CIM para PIM: esta transformação normalmente é executada de forma manual [43].
- PIM para PIM: transformações de PIM para PIM são para a mudanças pretendidas no modelo, ou, até mesmo, simplificação sem a necessidade de se preocupar com a plataforma.
- PIM para PSM: transformação padrão prevista pelo MDA, a qual o PSM combina as especificações do modelo PIM, com detalhes que especificam como um sistema usa um determinado tipo de plataforma.
- PSM para PSM: esta transformação permite a migração de modelos para plataformas diferentes, sendo que pode ser utilizada também para evolução dos modelos.
- PSM para código: transformação automática normalmente executada por uma ferramenta específica.

Para um modelo estar apto as transformações, ele precisa utilizar uma linguagem de modelagem bem definida que obedeça a um conjunto de regras formais. O MOF (*Meta Object Facility*) é uma linguagem para descrição de metamodelos, que define regras sobre como os modelos de uma aplicação devem ser escritos para serem considerados bem definidos [44].

2.3.2 Meta Object Facility- MOF

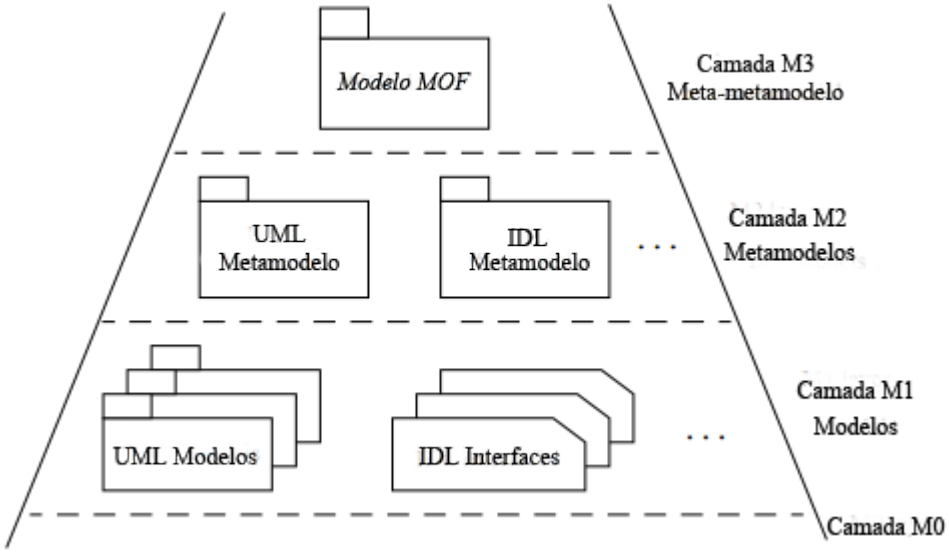
O *Meta Object Facility* - MOF é uma especificação criada pela OMG, a qual define uma linguagem abstrata e um *framework* para criação de um padrão de comportamento e a implementação de repositórios para metamodelos e metadados respectivamente [45]. Como no MDA os modelos devem ser criados em uma linguagem única, o MOF é utilizado para esta definição.

O MOF serve de base para diversas outras especificações. Destaca-se a linguagem de modelagem UML, o XMI (XML Metadata Interchange) e, em última instância, o próprio MDA.

O XMI define um mapeamento de tecnologia de metamodelos MOF para documentos XML. Esse mapeamento pode ser utilizado para definir um formato de troca de metadados em conformidade com um determinado metamodelo MOF. O uso mais difundido é na troca facilitada de metadados entre as ferramentas de modelagem UML.

A arquitetura do MOF, com suas camadas, está ilustrada na Figura 4 [45].

Figura 4 – Níveis do MOF. Fonte [1]



Na Figura 4, a camada M1 possuí os modelos, na camada M2 apresenta os meta-modelos e na camada M3 os meta-metamodelos ou o MOF propriamente dito. Assim, o MOF define os metamodelos e estes definem os modelos.

Algumas características importantes da arquitetura do MOF são [45]:

- O modelo MOF é orientado a objeto, sendo que possui elementos construtivos que estão alinhados com os do UML. Com isso, é possível que o MOF seja descrito a partir dos elementos do UML.
- Os quatro níveis apresentados na Figura 4 não são fixos, sendo que pode haver mais ou menos níveis (isso vai depender de como é implementado o MOF). Os níveis MOF são unicamente convenções para uma melhor compreensão da relação entre os diferentes tipos de dados e metadados.
- Um modelo baseado em MOF não precisa necessariamente estar limitado a um nível, sendo possível estar em mais de um nível, de acordo com a necessidade.
- O modelo MOF é auto-descritivo (isso quer dizer que é definido de acordo com suas próprias metamodelagem).

Para um melhor entendimento das relações das camadas apresentadas na Figura 4, a seguir serão relatados as quatros camadas M0, M1, M2 e M3 [1].

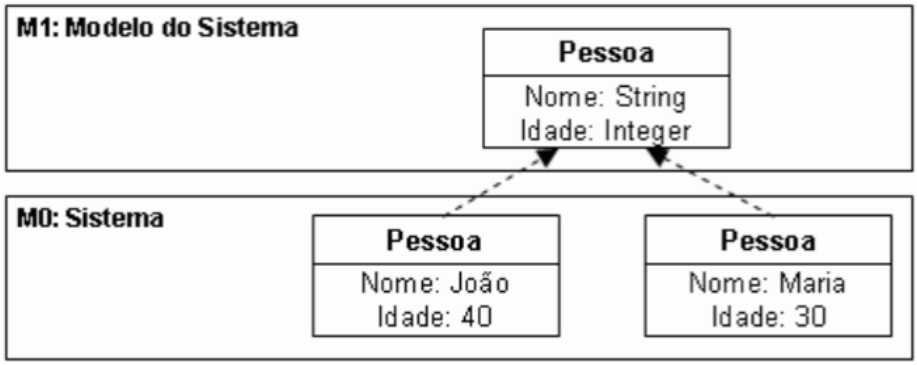
Camada M0 - Instâncias

Esta camada trata das instâncias reais, são informações para a composição do software. Esta camada é composta por dados e objetos. Por exemplo: nesta camada está contido o nome do consumidor/cliente, produtos que foram consumidos e etc.

Camada M1- Modelo do Sistema

O segundo nível refere-se a modelos que descrevem os dados reais do sistema. Estes dados são derivados da camada M0. Logo, a camada M1 faz uma categorização em um nível mais elevado de abstração das instâncias contidas na camada M0. Esta relação está presente na Figura 5.

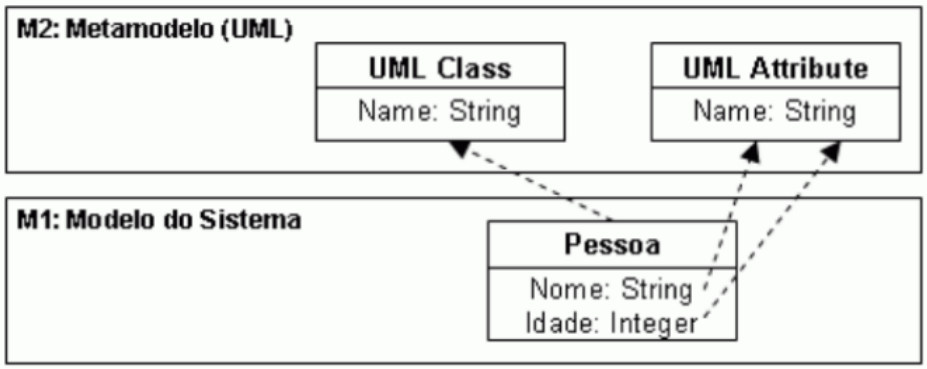
Figura 5 – Relação entre as Camadas M0 e M1. Fonte [1]



Camada M2- Modelo de Modelo

O modelo que reside na camada M2 é chamado de metamodelo que especifica como categorizar cada elemento existente no modelo da camada M1, sendo que cada elemento desse, é uma instância de um elemento em M2 e cada elemento em M2 categoriza elementos em M1. A mesma relação que está presente entre os elementos de camadas M0 e M1 existe entre os elementos de M1 e M2. A Figura 6 expõe esta relação.

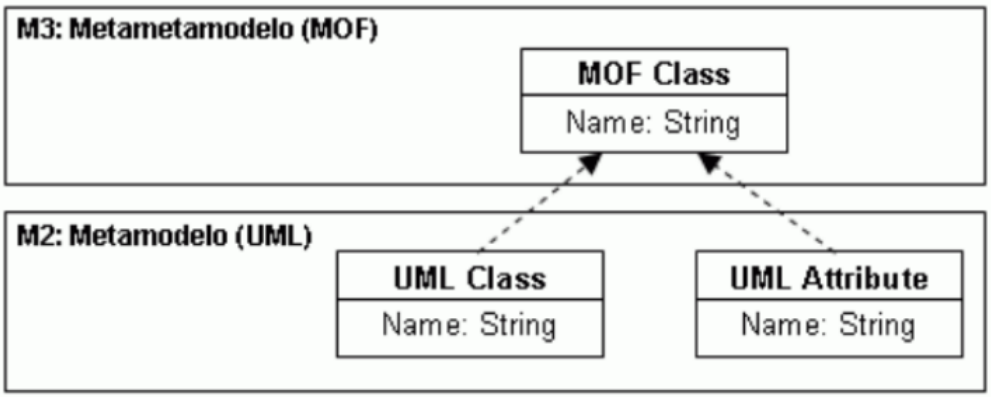
Figura 6 – Relação entre as camadas M1 e M2. Fonte [1]



Camada M3- Meta-metamodelo

A última camada compreende o meta-metamodelo, o qual define a linguagem aplicada para definição dos metamodelos contidos na camada M2. A mesma relação que está presente entre os elementos das camadas M0 e M1 e elementos de camadas M1 e M2 existe entre os elementos de M2 e M3. A especificação do MOF está dentro de M3, sendo que todas as linguagens de modelagem (como UML, CWM, e etc) são instâncias do MOF. O relacionamento entre as camadas M2 e M3 está apresentado na Figura 7.

Figura 7 – Relação entre as camadas M2 e M3. Fonte [1]

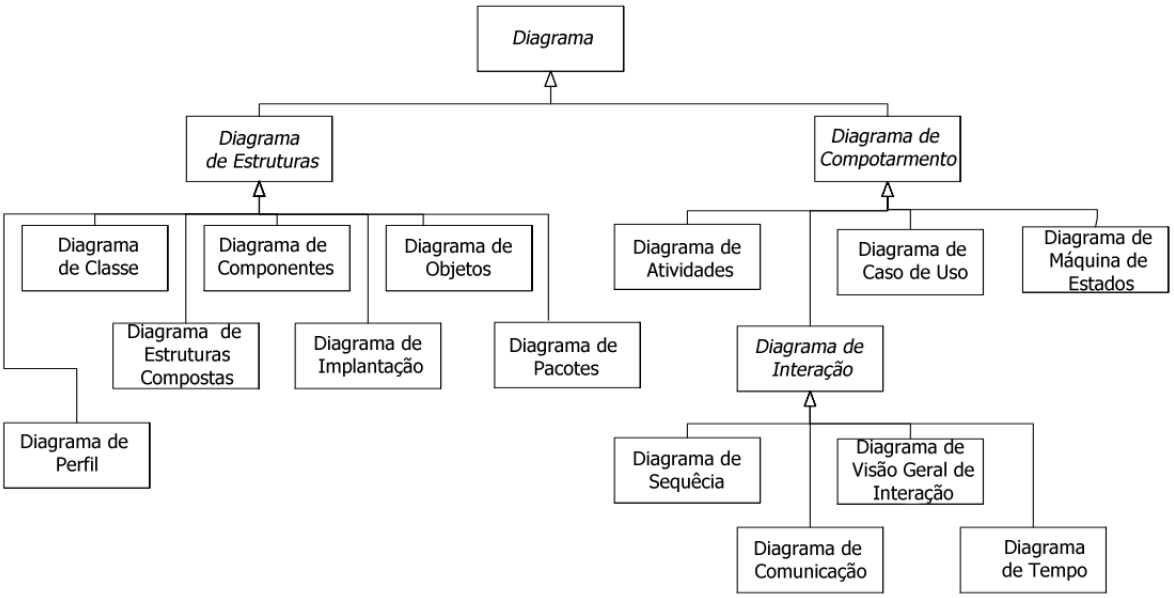


2.3.3 Unified Modeling Language-UML

A *Unified Modeling Language* - UML é uma linguagem visual utilizada para modelar softwares baseados no paradigma orientado a modelo. Criado em 1996, a UML surgiu da união de três métodos de modelagem: o método de Booch, o método OMT (*Object Modeling Technique*) de Jacobson e o método OOSE (*Object-Oriented Software Engineering*) de Rumbaugh. Em 1997, a UML foi adotada pela OMG como linguagem padrão de modelagem [46]. Atualmente, a UML se encontra na versão 2.5.

De acordo com a OMG existem dois principais tipos de diagrama: diagramas estruturais e diagramas comportamentais . Os diagramas estruturais representam a estrutura estáticas de objetos em um sistema, ou seja, a especificação dos elementos são independente do tempo. Os diagramas comportamentais demonstram o comportamento dinâmico dos objetos em um sistema, incluindo seus métodos, atividades, colaboração e histórias dos estados. Uma das possibilidades de mostrar o comportamento dinâmico de um sistema é através da descrição de uma série de alterações do sistemas ao longo do tempo [2]. A Figura 8 apresenta os diagramas UML classificados em estruturais e comportamentais .

Figura 8 – Diagramas da UML. Fonte [2]



Neste trabalho serão utilizados um sub-conjunto dos diagramas UML. Os diagramas utilizados serão: diagrama de caso de uso; diagrama de classe e diagrama de sequência. Os elementos apresentados destes diagramas serão utilizados para compor as regras de transformações de CIM para PIM do Capítulo 3 desta dissertação.

2.3.3.1 Diagrama de Classes

O diagrama de classes é um diagrama estrutural que caracteriza classes, interfaces e suas associações, além de determinar os métodos e atributos de cada classe [47]. De acordo com Guedes [46], o diagrama de classes é um dos mais importantes e mais utilizados da UML. A seguir serão apresentados os principais elementos que estão presentes neste diagrama, de acordo com a OMG [2]:

Abstração: a abstração traz o conceito de relacionamento entre dois ou mais elementos que representam o mesmo conceito em níveis diferentes de abstração.

Agregação: indica que a propriedade é agregada de forma compositiva, ou seja, o objeto composto tem a responsabilidade pela existência e armazenamento dos objetos compostos.

Associação: a associação caracteriza um relacionamento semântico que pode acontecer entre instâncias tipadas. Devem haver pelo menos duas extremidades representada por propriedades, cada uma está ligada ao tipo final.

Classe: a classe representa um conjunto de objetos que compartilham das mesmas especificações de características, restrições e semântica.

Classificador: classificador é uma classificação de instâncias, que descreve um

conjunto de instâncias que possuem características em comum.

Restrição: a restrição é uma limitação ou condição que é imposta a um elemento, sendo que esta restrição/condição pode ser expressada em linguagem natural ou linguagem de máquina.

Dependência: uma dependência indica um relacionamento cujo um ou mais elementos do modelo requer outros elementos do modelo para sua aplicação ou especificação.

Elemento: um elemento é um constituinte de um modelo, sendo que pode ainda possuir outros elementos.

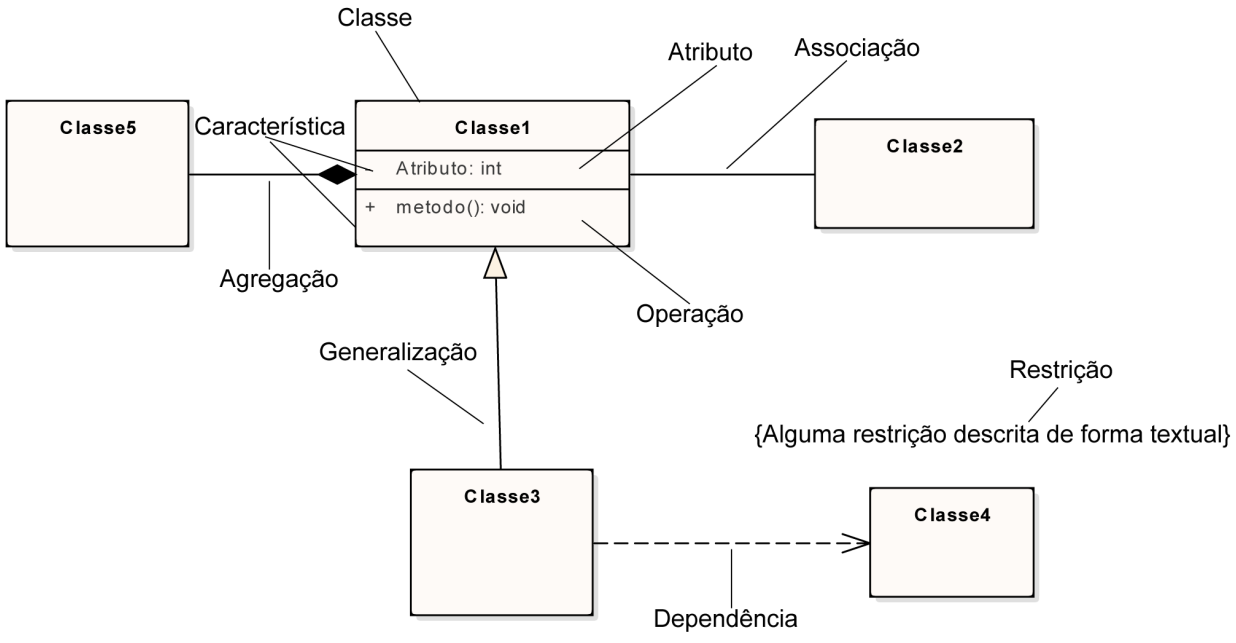
Característica: uma característica considera uma particularidade comportamental ou estrutural de uma instância de um classificador.

Generalização: uma generalização é um relacionamento sistemático entre um classificador mais geral e um classificador mais específico. Cada instância do classificador específico é também uma instância do classificador geral. Assim, o classificador específico herda as características do classificador mais geral.

Parâmetro: um parâmetro é uma especificação de um argumento utilizado para passar informações dentro ou fora de um invocação de um recurso comportamental.

A Figura 9 apresenta um sub-conjunto dos elementos do diagrama de classe descritos anteriormente.

Figura 9 – Principais elementos do Diagrama de Classes. Fonte: do autor.



2.3.3.2 Diagrama de Caso de Uso

O diagrama de caso de uso representa uma função do sistema e geralmente é utilizado na fase de análise de requisitos. De acordo com Guedes [46], este diagrama auxilia na identificação e compreensão dos requisitos do sistema, ajudando a especificar, visualizar e documentar as funções e serviços do sistema desejados pelo usuário. No caso de uso, uma sequência de interações entre ator e sistema é denominado fluxo, sendo que comumente um diagrama tem um fluxo normal e fluxos alternativos [48]. A seguir, são apresentados alguns dos elementos deste diagrama definidos pela OMG [2]:

Ator: um ator representa um papel desempenhado por um usuário ou outro sistema que interage com o objeto.

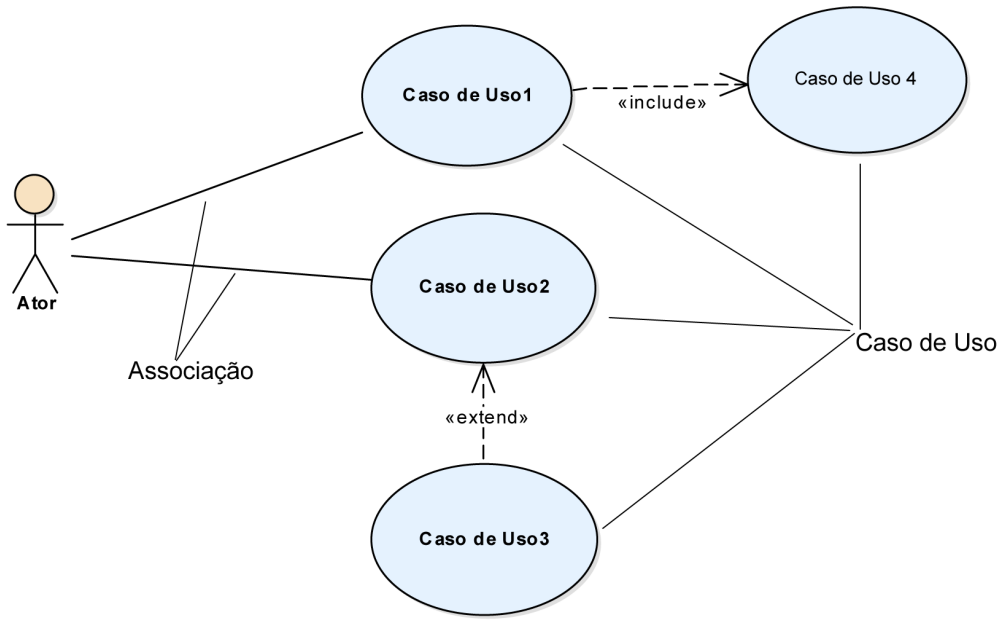
Associação: a associação representa a interação/relacionamento entre os atores do diagramas e os casos de uso. A associação é possível também entre casos de usos com outros casos de usos.

Extended : o *extended* especifica o comportamento de um caso de uso poder ser entendido de outro caso de uso. Essa extensão ocorre por um ou mais pontos específicos definidos no caso de uso que foi estendido.

Include: um *include* representa um relacionamento que define que um caso de uso possui o comportamento definido em outro caso de uso.

A Figura 10 apresenta os elementos do diagrama de caso de uso descritos anteriormente.

Figura 10 – Principais elementos do Diagrama de Caso de Uso. Fonte: do autor.



2.3.3.3 Diagrama de Sequência

O diagrama de sequência é um digrama comportamental e é usado para apresentar o comportamento dinâmico do sistema, preocupando-se com a ordem temporal em que as mensagens são trocadas entre os objetos envolvidos em um determinado processo [48] e [46]. Alguns dos elementos do diagrama de sequência definidos pela OMG são [2]:

Interação: a interação é uma unidade de comportamento que enfoca na troca de informação entre os elementos conectados.

Lifeline: uma *lifeline* representa um participante individual na interação.

Mensagem: uma mensagem consiste na comunicação entre *lifelines* na interação.

Especificação de Execução: é uma unidade de comportamento ou ação interna da *lifeline*. A sua duração é representada por duas *OccurrenceSpecifications*, uma inicial e outra final.

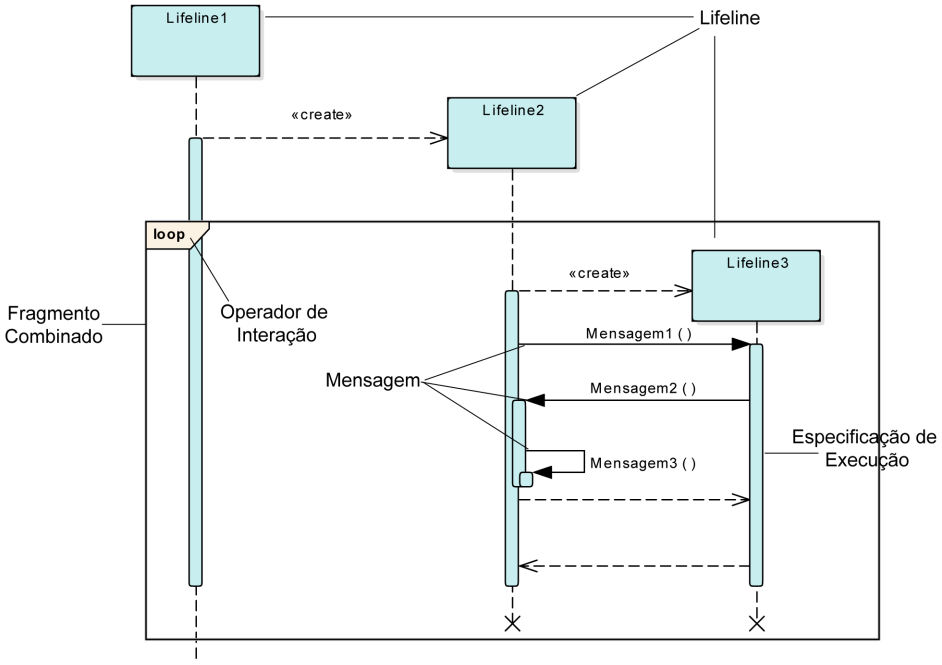
OccurrenceSpecification: é a unidade semântica básica das interações.

Fragmento Combinado: define uma expressão de fragmentos de interação. Um fragmento combinado é definido por um operador de interação e operandos de interação correspondentes.

Operador de Interação: é uma especificação que designa os diferentes tipos de operadores do fragmento combinado.

A Figura 11 apresenta os elementos do diagrama de sequência descritos anteriormente.

Figura 11 – Principais elementos do Diagrama de Sequência. Fonte: do autor.



2.3.4 Ferramentas MDA

No desenvolvimento de software baseado em MDA, as ferramentas possuem um papel importante. Uma ferramenta é capaz de partir de um modelo PIM, executar transformações e chegar no modelo PSM e/ou código. É importante ressaltar que nem todas as ferramentas executam as mesmas funcionalidades, sendo que algumas partem do PIM e já transformam-se em código, omitindo assim a fase do PSM. A escolha da ferramenta deve ser baseada nas necessidades dos desenvolvedores, ponderando as características de cada uma e seus pontos fortes e fracos [49]. De acordo com o site da OMG, existem no mercado, atualmente, mais de cinquenta ferramentas de MDA, nas quais pelo menos uma característica principal do MDA é abordada [50].

Calic [49] lista um conjunto de critérios importantes referentes às ferramentas MDA. Esses critérios servem para avaliar as ferramentas existentes e dar suporte a novas ferramentas em desenvolvimento. Alguns critérios são citados a seguir:

- Características do MDA: neste critério são analisados pontos definidos pela OMG para uma ferramenta ser compatível com MDA. Algumas características são: suporte de PIM e de PSM, perfil UML, transformações e entre outros [41].

- Recursos da Ferramenta: nesta especificação é avaliada a ferramenta de acordo com a sua capacidade e suas características. Alguns desses recursos são [51] e [49] : editor gráfico de UML, validação e verificação de modelos, testes, etc.

- Qualidade: neste critério é avaliada a qualidade da ferramenta MDA, sendo que os quesitos incluem a eficiência, a facilidade de compreensão e de execução. Algumas das características aqui abordadas são [52]: eficiência, completude, facilidade de implementação, entre outras .

- Usabilidade: neste critério é analisada a qualidade da interação entre os usuários e a ferramenta MDA. Algumas características da usabilidade são [53] e [54]: visibilidade, consistência, restrições, etc.

- Produtividade: nesta especificação é analisado se os benefícios esperados no uso do MDA são realmente alcançados. Algumas características são [55]: redução no tempo de desenvolvimento, redução da complexidade de implementação, qualidade do código e entre outros.

- Documentação: a avaliação é feita analisando a documentação da ferramenta de MDA, como por exemplo os tutoriais disponíveis, ajuda *on-line* e etc. Algumas características analisadas sobre a documentação são [52]: Organização, qualidade, facilidade de compreensão, ajuda on-line e etc.

2.4 Model-View-Controller- MVC

O MDA pode ser desenvolvido para um domínio específico [22]. Neste trabalho foi adotado o *Model-View-Controller*- MVC como padrão arquitetural. Os artigos [56], [57], [58] e [59] abordam o MDA juntamente com o MVC. Para melhor compreensão dos conceitos MVC, será apresentado uma breve descrição sobre este padrão arquitetural.

O paradigma MVC foi introduzido a partir da interface de usuário *Smalltalk*-80, desenvolvido pela Xerox na década de 80 [60]. Este padrão arquitetural separa os modelos de informação da lógica do aplicativo da interface do usuário. A estrutura MVC consiste em três tipos de objetos. O modelo é a aplicação do objeto, onde é implementada a lógica do aplicativo (domínio do problema). O *Controller* interpreta as entradas do usuário, informando o modelo e / ou a *View* para alteração, conforme apropriado. A *View* deve assegurar que a sua aparência reflete o estado do modelo. A abordagem do isolamento das unidades funcionais facilita o entendimento e as modificações, pois assim não existe a necessidade de entendimento de todas as outras unidades [61].

2.5 Trabalhos Relacionados

Nesta seção serão discutidos os trabalhos relacionados aos temas deste estudo, dando enfoque em modelos de referências e MDA. Como metodologia foi realizado uma busca sistemática baseada nas diretrizes propostas por Peterson et. al. [62].

Esta busca tem como intuito analisar o estado atual da área de pesquisa e, assim, poder responder a seguinte questão:

Q1- Qual é o cenário de pesquisas que correlacionam o MDD/MDA com os SPI's?

Com o levantamento da questão deve-se selecionar quais são as bases de dados em que serão realizadas as pesquisas. Neste trabalho, as buscas foram realizadas nas bibliotecas virtuais apresentadas na Tabela 4.

Tabela 4 – Bibliotecas Virtuais Utilizadas para o Mapeamento Sistemático

IEEE *Xplore Digital Library*	<http://ieeexplore.ieee.org>
ACM *Digital Library*	<http://dl.acm.org>
Springer	<http://www.springer.com/br/>

Após a seleção das bases de dados, foram elaboradas duas *strings* a partir das palavras chaves e objetivos deste trabalho. Optou-se por utilizar duas *strings*, pois assim houve a possibilidade de abordar nas buscas, tanto o *Model Driven Architecture* (MDA) quanto o *Model Driven Development* (MDD). Os outros termos das strings estão relacionados com modelos de qualidade. Portanto, um dos termos empregados é o MPS.BR, foco deste estudo. No entanto, este é um modelo de abrangência nacional, em vista disso, as

pesquisas em âmbito internacional são mais restritas. Logo, também empregou o CMMI, pois este é um modelo internacional, com o qual o MPS.BR possui compatibilidade. Para uma maior abrangência, e maior compreensão dos modelos de referências, foi utilizado o termo *Software Process Improvement*. Deste modo, as duas *strings* geradas para as buscas são:

1. (("Model Driven Architecture" OR "MDA") AND "Software Process Improvement") OR (("Model Driven Architecture" OR "MDA") AND "CMMI") OR (("Model Driven Architecture" OR "MDA") AND "MPS.BR")
2. (("Model Driven Development" OR "MDD") AND "Software Process Improvement") OR (("Model Driven Development" OR "MDD") AND "CMMI") OR (("Model Driven Development" OR "MDD") AND "MPS.BR")

2.5.1 Execução das Pesquisas

Com a metodologia e as strings definidas, as buscas foram realizadas no período de setembro a outubro de 2016. Foi estabelecido nas bases de dados que fossem procurados somente artigos do período posterior a 2005. A realização das buscas foi efetuada nas seguintes fases:

Fase 1: pesquisa nas bases de dados utilizando as duas strings;

Fase 2: eliminação de artigos duplicados;

Fase 3: leitura dos resumos;

Fase 4: leitura dos artigos selecionados.

A Tabela 5 apresenta o total de artigos das buscas pelas base de dados que são referentes à Fase 1.

Tabela 5 – Total de Publicações

	String 1	*String* 2
IEEE *Xplore Digital Library*	5	6
ACM *Digital Library*	0	0
Springer	114	157

Após as buscas realizadas, iniciou-se as Fase 2 e 3 que consistiu na eliminação de artigos duplicados pelos títulos e na leituras dos resumos. Essas fases visaram eliminar artigos que não estavam diretamente relacionados com o tema. Na fase 4, após realizada a triagem necessária, os artigos selecionados foram estudados. A tabela 6 apresenta todas essas fases discriminadas.

Tabela 6 – Total de artigos selecionados por Fases

Fase	Quantidade
Fase 1	282
Fase 2	72
Fase 3	26
Fase 4	5

O artigo de Valenzuela e Pavlich-Mariscal [63] tem como proposta um modelo de maturidade voltado para as pequenas e médias empresas, no qual inclui os conceitos do MDD. A estrutura deste modelo de maturidade é composta por processos do CMMI, ISO/IEC 12207 e do MDD, sendo um conjunto de 33 metas que são agrupadas pelas categorias de projeto, engenharia, suporte e MDD. Outra característica deste modelo de referência é que ele possui quatro níveis de maturidade. Como procedimento de avaliação para as organizações que adotarão o modelo proposto, foi utilizado o SCAMPI(*Standard CMMI Appraisal Method for Process Improvement*) (método aceito para avaliar organizações perante os modelos CMMI). Neste artigo foi aplicado o modelo proposto para três grupos de pequenas empresas. No entanto, os resultados obtidos mostraram pequenos ganhos, pois muitas ações propostas pelos modelos não foram colocadas em prática. Os motivos relatados foram falta de tempo e pouco apoio da gestão para a implementação e execução da proposta.

O trabalho de Vasconcelos et al. [64] analisa se é possível elaborar um processo MDD totalmente em conformidade com um modelo de qualidade. Para isso, foi analisada uma abordagem específica de MDD, denominado OO-*Method*. Já o modelo de referência adotado foi o CMMI. Um fator interessante deste artigo é o método de avaliação utilizado, pois é uma adaptação do modelo SCAMPI para analisar a possível adoção do MDD. A análise efetuada foi sobre uma área do processo do CMMI, o TS (*Techinical Solution*). Apesar do resultado não ter sido totalmente positivo, os pontos fracos identificados possuíam solução, sendo que o artigo aborda também as sugestões para que que todos esses pontos sejam atendidos. Os autores concluem e enfatizam a necessidade de analisar a conformidade das abordagens MDD em relação a qualquer modelo de qualidade e também defendem a adaptação da análise a outros modelos de qualidade e a outras abordagens de MDD .

Li et al. [65] elabora a definição de um processo de software voltado para o desenvolvimento de software geograficamente distribuído, no qual este processo de software denominado CSP(*CMM Software process*) está em conformidade com os requisitos do CMM(*Capability Maturity Model*). O MDA foi utilizado para transformar o modelo CSP em um modelo de implementação distribuído do CMM, sendo que este metamodelo é denominado MM-CSP. O MM-CSP descreve os conceitos e seus relacionamentos com a finalidade de construir e interpretar o modelo CSP. O modelo CSP não é um modelo de

processo de uma organização específica. Seu conceito parte que as equipes distribuídas possam adicionar suas características em seus modelos CSP para criar seus modelos de processo de implementação do CMM. O artigo também apresenta um protótipo de ferramenta que utilizou o MM-CSP, no qual essa ferramenta apoia a modelagem do modelo CSP.

Em Rios et al. [66] é explorado o modelo de maturidade MDD (MDD *Maturity Model*), desenvolvido dentro do projeto MODELWARE. Este modelo de maturidade possui cinco níveis de maturidade, sendo que cada nível descreve um conjunto consistente de práticas de engenharia, gestão e suporte dentro da abordagem MDD. Um dos principais requisitos do modelo de maturidade MDD é que ele seja compatível com o CMMI, além de definir como as atividades MDD podem ampliar as práticas específicas do CMMI.

Vasconcelos et al. em [67] apresenta um *framework* que integra a abordagens de requisitos GORE (*Goal-Oriented Requirement Engineering*) e MDD em conformidade com o modelo de maturidade CMMI. A abordagem GORE utilizada foi a *framework* i*. Os autores ressaltam que esta abordagem não é usual no cenário industrial. Uma alternativa para torná-la mais atraente para as empresas é o alinhamento da abordagem i* com os processos de um modelo de maturidade. Neste caso, o modelo escolhido foi o CMMI. O *framework* proposto pelo trabalho é chamado de GO-MDD, no qual utiliza uma abordagem de MDD denominada OO-*Method*, o *framework* i* e o CMMI na área de processos de desenvolvimento de requisitos (RD- *Requirements development*). Para demonstrar a adesão do *framework* proposto GO-MDD com o modelo de maturidade foi apresentado um mapeamento detalhado de conformidade.

Com esta pesquisa, pode-se observar que não existem muitos trabalhos que envolvem MDD/MDA com modelos de referência. Logo, a questão levantada no começo desta seção :“Q1- Qual é o cenário de pesquisas que correlacionam o MDD/MDA com os SPI’?” obteve-se as seguintes respostas:

- só foram encontrados trabalhos relacionados com o modelo de referência CMMI e o ISO/IEC 12207. Logo, pode-se constatar que nestes trabalhos não houve uma tratativa específica com o MPS.BR. No entanto, os artigos mostram uma relevância do MDD/MDA junto aos modelos de referência de qualidade;

- o emprego do MDD pode produzir produtos (artefatos) que servem como evidência de qualidade dentro do modelo de referência CMMI;

- o emprego do MDD como um paradigma de desenvolvimento de software não garante vantagens para as pequenas empresas na implantação do CMMI. A argumentação para este fato refere-se a dificuldade das empresas colocar em práticas ações definidas;

- o MDD pode produzir evidências de qualidade para os mais variados modelos de referência.

Como contribuição para esta dissertação pode-se destacar os trabalhos de Vasconcelos et al. [64] e [67], que destacam a necessidade de analisar a conformidade de abordagens MDD com qualquer modelos de referências, além de afirmar que a análise pode ser adaptada a outros modelos de qualidade e a outras abordagens de MDD.

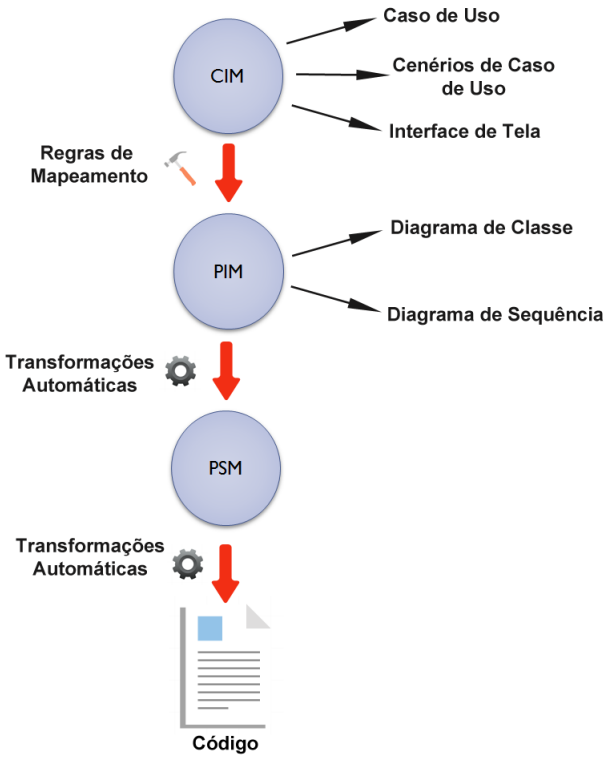
Destaca-se que o diferencial deste trabalho é criar um instância de MDA, a qual irá gerar modelos/artefatos que serão utilizados como evidências para certos quesitos de qualidade do MPS.BR. Portanto, esta instância será um apoio para analisar se juntamente com os conceitos do MDA, é possível antecipar resultados esperados de processos do MPS.BR, sendo analisado todos os níveis de maturidade que contenham processos.

3 PROPOSTA DE UMA INSTÂNCIA DE MDA

O MDA é uma metodologia de desenvolvimento composta pelas fases CIM, PIM e PSM, sendo que para permear de uma fase para outra, é necessário que ocorra transformações. As transformações no MDA lidam com a produção de diferentes modelos ou artefatos, partindo de um modelo baseado em um padrão de transformação. Cabe ressaltar que existe uma dificuldade encontrada na realização da transformação automática de CIM para PIM, sendo que, esta transformação, quando realizada, é de forma manual. Isto acontece por não haver artefatos e/ou regras específicas que definam quais elementos necessitam ser utilizados no CIM. Desta forma, não há uma normatização para o mapeamento de CIM-PIM [49], [68], [69] e [70].

Portanto, para criar uma normatização para este mapeamento, foi elaborada uma instância de MDA sobre a arquitetura MVC. Nesta instância foi definido o CIM, seus artefatos e regras para a elaboração destes modelos, criando assim uma normatização. A partir disto, foram formuladas as regras de conversão de CIM para PIM. As transformações de PIM para PSM e de PSM para código serão realizadas conforme as especificações do MDA. A Figura 12 apresenta estas etapas. Nesta Figura, também estão apresentados os artefatos gerados em cada fase.

Figura 12 – Fases do MDA com seus artefatos. Fonte: do autor.



Conforme exposto na Figura 12, a seguir será explicado o que será executado em cada fase da instância elaborada.

Para a elaboração do CIM, o diagrama de caso de uso e a interface de tela serão adotados. Para a elaboração dos cenários que ilustram os casos de uso serão empregadas as regras da sintaxe BNF elaboradas neste trabalho. Logo as frases dos cenários devem estar dentro da sintaxe estabelecida.

Na fase PIM são definidos dois diagramas: o diagrama de classe e o diagrama de sequência. Os dois diagramas são desenvolvidos a partir da derivação do diagrama de caso de uso e da interface de tela, sendo que essa derivação é realizada a partir das regras definidas. As regras elaboradas são baseadas na gramática BNF desenvolvida.

As derivações dos diagramas da fase PIM para a fase PSM ocorrem de forma automática, logo, esta transformação resultará nos diagramas de classe e sequência específicos para plataforma. Do mesmo modo, o código gerado pelos diagramas ocorre de maneira automática.

Também, neste capítulo apresenta-se um problema em que serão aplicadas todas as regras definidas. Para este problema, serão desenvolvidos os modelos de CIM e as transformações para o PIM.

3.1 Modelo CIM proposto

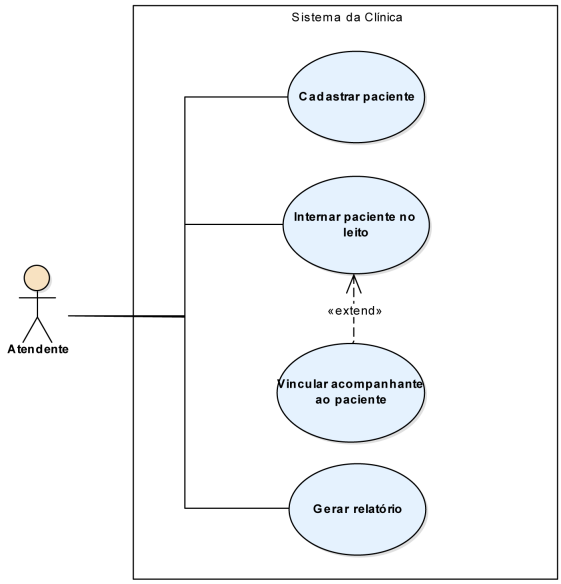
Para a fase CIM é adotado o diagrama de caso de uso, sendo que este será utilizado para a documentação dos requisitos. Para a elaboração das orações dos cenários de casos de uso, adotam-se as regras da sintaxe definida do BNF, com o propósito de viabilizar as transformações de CIM para PIM. Assim sendo, todas as frases elaboradas devem seguir o que foi especificado pela sintaxe BNF. Para o modelo CIM também são definidas interfaces de telas. Neste caso, as interfaces de tela serão somente uma representação gráfica, apresentando uma visão do sistema. Destaca-se que a OMG não estabelece um padrão para o CIM [22].

Alguns trabalhos trazem os diagramas de caso de uso na fase PIM [71], [72], [73] e [74]. Porém, Osis et al. [75] argumenta que os casos de uso descrevem o domínio do problema como uma “caixa preta”, sendo que sua prioridade de modelagem de domínio é baixa, portanto, podem ser utilizados no CIM. Os seguintes trabalhos [76] , [77], [78], [79] e [80] também abordam o diagrama de caso de uso na fase CIM. Destaca-se o artigo [76], pois este utiliza cenários de casos de uso para compor a fase CIM.

3.1.1 Artefatos do CIM

O diagrama de caso de uso auxilia na identificação e compreensão dos requisitos [46]. Nesta dissertação, o caso de uso foi elaborado para a fase CIM, pois é independente de computação, além de ser definido pela UML [2]. Com isso, agrega-se ao fato de ser um diagrama especificado pelas ferramentas de MDA. Portanto, indica-se a utilização de ferramentas de MDA para melhor suporte à atividade e também interligação deste diagrama com suas evoluções. Na Figura 13, foi exibido o diagrama de caso de uso gerado por este trabalho.

Figura 13 – Caso de Uso. Fonte: do autor.

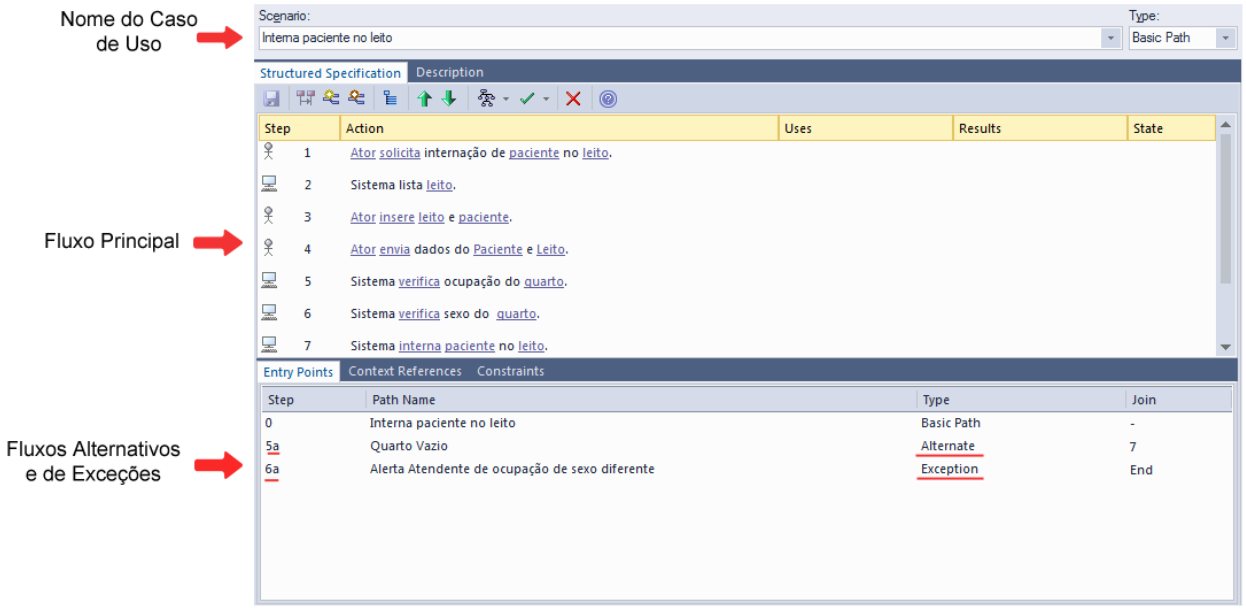


O cenário de caso de uso consiste em descrever as ações de forma sequencial, ilustrando assim, o comportando de cada caso de uso. Devem-se considerar os seguintes elementos para a elaboração dos cenários dos casos de uso:

- Fluxo Principal: sequência de ações que serão executadas, considerando que nada de errado ocorrerá durante essa execução;
- Fluxo Alternativo: descreve o que deve ocorrer quando o ator faz uma escolha alternativa do fluxo principal;
- Fluxo de Exceção: corresponde à descrição das situações de exceção, quando algo inesperado ocorre na interação com o sistema;
- Pré-Condição: define que hipóteses são assumidas como verdadeiras para que o cenário ocorra;
- Pós-Condição: estado que o sistema alcança, após o caso de uso ter sido realizado.

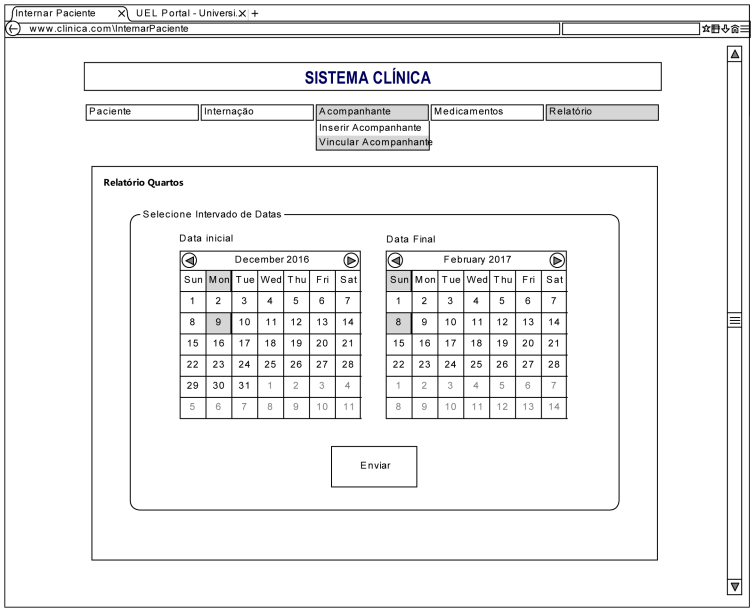
A Figura 14 apresenta o exemplo de um cenário de caso de uso que pode ou não ser realizado nas ferramentas de MDA.

Figura 14 – Cenário do caso de uso. Fonte: do autor.



No CIM também é definido pelo menos uma interface de tela, para cada caso de uso que há interações com usuários do sistemas. A interface de tela definida deve ser uma visão do sistema, sem a preocupação com conhecimentos específicos de computação. Com a interface pode-se ter uma visualização mais clara da proposta do sistema e também a conversão de informações importantes, como por exemplo, dados e interações relevantes à especificação. Outro ponto no qual a interface auxilia neste trabalho é com a arquitetura MVC. A interface faz o papel da *View* do MVC. A interface de tela deve estar aderente aos termos/palavras empregadas no caso de uso. Portanto, quando os diagramas estão se referindo as mesmas funcionalidades, estes devem ter a mesma notação. Na Figura 15 está apresentada uma interface de tela gerada neste trabalho.

Figura 15 – Interface de Tela. Fonte: do autor.



3.1.2 Definição de uma Sintaxe para os Cenários dos Casos de Uso

Neste trabalho o BNF [1] foi criado com o objetivo de elaborar uma notação formal para especificar os cenários que descrevem os caso de uso do CIM, sendo utilizado também para a extração de informação e regras para o mapeamento entre os diagramas.

O BNF proposto é baseado na Língua Portuguesa e na sua estrutura de análise sintática. A análise sintática tem como objetivo examinar a estrutura da frase, sendo que cada palavra na oração possui uma função sintática. Portanto, o BFN que será apresentado possui um subconjunto de elementos que constituem a análise sintática.

Essa sintaxe tem como intuito delimitar o tamanho e a composição das frases que integram os cenários dos casos de uso. Assim cria-se uma estrutura e uma normatização que possibilita o mapeamento dos modelos, viabilizando as transformações de modelos de CIM para PIM.

A sintaxe BNF criada neste estudo é apresentada a seguir. Foram utilizadas as mesmas regras e notações utilizadas pela OMG na documentação da especificação da UML [82]. O detalhamento destas especificações estão no Anexo A desta dissertação.

[1] O Backus-Naur Form (BNF) é uma notação utilizada para expressar a gramática de uma linguagem sob a forma de regras de produção. As gramáticas BNF são compostas por terminais e não terminais, sendo que os não terminais podem ser expandidos para um ou mais terminais ou não terminais [81].

<oração> ::= ***<sujeito> <predicado>***

<predicado> ::= 'informa' ***<msg>***| ***<verbo> <objeto>*** | ***<verbo> <objeto> <complemento>***

<complemento> ::= ***<adjunto adnominal> <complemento1>*** | ***<adjunto adverbial > <complemento2>*** | ***<complemento nominal> <complemento3>*** |

<complemento1> ::= ϵ [2] |***<adjunto adverbial>*** | ***<complemento nominal>***

<complemento2> ::= ϵ | ***<adjunto adnominal>*** | ***<complemento nominal>***

<complemento3> ::= ϵ | ***<adjunto adnominal>*** | ***<adjunto adverbial>***

Os terminais serão gerados de acordo com as sentenças produzidas nos cenários dos casos de usos. Considerando os não- terminais ***<sujeito>***, ***<adjunto adverbial>*** ,***<adjunto adnominal >*** e ***<complemento nominal>*** derivam em terminais compostos por um único substantivo, correspondente a sua classe gramatical e o não- terminal ***<verbo>*** deriva em um único terminal verbo, sendo que todos estes terminais gerados podem ser acompanhados por artigo(s) e/ou preposição (s).

O terminal que é derivado pelo não-terminal ***<msg>*** não possui uma restrição de palavras, podendo conter verbos e substantivos ilimitados e em qualquer conjugação verbal e sintática. No entanto, o terminal derivado por ***<msg>*** só acompanhará o terminal 'informa' e com isso, denotará o envio de uma mensagem para o usuário do sistema.

Algumas regras, além da sintaxe do BNF, são estabelecidas para o desenvolvimento dos cenários dos casos de uso. Essas regras são:

- para o não-terminal ***<sujeito>*** pode-se derivar em um terminal 'sistema', a qual é uma palavra reservada que denota o próprio sistema que controla as ações do software. Porém, o não-terminal ***<sujeito>*** pode derivar em um terminal que denote um ator que interaja com o sistema. Este terminal será gerado quando forem elaborados os cenários de caso de uso.

- o terminal derivado pelo não-terminal ***<verbo>*** contido em cada sentença deve estar em uma conjugação verbal específica: o presente do modo Indicativo. O não-terminal ***<verbo>*** têm dois terminais que são palavras reservadas :

  - um verbo em específico será utilizado para indicar a entrada de dados pelo ator que interage com o sistema. Esse verbo será o INSERE.

[2] ϵ denota a cadeia vazia

  - o verbo INFORMA é uma palavra reservada e será utilizada para denotar o envio de mensagem pelo sistema.

- os não-terminais ***<adjunto adverbial>*** ,***<adjunto adnominal >*** e ***<complemento nominal>*** não derivam em terminais que são palavras reservadas. Os terminais que serão derivados por esses não-terminais serão gerados nos cenários de casos de uso.

- uma característica muito importante que as frases dos cenários dos casos de uso devem conter, além da sintaxe do BNF, é que a última palavra que compõem a frase deve ser uma palavra importante para o caso de uso. Geralmente essa palavra responde as perguntas do gênero: onde?, quem? ... . Essa especificação é de grande importância, pois essa "última palavra" da frase que será convertida e se tornará a classe e a *lifeline* nas regras que serão apresentadas mais adiante neste capítulo.

3.2 Conversão de CIM para PIM

Existe uma dificuldade encontrada na realização da transformação automática de CIM para PIM, sendo que esta transformação, quando realizada, é de forma manual. Isto é devido por não haver artefatos e/ou regras específicas que definam quais elementos necessitam ser utilizados no CIM. Portanto, não existe uma padronização para a transformação de CIM para PIM [49], [68], [69] e[70].

Nesta dissertação para a conversão de CIM para PIM serão estabelecidas regras de conversão, as quais são baseadas na sintaxe BNF definida. O diagrama de classe e o diagrama de sequência são desenvolvidos a partir do diagrama de caso de uso, cenários de casos de uso e da interface de tela.

As regras que serão expostas estão denotadas com os símbolos não-terminais. Esta notação foi utilizada para denotar o BNF desenvolvido e referenciar os elementos que compõem uma frase. No entanto, destaca-se que as análises e as aplicações das regras são realizadas sobre os símbolos terminais.

3.2.1 Extração de Informação do diagrama de Caso de Uso e da Interface para o diagrama de Classe

Na arquitetura MVC, todo caso de uso que possuir ao menos uma interface do sistema terá uma classe com o estereótipo *interface*. A classe com esteriótipo *Controller* também deve ser criada, pois ela receberá os métodos da classe *View* e distribuirá certas atividades do cenário caso de uso. As classes *Controller* e *View* devem possuir o mesmo nome do caso de uso a qual estão analisadas/relacionadas. Para identificar as classes *Controller* e *View*, além dos esteriótipos, deve-se colocar C_ para o *Controller* e V_ para

a *View* , antes do nome do caso de uso. As classes referentes ao *Model* serão adicionadas conforme ocorrer a aplicação das regras.

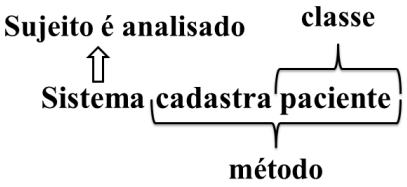
Uma regra estabelecida, que deve ser levada em consideração desde o desenvolvimento do próprio caso de uso, é que toda frase do cenário de caso de uso deve ser possível de conversão. Essa conversão é viabilizada e fundamentada pela sintaxe BNF. Portanto, a partir desta regra deve-se analisar todas as frases do cenário do caso de uso, decompondo-as e analisando-as conforme as regras dadas a seguir:

1. Para a frase com a seguinte estrutura ***<sujeito> <verbo> <objeto>*** adota-se que o ***<verbo>*** seguido do ***<objeto>*** é transformado no nome do método. Para definir a qual classe o método pertence, o ***<sujeito>*** deve ser analisado:

   - Se o ***<sujeito>*** for o 'sistema':
     - o ***<objeto>*** é transformado em classe e o método deve ser inserido nesta classe.
   - Se o ***<sujeito>*** for um ator diferente de 'sistema':
     - o método deve inserido nas classes *View* e *Controller* referentes ao diagrama de caso de uso analisado.

   A Figura 16 expõe um exemplo da aplicação da regra apresentada anteriormente.

Figura 16 – Exemplo da aplicação da Regra 1 para o Diagrama de Classe. Fonte: do autor.



2. Para a oração com a seguinte estrutura ***<sujeito> <verbo> <objeto> <complemento>*** adota-se que o ***<verbo>*** seguido do ***<objeto>*** é transformado no nome do método. Para esta regra, considera-se que os ***<complemento1>***, ***<complemento2>*** e ***<complemento3>***, geraram a cadeia vazia (ϵ). Portanto, as frases só apresentam a seguinte estrutura: ***<sujeito> <verbo> <objeto> <adjunto adnominal>*** | ***<sujeito> <verbo> <objeto> <adjunto adverbial>*** | ***<sujeito> <verbo> <objeto> <complemento nominal>***.Para definir a qual classe o método pertence, o ***<sujeito>*** deve ser analisado:

   - Se o ***<sujeito>*** for o 'sistema':
     - o ***<complemento>*** é transformado em classe e o método deve ser inserido nesta classe.

- Se o ***<sujeito>*** for um ator diferente de 'sistema':
  - o método deve inserido nas classes *View* e *Controller* referentes ao diagrama de caso de uso analisado.

A Figura 17 expõe um exemplo da aplicação da regra apresentada anteriormente.

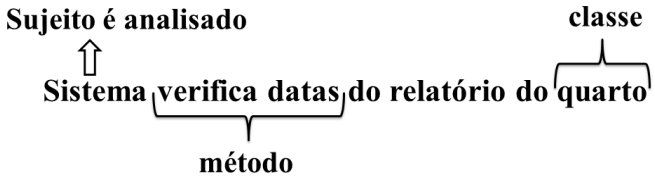
Figura 17 – Exemplo da aplicação da Regra 2 para o Diagrama de Classe. Fonte: do autor

3. Para a oração com a seguinte estrutura ***<sujeito>*** ***<verbo>*** ***<objeto>*** ***<complemento>*** ***<complementoN>***, adota-se que o ***<verbo>*** seguido do ***<objeto>*** é transformado no nome do método. Para esta regra o ***<complementoN>*** está representando os ***<complemento1>***, ***<complemento2>*** e ***<complemento3>***, pois não há diferença na regra de conversão para esses não-terminais. Neste caso, o ***<complementoN>*** não irá gerar a cadeia vazia(ϵ), pois esta restrição já está sendo abordada na regra 2. Para definir a qual classe o método pertence, o ***<sujeito>*** deve ser analisado:

   - Se o ***<sujeito>*** for o 'sistema':
     - o ***<complementoN>*** é transformado em classe e o método deve ser inserido nesta classe.
   - Se o ***<sujeito>*** for um ator diferente de 'sistema':
     - o método deve inserido nas classes *View* e *Controller* referentes ao diagrama de caso de uso analisado.

A Figura 18 expõe um exemplo da aplicação da regra apresentada anteriormente.

Figura 18 – Exemplo da aplicação da Regra 3 para o Diagrama de Classe. Fonte: do autor



Outras regras que são válidas para todas as sentenças do cenário do caso de uso são:

- o verbo Insere é uma palavra reservada que tem como significado a entrada de dados pelo usuário do sistema. Portanto, quando a sentença apresentar este verbo, esta não deve ser convertida para o diagrama de classe;
- após a sentença com o verbo Insere, as próximas sentenças utilizam os dados inseridos como seus parâmetros dos métodos. Cabe ressaltar que nem todos os métodos criados utilizaram os dados acrescentados;
- frases que contém a palavra reservada 'informa' não serão convertidas nas regras definidas anteriormente. Neste caso, a regra válida consiste: 'informa' se transforma no nome do método e o terminal que deriva do não-terminal ***<msg>*** será passado como parâmetro. A classe a qual o método será inserido será somente a *View*, visto que este método é somente um comunicado para o usuário do sistema;
- outro aspecto a ser considerado é a questão das preposições, artigos e pronomes que compõe as frases. Para a conversão, esses devem ser desconsiderados;
- para facilitar a leitura, na criação do método, o terminal derivado de objeto deve estar com a primeira letra em maiúsculo.

Após as regras para o caso de uso, pode-se extrair informações da interface de tela, pois além de estabelecer uma classe com o estereótipo interface no diagrama de classe, esta permite também a extração de dados, os quais estão representados na tela. Portanto, quando houver campos e representações que reproduzem dados das possíveis classes já criadas, esses devem ser adicionados como atributos e devem ser privados.

3.2.2 Extração de Informação do diagrama de Caso de Uso e da Interface para o diagrama de Sequência

Na arquitetura MVC todo caso de uso que possuir pelo menos uma interface do sistema será uma instância *lifeline View*. Haverá também uma instância da *lifeline Controller*. Essas instâncias apresentam o mesmo nome do caso de uso em análise. Para identificar as *lifeline Controller* e *View*, além dos esteriótipos deve-se colocar C_ para o *Controller* e V_ para a *View* , antes do nome do caso de uso.

O ator 'sistema' será a própria *lifeline Controller* e os demais atores que interajam com a interface, serão atores também no diagrama de sequência.

Do mesmo modo que foi feita a análise de todas as frases dos cenários dos casos de uso para o diagrama de classe, será realizado o mesmo para o diagrama de sequência. Portanto, as regras de conversão são dadas a seguir:

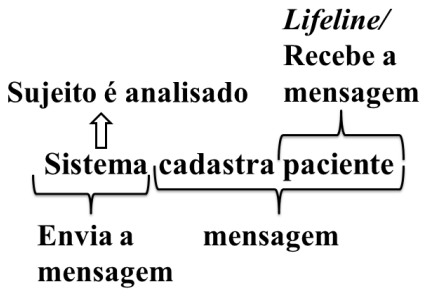
1. Para a oração com a seguinte estrutura ***<sujeito> <verbo> <objeto>*** adota-se que o ***<verbo>*** e o ***<objeto>*** se unam e formem o nome da mensagem enviada.

O ***<sujeito>*** da oração é quem envia a mensagem. Para definir onde a mensagem irá ser recebida deve-se analisar o ***<sujeito>*** :

- Se o ***<sujeito>*** é o 'sistema':
  - ***<objeto>*** é transformado em *Lifeline* , sendo que esta *Lifeline* criada irá receber a mensagem.
- Se o ***<sujeito>*** é um ator diferente de 'sistema':
  - a mensagem deve passar pela *View* e depois chegar no *Controller*.

A Figura 19 apresenta um exemplo da aplicação da regra exposta anteriormente.

Figura 19 – Exemplo da aplicação da Regra 1 para o Diagrama de Sequência. Fonte: do autor.

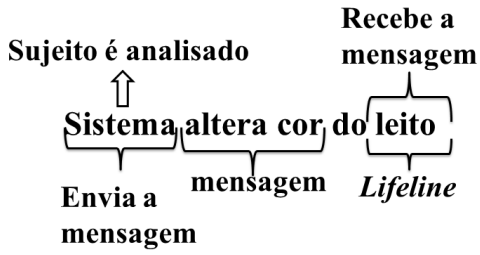


2. Para a oração com a seguinte estrutura ***<sujeito>*** ***<verbo>*** ***<objeto>*** ***<complemento>*** adota-se que o ***<verbo>*** e o ***<objeto>*** se unam e formem o nome da mensagem enviada. Para esta regra considera-se que os ***<complemento1>***, ***<complemento2>*** e ***<complemento3>*** geraram a cadeia vazia (ϵ). O ***<sujeito>*** da oração é quem envia a mensagem. Para definir onde a mensagem irá ser recebida, deve-se analisar o ***<sujeito>*** :

   - Se o ***<sujeito>*** é o sistema:
     - ***<complemento>*** é transformado em *Lifeline*, sendo que esta *Lifeline* criada irá receber a mensagem.
   - Se o ***<sujeito>*** é um ator que interage com sistema:
     - a mensagem deve passar pela *View* e depois chegar no *Controller*.

   A Figura 20 apresenta um exemplo da aplicação da regra exposta anteriormente.

Figura 20 – Exemplo da aplicação da Regra 2 para o Diagrama de Sequência. Fonte: do autor.

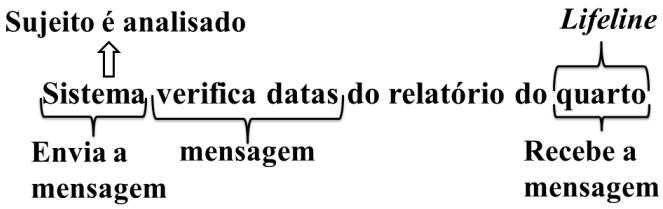


3. Para a oração com a sintaxe ***<sujeito> <verbo> <objeto> <complemento> <complementoN>*** adota-se que o ***<verbo>*** e o ***<objeto>*** se unam e formem o nome da mensagem enviada. Para esta regra o ***<complementoN>*** está representando os ***<complemento1>***, ***<complemento2>*** e ***<complemento3>***, pois não há diferença na regra de conversão para esses não-terminais. Neste caso, o ***<complementoN>*** não irá gerar a cadeia vazia(ϵ), pois esta restrição já está sendo abordada na regra 2. O ***<sujeito>*** da oração é quem envia a mensagem. Para definir onde a mensagem irá ser recebida, deve-se analisar o ***<sujeito>*** :

   - Se o ***<sujeito>*** é o sistema:
     - ***<complementoN>*** é transformado em *Lifeline* , sendo que esta *Lifeline* criada irá receber a mensagem.
   - Se o ***<sujeito>*** é um ator que interage com sistema:
     - a mensagem deve passar pela *View* e depois chegar no *Controller*.

   A Figura 21 apresenta um exemplo da aplicação da regra exposta anteriormente.

Figura 21 – Exemplo da aplicação da Regra 3 para o Diagrama de Sequência. Fonte: do autor.



Outras regras que são válidas para todas as sentenças são:

- o verbo INSERE é uma palavra reservada que tem como significado a entrada de dados pelo usuário do sistema, por exemplo, corresponde ao preenchimento de campos de um formulário. Portanto, quando a sentença apresentar este verbo não deve ser convertida para o diagrama de sequência;

- após a sentença com o verbo Insere, as próximas sentenças podem utilizar os dados inseridos como seus parâmetros no diagrama de sequência;
- frases com a palavra reservada 'informa' não seguirão as regras de conversão definidas para o diagrama de sequência. Neste caso, a regra válida consiste: a mensagem será a palavra INFORMA e o parâmetro será o terminal derivado do não-terminal ***<msg>***. O envio da mensagem será realizado pelo terminal derivado do não-terminal ***<sujeito>*** da oração, mas, no entanto, quem recebe a mensagem deve ser a Interface;
- os fluxos alternativos e de exceções do caso de uso geram um fragmento combinado no diagrama de sequência;
- para facilitar a leitura na criação da mensagem, o terminal derivado de objeto deve estar com a primeira letra em maiúsculo.

3.3 Outras Fases do MDA

As fases apresentadas anteriormente foram a CIM e a PIM. O MDA também é composto pela fase PSM. No entanto, da fase PIM para a fase PSM os modelos são gerados de forma automatizada, via ferramentas de MDA.

O PIM, gerado pelas regras expostas neste trabalho, é uma derivação dos requisitos levantados e documentados pelos modelos de CIM. Portanto, existe uma necessidade de se adicionar elementos a este PIM, que o complemente, como por exemplo, elementos de bancos de dados, componentes da classe *View*, adição de relacionamento ou associação entre as classes do modelo e entre outros. Esta complementação do PIM irá gerar um PSM e um código mais completo.

As ferramentas não possuem suporte para transformações que envolvam os modelos comportamentais, como é o digrama de sequência. No entanto, é possível extrair todas as informações necessárias a partir do XMI gerado do modelo, realizando assim, um mapeamento de modo manual.

O PSM combina as especificações do PIM com o detalhamento da plataforma escolhida. Destaca-se, que um PSM é capaz de ser transformado para diversas plataformas. A partir de um PSM é obtido o código de modo automático, através de uma ferramenta que suporta MDA.

3.4 Estudo de Caso

Nesta seção será apresentada a aplicação do que foi proposto neste capítulo. Para isso, será criado a fase CIM e seus modelos correspondentes e as transformações propostas

para a fase PIM. O problema, que será utilizado para a aplicação deste estudo, retrata o sistema de uma clínica médica de reabilitação. O sistema possui diversas funcionalidades, desde cadastrar paciente até geração de relatórios dos quartos. Cabe ressaltar que este problema é um estudo de caso real, sendo que os requisitos foram fornecidos por uma empresa desenvolvedora de software local, para que fosse possível aplicar este estudo em um âmbito mais concreto.

O sistema da clínica de reabilitação possui muitas funcionalidades, por isso foi selecionado um subgrupo dos requisitos requeridos, e estes serão apresentados a seguir.

A funcionalidade do cadastro de paciente tem como propósito a inserção de novos pacientes no sistema. Para isto, o usuário do sistema deve preencher primeiro uma identificação. O sistema fará uma busca para detectar se aquele paciente já está cadastrado. Caso aquele paciente não conste no sistema, o usuário poderá preencher os outros campos. No entanto, se o paciente já estiver cadastrado, uma mensagem avisará sobre este ocorrido.

Outra necessidade da clínica é a internação dos pacientes nos leitos dos quartos. Para isso, o usuário do sistema deve selecionar o paciente, juntamente com o leito desejado de um quarto. Quando uma pessoa é internada em um leito, esse leito muda de cor conforme o sexo de pessoa(rosa- feminino, azul- masculino). Caso o quarto esteja vazio, a classificação do quarto muda de acordo com o sexo do paciente a ser internado. Porém, se o quarto estiver ocupado, e o atendente tentar internar duas pessoas com sexos diferentes no mesmo quarto, uma mensagem será enviada, mas isso não impede que a ação internar seja executada. O sistema deve abordar, de forma interativa e visual, a questão dos leitos e quartos ocupados.

Em alguns casos, os pacientes precisam de acompanhantes nas suas internações. Portanto, deve haver uma vinculação entre eles. Um paciente só pode ter um acompanhante por vez, sendo que este já deve estar cadastrado no sistema. O leito que o acompanhante irá ocupar mudará para a cor verde, independente do sexo.

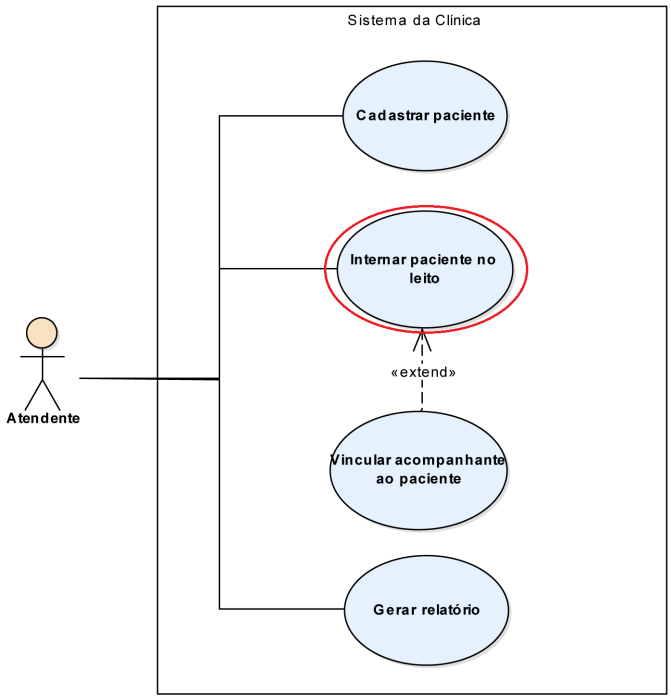
A última funcionalidade do sistema, que será empregada neste trabalho, é gerar relatório de quartos. Para isto, o usuário deve entrar com as datas iniciais e finais e o relatório dos quartos será gerado.

Para a edição dos diagramas apresentados nesta seção será utilizada a ferramenta *Enterprise Architect* (EA). Esta ferramenta dá suporte ao MDA, sendo reconhecida pela OMG. Cabe ressaltar que poderia ser utilizada qualquer ferramenta MDA para executar essa funcionalidade.

O caso de uso de todas as funcionalidades da clínica médica descritas anteriormente estão representadas na Figura 22. Porém, neste capítulo só será utilizado como exemplo o caso de uso “Internar Paciente no leito”, que está em destaque na Figura. Com ele, serão

aplicados todos os passos do MDA. Os demais diagramas elaborados estão no Apêndice A.

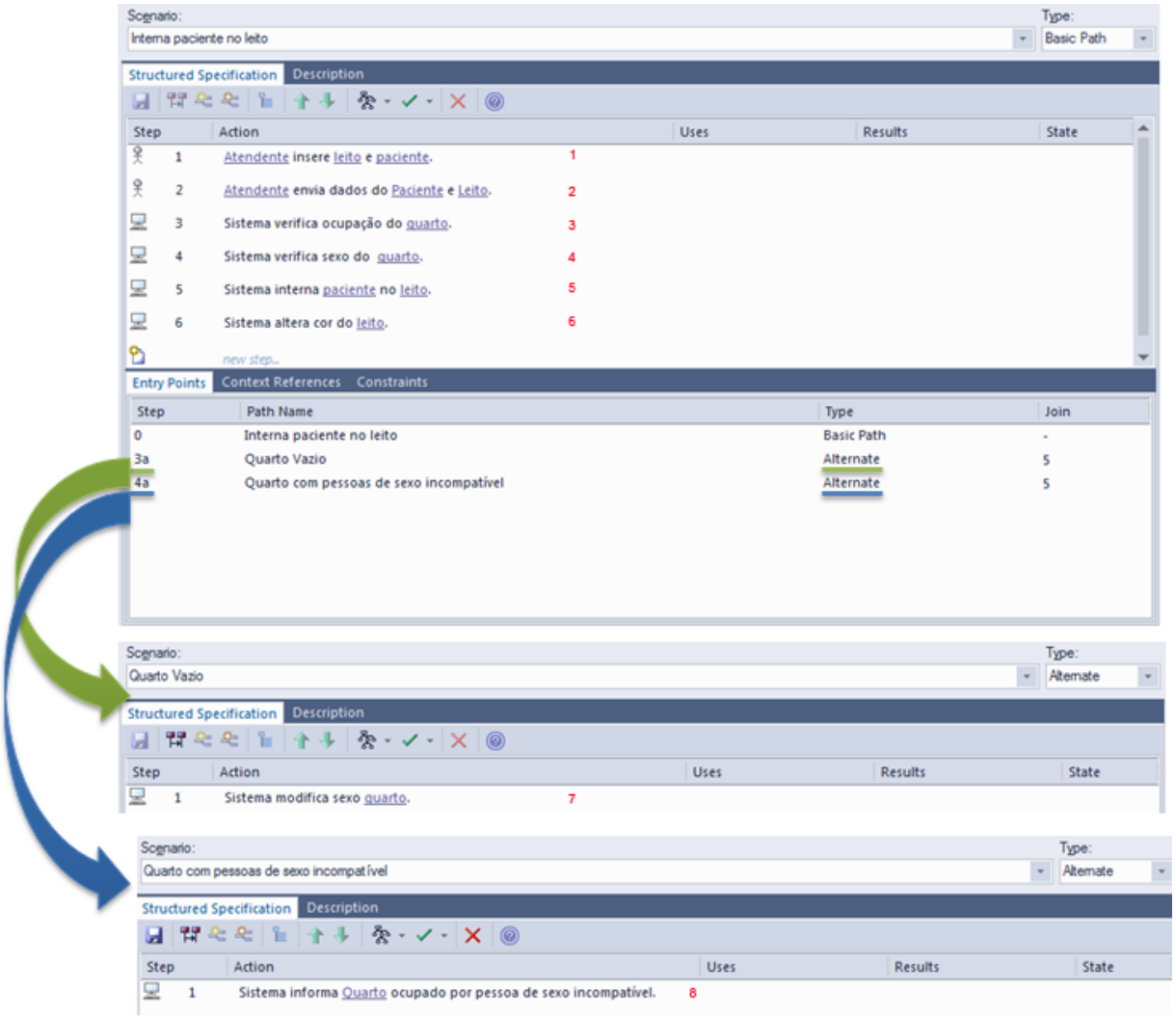
Figura 22 – Caso de Uso da Clínica. Fonte: do autor.



Para cada caso de uso contido no diagrama da Figura 22 é elaborado um cenário de caso de uso e uma interface de tela correspondente. Os cenários devem seguir as regras apresentadas neste capítulo. Portanto, as orações dos cenários de casos de uso devem seguir a sintaxe BNF e também as outras regras estabelecidas.

A Figura 23 é o cenário de caso de uso de “Internar Paciente no leito” . As partes indicadas pelas setas são os fluxos alternativos. Os números que estão em vermelho serão utilizados para demonstrar as transformações para os diagramas da fase PIM.

Figura 23 – Cenário de Caso de Uso de Internar Paciente no Leito. Fonte: do autor



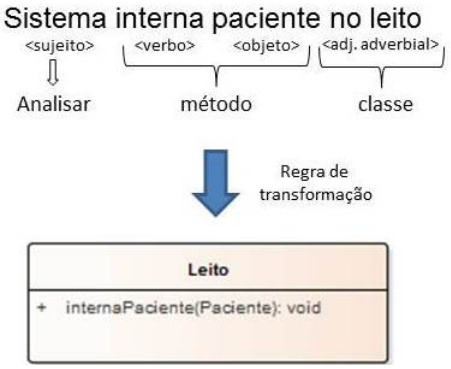
A Figura 24 é a interface de tela correspondente ao caso de uso de “Internar Paciente no leito”.

Figura 24 – Interface de Tela do Internar Paciente no Leito. Fonte: do autor.

Com todos os modelos apresentados anteriormente têm-se a etapa CIM completa. Portanto, a próxima fase será executar a transformação proposta neste capítulo, sendo que os diagramas da etapa PIM são desenvolvidos a partir da derivação dos diagramas da fase CIM.

Para ilustrar, a Figura 25 apresenta a transformação de uma oração retirada do cenário de caso de uso para o diagrama de classe. Nesta transformação é apresentada a oração, quais os elementos sintáticos que compõem a frase de acordo com o BFN e, por último, a classe já mapeada, de acordo com as regras estabelecidas.

Figura 25 – Exemplo de Transformação para o Diagrama de Classe

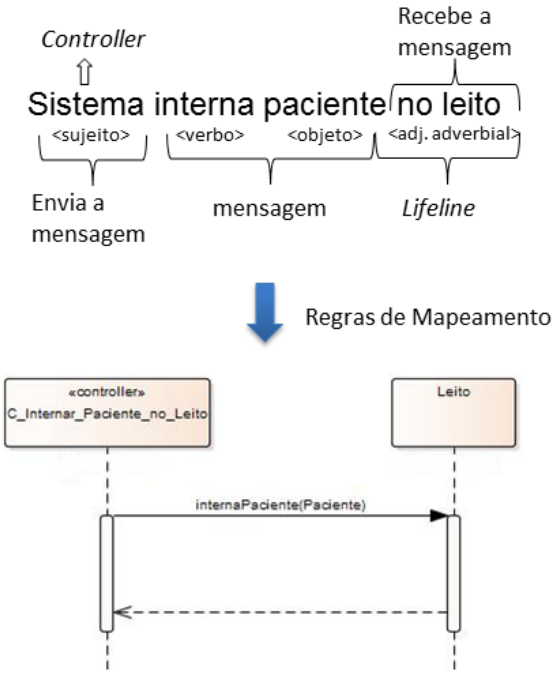


Pode-se notar que na Figura 25 o método resultou da junção do verbo e objeto da oração. A próxima etapa foi analisar o sujeito, como neste caso o sujeito é “sistema”, o adjunto adverbial da oração foi transformado no nome da classe.

Para realizar a mesma exemplificação da transformação com o diagrama de sequên-

cia, a Figura 26 apresenta uma oração retirada do caso de uso para a transformação em diagrama de sequência. Na Figura 26 é exposta a oração em estudo, e a sua análise sintática de acordo com o BNF e qual é o diagrama resultante após a aplicação da regra de mapeamento.

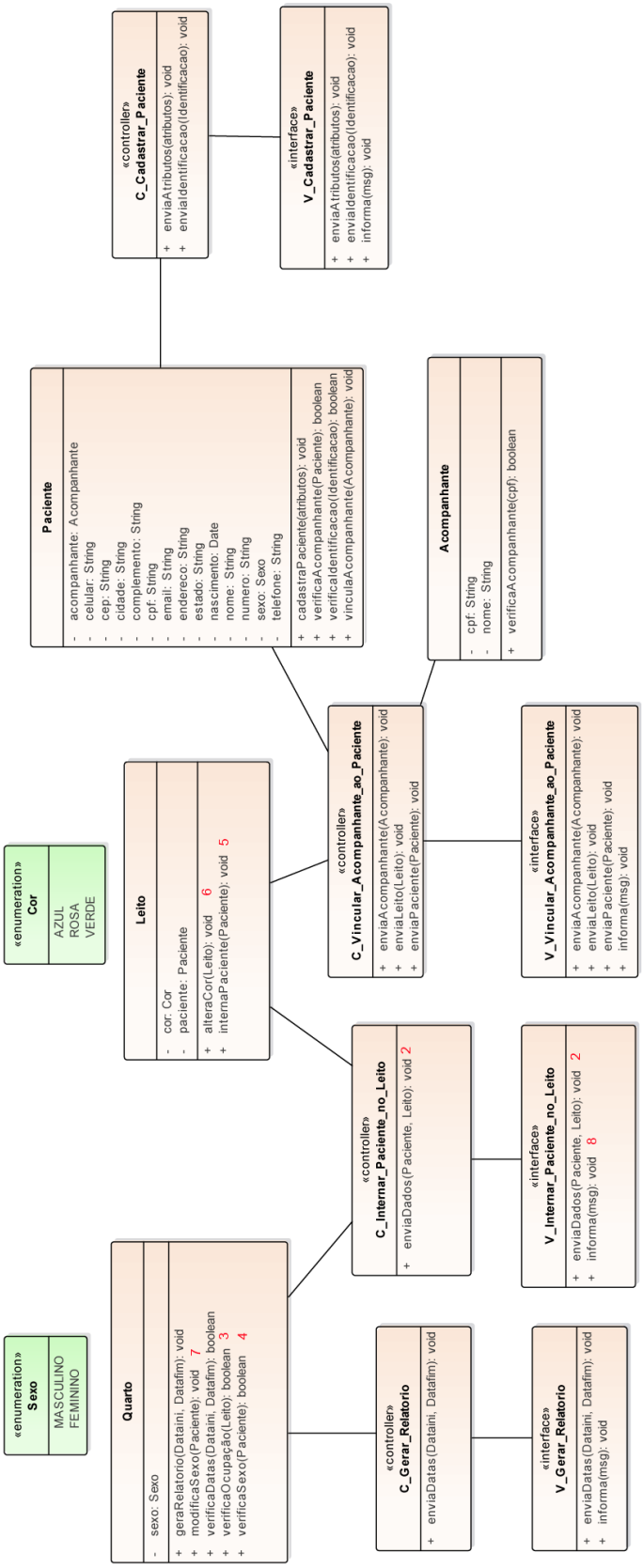
Figura 26 – Exemplo de Transformação para o Diagrama de Sequência



Na Figura 26, a mensagem resultou da junção do verbo e objeto da oração. Para identificar quem iria enviar a mensagem, analisou-se o sujeito, neste caso, como o sujeito é “sistema” e este pelas regras é definido *Controller* e possui o mesmo nome do caso de uso, portanto, assim sendo, é o *Controller* quem enviou a mensagem. Para definir onde a mensagem seria recebida, o sujeito também foi analisado, neste caso como o sujeito é “sistema” a *Lifeline* Leito foi criada a partir do adjunto adverbial e, é ela quem recebeu a mensagem.

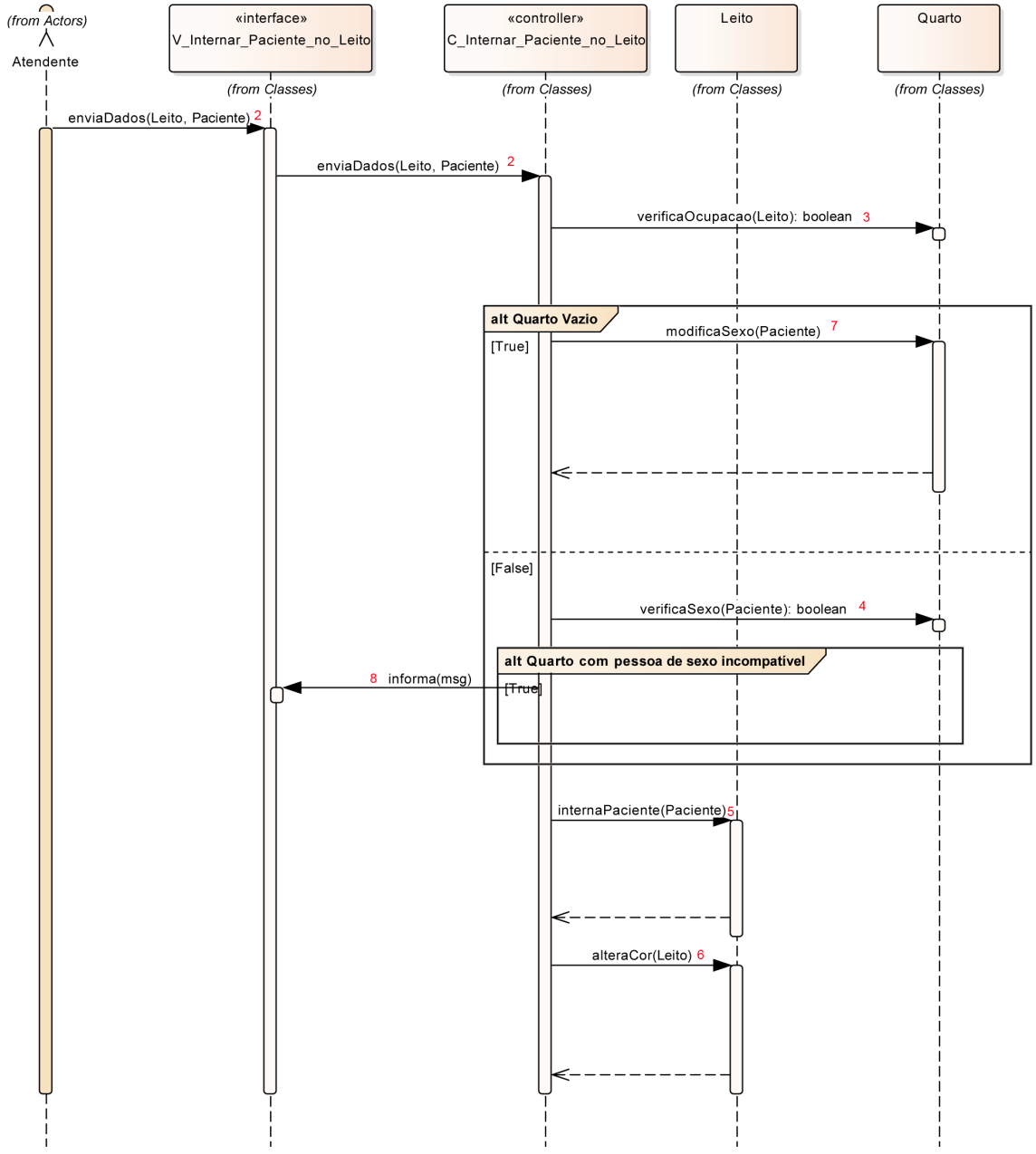
A Figura 27 apresenta o diagrama de classe derivado dos modelos de CIM a partir das regras de conversão definidas. Este diagrama apresenta todas as classes e métodos referentes ao caso de uso da Figura 22. Os métodos indicados com números em vermelho, são os métodos que derivaram do cenário de caso de uso da Figura 23. As classes também são derivadas das orações do cenário de caso de uso. Neste caso, por exemplo, as classes geradas foram Quarto e Leito. Os atributos contidos nas classes foram extraídos da interface de tela. Por exemplo, a interface de tela da Figura 24 apresenta que o leito possui paciente e cor, já o quarto possui gênero (masculino/feminino). Logo, a classe Leito deve possuir os atributos cor e paciente, e a classe Quarto deve possuir o atributo sexo.

Figura 27 – Diagrama de Classes. Fonte: do autor.



O diagrama de sequência derivado dos modelos CIM e das regras de mapeamento está exposto na Figura 28. Este diagrama é referente ao caso de uso “ Internar Paciente no Leito”. Os números em vermelho em frente as mensagens mostram as derivações do cenário de caso de uso da Figura 23. As *lifelines* geradas pelas regras também se encontram nas orações do cenário do caso de uso. Nota-se que os fragmentos combinados são os fluxos alternativos do cenário de caso de uso, como especificados nas regras.

Figura 28 – Diagrama de Sequência. Fonte: do autor.



Nos dois diagramas apresentados todas as frases do cenário de caso de uso foram convertidas, exceto a linha *um*, pois esta linha possuía o verbo Insere. Nas regras para conversão do diagrama de classe e do diagrama de sequência era explícito que a frase que possuísse esse verbo não seria convertida.

As demais etapas (evolução do PIM, PIM para PSM, PSM para código) estão apresentadas no Apêndice A.

3.5 Considerações Finais

Neste capítulo foram apresentados os conceitos para a elaboração dos modelos de CIM e para a conversão dos mesmos para modelos de PIM, voltados para o padrão MVC. Na fase CIM foram propostos o diagrama de caso de uso e a interface de tela. Já para a fase PIM foram apresentados os diagramas de classe e o de sequência.

O fato de estabelecer um BNF para o desenvolvimento do cenário de caso de uso restringe a escrita das orações. Porém, essa restrição oferece benefícios, como por exemplo, uma normatização de modelos padronizados que facilitam a extração de informações, o que consequentemente, auxilia em um mapeamento para a conversão de CIM para PIM. Outro ponto positivo agregado pelo BNF é a homogeneização da linguagem. Portanto, isso manterá um padrão entre os desenvolvedores dos modelos e uma padronização entre os modelos desenvolvidos na linha do tempo, o que auxilia na qualidade, manutenção e criação de uma linguagem para cada sistema, estendida até o nível de código. Outro ponto relevante é que a gramática BNF se baseia na linguagem natural. Desta forma, os desenvolvedores dos modelos não precisam aprender uma nova linguagem.

Para a conversão em modelos do PIM, regras foram estabelecidas. Com isso, foram divididos em dois conjuntos de regras: um conjunto para o diagrama de classe e o outro para o diagrama de sequência. Cabe ressaltar que os modelos de PIM gerados por estas regras, ainda podem evoluir e passar por mais transformações, antes de serem convertidos para PSM. Assim, algumas informações poderão ser supridas caso não sejam contempladas pelas regras de transformações propostas.

Pode-se observar nos trabalhos [49], [68], [69] e [70] uma dificuldade na normatização para o mapeamento de CIM para PIM. Neste capítulo, apresentou-se regras que possibilitam o mapeamento entre as fases CIM e PIM, e especificam os artefatos e regras a serem utilizados.

4 EMPREGO DOS ARTEFATOS DO MDA NOS PROCESSOS DO MPS.BR

O desenvolvimento deste capítulo origina-se de processos prontos do MPS.BR nível G, sendo que estes processos foram cedidos pela empresa desenvolvedora de software Guenka. Não foram criados novos processos de qualidade junto com o MDA, visto que a implementação, avaliação e certificação são procedimentos que precisam de tempo, pessoas e recursos financeiros para as avaliações. Assim, seria inviável ao presente projeto. Portanto, utilizou-se de um processo já avaliado pela Softex. Assim, os processos que serão utilizados neste trabalho já passaram por avaliação e foram certificados nível F.

Deste modo, será realizada uma análise em todas as atividades que compõem os processos, averiguando onde seria preciso realizar alterações, caso a empresa optasse pelo método de desenvolvimento MDA. Assim, onde for identificada a necessidade de inserção do MDA, serão sugeridas modificações nos processos, baseadas na instância proposta no Capítulo 3 e nos conceitos do MDA.

Para analisar se estas modificações sugeridas não infringem o MPS.BR, os processos serão apresentados à especialistas de MPS.BR, que analisarão se este processo poderia manter a certificação já obtida com o novo método de desenvolvimento inserido.

4.1 Materiais e Métodos

Os materiais e métodos utilizados neste capítulo estarão expostos a seguir:

EPF- O EPF *Eclipse Process Framework* é uma ferramenta desenvolvida pela comunidade do Eclipse, comumente utilizada para a documentação e o gerenciamento de processos. Neste capítulo, o EPF é usado para a apresentação dos processos. Abaixo estão descritos três ícones do EPF que serão relevantes no decorrer deste capítulo.

- Fase: uma fase pode ser composta por atividades, tarefas e outras fases. O ícone da fase está representado na Figura 29.

Figura 29 – Ícone da Fase. Fonte: ferramenta EPF

- Atividade: uma atividade contém várias tarefas, podendo conter também outras atividades dentro dela. O ícone da atividade está representado na Figura 30.

Figura 30 – Ícone de Atividade. Fonte: ferramenta EPF

- Tarefa: uma tarefa é composta por vários passos. O ícone que representa a tarefa é apresentado na Figura 31.

Figura 31 – Ícone de Tarefa. Fonte: ferramenta EPF

A ferramenta *Enterprise Architect* (EA) neste capítulo, será referenciada juntamente com os artefatos gerados por ela, já apresentados no capítulo anterior. Cabe ressaltar que a ferramenta utilizada não precisa ser necessariamente o EA, no entanto, precisa ser uma ferramenta que suporte o MDA, mas nesse capítulo os exemplos dados utilizando ferramentas serão baseados no EA.

Os processos do MPS.BR, que foram cedidos pela empresa, serão utilizados como base para este capítulo. Os processos utilizados são correspondentes à Gerência de Requisitos (GRE) e à Gerência de Projetos (GPR) do MPS.BR nível G. Os processos serão analisados, sendo que para cada tarefa que precisar de alterações para contemplar o MDA, sugestões serão fornecidas. As tarefas em que os conceitos do MDA não são necessários, serão mantidas as definições descritas inicialmente pela organização.

A proposta MDA apresentada no Capítulo 3, juntamente com os seus artefatos gerados, serão utilizados para dar embasamento às sugestões realizadas nas tarefas dos processos.

Os conceitos base do MDA, presentes no guia e em seus conjuntos de normas, mesmo que não tenham sidos aplicados na proposta, serão utilizados para apoiar as sugestões realizadas nos processos.

4.2 Gerência de Requisitos- GRE

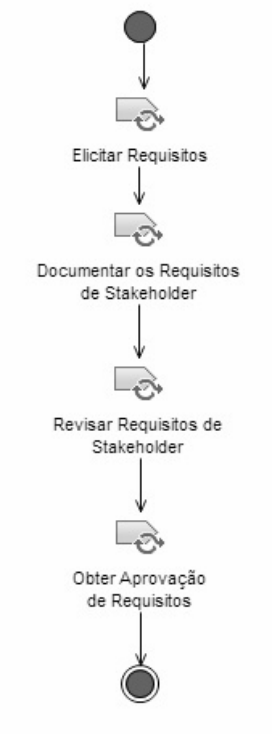
A Gerência de Requisitos tem como objetivo controlar a evolução dos requisitos, sendo que o processo deve gerenciá-los, incluindo requisitos funcionais e não-funcionais, bem como os estabelecidos ao projeto pela organização [83].

O processo da organização correspondente ao GRE é denominado Elicitação e Análise de Requisitos (EAR) e consiste em identificar os requisitos do cliente, analisá-los, derivá-los em requisitos do sistema, priorizá-los, documentá-los e verificá-los a fim de identificar inconsistências prejudiciais ao projeto. Este processo está apresentado na Figura 32.

Figura 32 – Fluxo de Atividades de GRE. Fonte: repositório da empresa.

A atividade de Levantar Requisitos é constituída pelas tarefas apresentadas na Figura 33.

Figura 33 – Tarefas da atividade Levantar Requisitos. Fonte: repositório da empresa.



As definições das tarefas que compõem a atividade Levantar Requisitos estão descritas a seguir:

- "Elicitar Requisitos": os requisitos são definidos ao longo da atividade de Levantamento de Requisitos. Geralmente estes são elicitados em entrevistas ou *workshops* de requisitos. Posteriormente, quando aqueles requisitos são usados para construir e verificar modelos, podem ser encontrados *gaps* (lacunas), que irão exigir que os detalhes dos requisitos sejam identificados.

- "Documentar os Requisitos de *Stakeholder*": consiste em registrar as informações obtidas junto ao cliente, de forma que possam ser consultadas a qualquer momento, conforme necessário.
- "Revisar Requisitos de *Stakeholder*": visa analisar pontos considerados críticos para a qualidade e sucesso do projeto, ou seja, nas informações capturadas diretamente do cliente.
- "Obter Aprovação de Requisitos": visa obter um posicionamento do cliente em relação aos requisitos identificados.

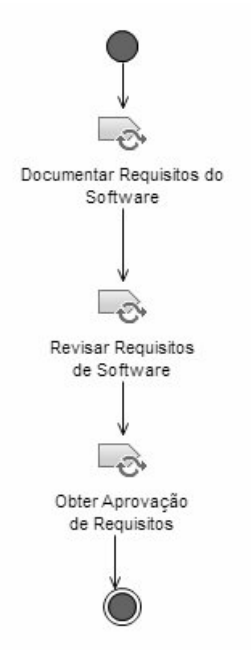
Das tarefas que compõem a atividade Levantar Requisitos, somente a tarefa Documentar os Requisitos de *Stakeholder* precisa ser alterada para atender os conceitos do MDA. Foi analisado que esta tarefa corresponde a dois resultados esperados do MPS.BR: o GRE1 e o GRE3. Os tópicos explicam sobre o que é executado na empresa, já os subtópicos são os conceitos do MDA sugeridos. Portanto, os conceitos propostos são:

- A empresa possui uma especificação técnica, que apresenta diversos itens para serem especificados na documentação dos requisitos do *Stakeholders*. Vale ressaltar que, dentre esses itens, é determinada a elaboração do caso de uso e que a empresa já utilizava a ferramenta EA para elaboração dos diagramas.
  - Apesar de a empresa determinar a elaboração do caso de uso, ela não especifica regras para elaboração do mesmo. Portanto, isso pode causar modelagens diferentes entre os colaboradores, sendo que isso para o MDA já impossibilita as transformações. Logo, para documentar os requisitos de *Stakeholder*, deve-se utilizar diagrama de caso de uso, cenários de caso de uso e interface de tela, utilizando a metodologia apresentada no Capítulo 3 e os conceitos UML [2] para a elaboração dos mesmos. Com isso, além de modelos padronizados que possibilitam as transformações pelo MDA, têm-se modelos com informações relevantes que irão ser utilizados, além da documentação. O cenário de caso de uso, por exemplo, não é utilizado só para documentação ou entendimento do sistema. De acordo com o que foi estabelecido, através deste modelo é possível extrair diversas informações e gerar dois diagramas: o diagrama de classe e o de sequência. A documentação de todos esses diagramas deve ser feita no EA.
- A empresa para cumprir o GPR3, que é referente a instaurar a rastreabilidade bidirecional em seu processo, determina o estabelecimento da rastreabilidade manualmente entre os requisitos de negócios e dos usuários.
  - Com o MDA, a rastreabilidade vem de maneira intrínseca, pois pelas transformações das fases (CIM-PIM-PSM-Código) é possível rastrear os requisitos[1].

Portanto, este quesito a empresa cumpriria somente por ter o MDA como metodologia de desenvolvimento. Outro meio de executar a rastreabilidade é pela ferramenta utilizada, pois o EA gera, de maneira automática, a matriz de relacionamento, que permite analisar a rastreabilidade dos requisitos a partir dos modelos documentados.

A atividade de Análise de Requisitos é composta pelas seguintes tarefas expostas na Figura 34.

Figura 34 – Tarefas da atividade Análise de Requisitos. Fonte: repositório da empresa.



As definições das tarefas que constituem a atividade Analisar Requisitos estão apresentadas a seguir:

- "Documentar os Requisitos do Software": esta tarefa de derivação tem como finalidade traduzir os Requisitos de *Stakeholder* em Requisitos de Software, onde ficarão mais claras as funcionalidades (requisitos funcionais) no nível de sistema, assim como os comportamentos (requisitos não-funcionais) que o sistema deverá possuir.
- "Revisar Requisitos de Software": o procedimento de revisão dos requisitos visa analisar pontos considerados críticos para a qualidade e sucesso do projeto, mas com foco nos Requisitos de Software, ou seja, nas informações derivadas dos Requisitos de Usuário.
- "Obter Aprovação de Requisitos": essa aprovação é um marco de importância para o projeto, pois a partir desta aprovação garante-se que há um entendimento comum entre as partes. A partir deste momento, as mudanças nos requisitos são tratadas formalmente por solicitações de mudança.

Dentre as tarefas que constituem o Analisar Requisitos, foi observado que somente a tarefa Documentar os Requisitos do Software precisa ser modificada para atender aos conceitos MDA. Esta, corresponde novamente aos resultados esperado GRE1 e GRE3 do MPS.BR nível G. Os tópicos explicam sobre o que é executado na empresa, já os subtópicos são os conceitos do MDA sugeridos. Os conceitos propostos são:

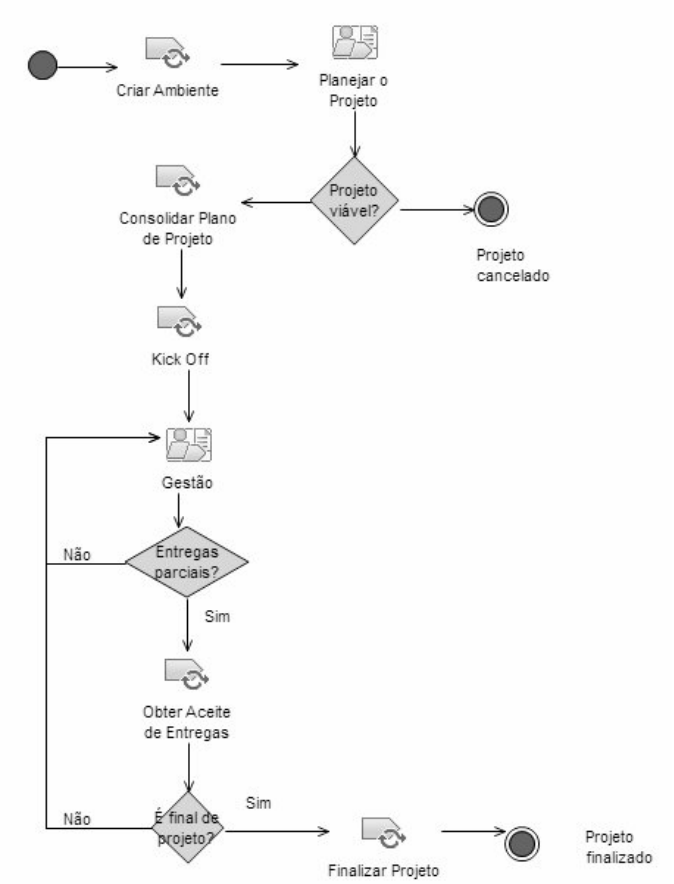
- Para documentar os requisitos do sistema é utilizada a mesma especificação técnica para requisitos do *Stakeholder* e, portanto é determinado a elaboração do caso de uso (neste caso também é requisitado a interface de tela).
  - Esta atividade possui o mesmo problema que o processo anterior. Para o caso de uso e para a interface de tela não existe uma especificação de como estes devem ser desenvolvidos, podendo gerar modelos diferentes. Logo, para documentar os requisitos funcionais do sistema deve-se utilizar diagrama de caso de uso, cenários de caso de uso e interface de tela, utilizando a metodologia apresentada no Capítulo 3 e conceitos da UML [2]. Com isso, além de modelos padronizados que possibilitam as transformações pelo MDA, têm-se modelos com informações relevantes que irão ser utilizados, além da documentação. A interface de tela, por exemplo, não será utilizada somente para validação com o cliente. A interface pode trazer atributos de classes que serão extraídos a partir das regras. A documentação de todos esses diagramas deve ser realizadas na ferramenta utilizada.
- Esta tarefa é referente a rastreabilidade bidirecional (GRE3), sendo que a empresa determina o estabelecimento da rastreabilidade de forma manual entre os Requisitos de Usuário e os Requisitos de Sistema.
  - Como já foi descrito anteriormente, com o MDA como metodologia de desenvolvimento, a rastreabilidade bidirecional é algo inerente. Portanto, a empresa já executaria este quesito só de adotar o MDA. Como já ressaltado, a ferramenta utilizada possui a funcionalidade de gerar uma matriz de relacionamento, que apresenta a rastreabilidade entre os requisitos dos modelos documentados.

4.3 Gerência de Projetos

A Gerência de Projetos (GPR) envolve diversas atividades correspondentes aos projetos, desde elaborar um ciclo de vida até ações corretivas. No nível G, o GPR é composto por 19 resultados esperados, sendo que este processo evolui nos níveis E e B.

O processo da empresa também é denominado de Gerência de Projetos e está apresentado na Figura 35.

Figura 35 – Processo da Gerência de Projetos. Fonte: repositório da empresa.



As definições das atividades e tarefas que compõem o Gerência de Projetos estão apresentadas a seguir:

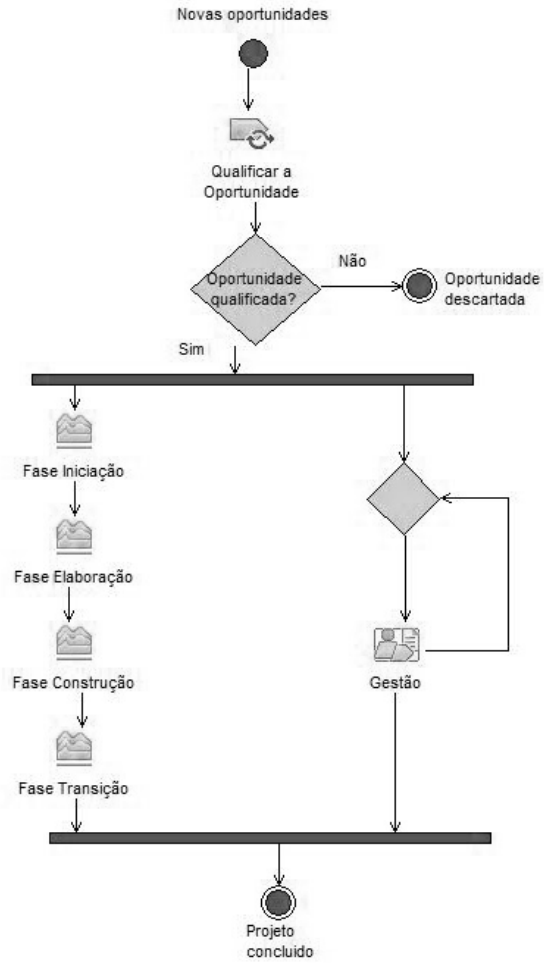
- a tarefa "Criar Ambiente" visa criar todo ambiente e configuração necessárias para documentar e acompanhar a oportunidade;
- a atividade "Planejar Projeto" visa planejar os processos, as tarefas, os recursos, as aquisições, analisar reutilização para minimizar os esforços, determinar o tamanho do projeto e, por fim, determinar o custo;
- a tarefa "Consolidar o Plano de Projeto" consiste em finalizar o preenchimento dos capítulos restantes, fazendo o devido ligamento com demais planos necessários;
- o "*Kick off*" consiste em apresentar os objetivos do projeto, o papel e responsabilidade de cada um no projeto;
- atividade de "Gestão" concentra as tarefas relacionadas ao monitoramento e controle do projeto;
- a tarefa "Obter o aceite das entregas" baseia-se em obter um posicionamento do cliente sobre a entrega realizada;

- a última tarefa do processo “Finalizar Projeto” consiste em fechar todos os itens do projeto que ainda possam estar abertos.

Para o processo de Gerência de Projetos não houve a necessidade de modificações para a inserção dos conceitos do MDA. Não haviam atividades/tarefas em que eram necessárias modificações para contemplar o MDA.

O ciclo de vida é um resultado esperado do MPS.BR, mais especificamente é definido em GPR3 no guia [83]. A empresa define seu ciclo de vida separadamente do seu processo. O ciclo de vida adotado pela organização está apresentado na Figura 36.

Figura 36 – Ciclo de vida da Organização. Fonte:repositório da empresa..



As definições de cada elemento que o compõe o ciclo de vida da Figura 36 são:

- a tarefa “Qualificar a Oportunidade” consiste em analisar os critérios de avaliação e determinar a classificação da oportunidade perante as demais;
- a fase de “Iniciação” corresponde à obtenção inicial das informações para possibilitar o planejamento de projeto. Nesta fase, são realizadas as atividades/tarefas referentes ao levantamento de requisitos;

- a fase de “Elaboração” refere-se à elaboração dos detalhes necessários para construção do projeto;
- a fase de “Construção”, como o próprio nome já menciona, corresponde à construção do projeto;
- na fase de “Transição” são realizados os testes e a entrega do projeto;
- a atividade de “Gestão” concentra as tarefas relacionadas ao monitoramento e controle do projeto.

As modificações que serão sugeridas para o ciclo de vida ocorrerão nas fases de iniciação, elaboração e construção. Para apresentar essas modificações, os tópicos explicam sobre as fases do ciclo de vida executadas na empresa, já os subtópicos são os conceitos do MDA sugeridos para estas fases.Os conceitos propostos são:

- Na fase de iniciação, os requisitos são levantados, como já foi mostrado anteriormente neste trabalho, no processo de Elicitação e Análise de Requisitos. Portanto, sabe-se que a empresa executa a documentação dos requisitos (*stakeholder* e sistema) em casos de uso, mas, no entanto, não utiliza de especificações para a elaboração dos mesmos.
  - Esta fase do ciclo de vida será correspondente à fase CIM do MDA. Assim, nesta fase serão levantados os requisitos, documentados em caso de uso, cenários de caso de uso e interface de tela, sendo utilizada a metodologia de desenvolvimento dos modelos apresentada no Capítulo 3 e conceitos da UML [2].
- Na fase de elaboração, a empresa executa diversas ações, como por exemplo, planejar a configuração, a qualidade, a elaboração dos planos de testes e também executa a tarefa de projetar arquitetura e módulo do sistema. Nestas atividades de projetar arquitetura e módulo do sistema são elaborados diagramas de sequência e de classe.
  - O que irá ser modificado na fase de elaboração serão somente as tarefas de projetar arquitetura e módulo do sistema. As outras ações não sofrerão alterações. Estas tarefas modificadas serão correspondentes à fase PIM do MDA. Portanto o PIM deve ser fruto da derivação do CIM que, consequentemente, irá gerar o diagrama de classe e o de sequência, utilizando as regras estabelecidas no Capítulo 3. O PIM pode passar por uma evolução para agregar mais funcionalidades que não foram abordadas com a transformação (CIM-PIM). Desta maneira, um PIM pode ser evoluído para outro PIM.

- A fase de construção executada pela organização consiste em implementar códigos de acordo com a documentação gerada durante toda a fase de análise.

  - Esta fase do ciclo de vida será correspondente à fase PSM do MDA. Portanto, os modelos PIM serão transformados automaticamente (via ferramenta) para modelos PSM, os quais também serão transformados automaticamente para o código. O código que não foi gerado deve ser adicionado manualmente.

4.4 Considerações Finais

Neste capítulo foram apresentadas sugestões de conceitos MDA para processos de uma organização que já possuía certificação MPS.BR. Os processos que foram analisados neste capítulo, eram correspondentes à Gerência de Requisitos e à Gerência de Projetos do MPS.BR nível G. Assim, apontou-se onde em um processo, já certificado, haveria a necessidade de modificações para se instaurar uma abordagem de desenvolvimento orientado a modelo.

O processo correspondente à Gerência de Requisitos da organização possuiu poucas modificações sugeridas para a inserção dos conceitos MDA. O conceito que foi proposto era a documentação dos requisitos (*Stakeholder* e Sistema) em diagramas de casos de uso, cenários de casos de uso e interface de tela.

O processo referente à Gerência de Processos da empresa não passou por modificações. No entanto, o ciclo de vida da organização teve mudanças em suas fases. As fases de iniciação, elaboração e construção foram modificadas para as fases do MDA, que são CIM, PIM e PSM respectivamente.

Em ambos os processos pode-se notar que não foram realizadas muitas sugestões de modificações. Portanto, pode-se constatar que instituições que já possuem certificação do MPS.BR nível G, não passariam por mudanças bruscas em seus processos para adotarem o MDA como método de desenvolvimento.

As vantagens que as organizações poderiam encontrar com a incorporação do MDA, quando as mesmas já possuem processos MPS.BR, são a adição dos benefícios do MDD/MDA, além de minimizar alguns problemas encontrados com a implementação e manutenção do modelo de referência brasileiro.

Conforme Cunha et al. [13] determinadas atividades propostas pelo MPS.BR não são executadas como deveriam. Com a inserção do MDA, têm-se a necessidade de implementar e executar todas as fases e atividades propostas, pois isso possibilita as transformações entre as fases.

O problema de alta rotatividade da equipe [17], [9] e [15] pode gerar um tempo de aprendizagem do método utilizado para desenvolvimento. No entanto, a linguagem

utilizada é bem estabelecida, logo, essa rotatividade não influencia na geração de modelos, pois o BNF normatiza a maneira de redigir os modelos de CIM e isso implica diretamente no código gerado.

A ausência de ferramentas para apoiar os processos [17] é atenuada com o uso do MDA, pois este possui diversas ferramentas, que além de realizar as transformações automatizadas de um modelo para outro ou para o código, possui características que podem suprir determinados pontos do MPS.BR [49].

A subjetividade do MPS.BR [13] é minimizada, pois o fato de exigir um determinado formalismo para a geração das orações, assim como para a geração dos diagramas de classe e de sequência, pode minimizar o processo de subjetividade.

De acordo com Cunha et al. [13] o MPS.BR não pensa na produtividade da empresa, no entanto o MDA é focado em produtividade, reaproveitamento e transformações automatizadas. Assim, com o MDA, busca-se atender esta dificuldade.

5 AVALIAÇÃO DOS ARTEFATOS PARA O MPS.BR

Este capítulo apresenta uma pesquisa, que visa analisar, sobre uma perspectiva de especialistas em qualidade de software, o quão a instância desenvolvida e os conceitos do MDA podem agregar valor para o MPS.BR.

O método utilizado para a pesquisa foi o questionário. Este, tinha como objetivo analisar duas propostas, sendo que essas propostas avaliadas seriam base para análise das hipóteses levantadas neste trabalho.

Um dos objetivos desta pesquisa foi analisar se as modificações sugeridas nos processos do MPS.BR nível G para a inserção dos artefatos da instância e os conceitos do MDA, realizadas no Capítulo 4, eram válidos e continuam atendendo o MPS.BR, além de verificar quais são os benefícios e desvantagens com esta nova abordagem estabelecida no processo.

Outra análise realizada pelo questionário tem por objetivo verificar se os conceitos e artefatos gerados pela instância MDA contribuem para alcançar resultados esperados do MPS.BR.

Portanto, neste capítulo serão apresentadas as repostas dos especialistas para as perguntas do questionário, uma síntese de cada conjunto de respostas, e analise e correlação com as devidas propostas.

5.1 Materiais e Métodos

O material utilizado para realizar a avaliação foi um questionário, o qual foi elaborado em duas partes. O questionário ao todo é composto por 30 questões, sendo que o mesmo está exposto no Anexo B desta dissertação. A primeira parte foi direcionada para analisar os processos que foram modificados com a incorporação das sugestões do MDA, sendo avaliado também se esta abordagem inserida traria benefícios e/ou desvantagens.

Já na segunda parte, primeiro foi realizado uma análise de todos os processos do MPS.BR e todos os resultados esperados que poderiam ser antecipados pelos conceitos e artefatos do MDA. Após este levantamento, foram listados todos os possíveis resultados esperados do MPS.BR que o MDA poderia auxiliar.

Este questionário é uma pesquisa qualitativa [84] e foi aplicado a três especialistas da área de qualidade de software. O questionário foi aplicado presencialmente no ambiente de trabalho dos especialistas, o SENAI-Londrina (Serviço Nacional de Aprendizagem Industrial). O SENAI de Londrina é uma instituição que já implantou 63 certificações do MPS.BR em empresas paranaenses. Os artefatos gerados neste trabalho e conceitos do

MDA foram apresentados previamente e simultaneamente para possibilitar ao entrevistado responder as perguntas presentes no questionário.

5.2 Aplicação do Questionário

Para aplicação do questionário foram selecionados três especialistas da área de qualidade de software, também conhecedores do modelo de referência MPS.BR. A qualificação de cada especialista está especificada na tabela 7.

Tabela 7 – Especialistas de Qualidade de Software

Especialista	Qualificação
Especialista 1	Implementador dos 3 modelos do MPS.BR (SW/ SV e RH). Especialista em gerenciamento de projetos. Especialista em Ciência da Computação. Docente nos cursos MBA em Gestão de Projetos e Gestão Individual. Possui 17 anos na área de TI e 7 anos com Gerência de Projetos e Consultor.
Especialista 2	Implementador MPS.BR Software e Serviços Credenciado. Especialista em gestão de projetos (PMP). Trabalhou 12 anos em uma fábrica de software CMMI. Implementação em mais de 20 empresas (MPS.BR, CMMI, ITMARK e MOPROSOFT).
Especialista 3	Trabalhou por 8 anos na área de qualidade de software, mais especificamente em desenvolvimento e melhoria de processos. Realizou o curso C1 do MPS.BR.

A aplicação do questionário foi realizada individualmente, sendo que este possui duas partes. Em cada parte foi apresentado artefatos da instância desenvolvida e conceitos do MDA, para que os especialistas pudessem conhecer o modelo e ter embasamento para responder certas perguntas.

A primeira parte é referente ao Capítulo 4 e a avaliação sobre as modificações realizadas nos processos. Portanto, nesta parte foi explicado o que era realizado antes na organização e as modificações que seriam realizadas para a inserção da instância elaborada de MDA. Sobre esse assunto foram elaboradas quatro questões abertas, que questionavam sobre a validade dessas modificações, vantagens e desvantagens da inserção de um MDA em um processo MPS.BR.

A segunda parte do questionário consiste em analisar se a instância elaborada e os conceitos do MDA podem antecipar resultados esperados (RE) do MPS.BR de diversos níveis de maturidade. Assim sendo, foram listados 23 RE do MPS.BR que poderiam ter relação com o MDA. Os especialistas poderiam responder para estas questões sim, parcialmente ou não, sendo possível comentar cada reposta dada. Nesta parte também foram elaborada três perguntas abertas, para poder saber a opinião dos especialistas sobre a proposta apresentada.

As respostas de todas as perguntas do questionário, mesmo que expostas neste Capítulo, estão presentes no Apêndice C.

5.2.1 Questionário Parte I

Nesta seção serão apresentadas as repostas para as questões da parte I do questionário. Como as questões elaboradas eram dissertativas, as respostas dos especialistas serão sintetizadas.

1) As modificações apresentadas no processo GRE atendem o MPS.BR?

Os especialistas concordam que as mudanças realizadas no GRE continuam atendendo o MPS.BR. Eles ainda concluem que, essas modificações incluíram uma documentação melhor e mais detalhada.

2) As modificações realizadas no ciclo de vida apresentado continuam atendendo os quesitos do MPS.BR?

Sobre as modificações realizadas no ciclo de vida, todos os especialistas afirmam que as mudanças também continuam atendendo o que é pedido pelo MPS.BR. Um especialista ressalta a importância do ciclo de vida estar alinhada com o cronograma. Porém, as modificações realizadas não o afetam sendo que este alinhamento continua, o que será gerado deve ser especificado do mesmo modo como era realizado antes, porém seguindo as fases determinadas (CIM-PIM-PSM).

3) Em sua opinião, quais as desvantagens da inserção do MDA em um processo MPS.BR?

Nesta questão um dos especialistas relata não ver desvantagem com a implementação do MDA em um processo do MPS.BR. Este mesmo especialista alerta que a organização pode necessitar de treinamento para seus colaboradores aprender um novo modo de trabalho. Os outros dois especialistas já acreditam que o MDA pode de fato trazer algumas desvantagens, como burocratização e uma dificuldade de implantação, já que o MDA não possui níveis de maturidade, como o MPS.BR.

4) Em sua opinião, a inserção do MDA em um processo MPS.BR já instaurado, ofereceria benefícios ao processo? Se sim quais?

Sobre os benefícios que o MDA poderia oferecer a um processo do MPS.BR já instaurado, todos os especialistas concordaram que o MDA traz algum benefício. Pode-se perceber que os artefatos gerados e o conceito do MDA de transformação entre modelos são os maiores benefícios para o MPS.BR.

5.2.2 Questionário Parte II

Nesta seção serão apresentadas as respostas dos especialistas sobre os resultados esperados que a instância desenvolvida e os conceitos do MDA podem antecipar do MPS.BR. Essas repostas estão expostas na Tabela 8, sendo que, nesta tabela, estão listados todos os RE, e a resposta de cada especialista. As respostas possíveis são: Sim, Parcialmente e Não.

No Apêndice B está a especificação de cada artefato e/ou conceito apresentada simultaneamente para os resultados esperados da tabela. Esta especificação está entre parênteses diante do RE correspondente. No Apêndice D está uma breve explicação de todos os RE e qual seria a justificativa do mesmo ser antecipado pela instância elaborada ou conceito do MDA. No Apêndice C estão os comentários, quando especificados pelos especialistas, justificando ou esclarecendo a escolha de resposta.

Foi incluída a possibilidade de resposta“ Parcialmente”, pois vários resultados esperados do MPS.BR abordam diversos assuntos distintos em suas descrições. Por isso, existe a possibilidade de um(s) artefato(s) do MDA abordar uma certa parcela de um resultado esperado, porém não cumpri-lo totalmente.

Tabela 8 – Respostas dos Especialistas para os Resultados Esperados do MPS.BR

Resultado Esperado	Resposta 1	Resposta 2	Resposta 3
GPR3	Parcialmente	Sim	Parcialmente
GRE1	Sim	Parcialmente	Sim
GRE3	Sim	Sim	Sim
GRE5	Parcialmente	Parcialmente	Sim
MED2	Parcialmente	Não	Parcialmente
MED3	Parcialmente	Não	Não
MED4	Não	Não	Não
MED5	Parcialmente	Sim	Parcialmente
AMP7	Parcialmente	Sim	Sim
DFP4	Parcialmente	Sim	Sim
DRE1	Sim	Sim	Sim
DRE4	Sim	Sim	Sim
DRE6	Sim	Sim	Sim
ITP2	Sim	Sim	Sim
PCP3	Sim	Sim	Sim
PCP7	Sim	Sim	Sim
PCP8	Sim	Sim	Sim
VAL2	Sim	Parcialmente	Parcialmente
VAL3	Sim	Sim	Parcialmente
VAL4	Sim	Sim	Parcialmente
VER2	Parcialmente	Não	Parcialmente
VER4	Parcialmente	Parcialmente	Parcialmente
DRU7	Parcialmente	Não	Não

Nota-se que algumas respostas divergiram de um especialista para o outro. Isso se deve ao que o especialista julga como um resultado esperado cumprido totalmente, parcialmente ou não foi executado. No entanto, pode-se notar que não houve respostas do tipo “sim” e “ não” para o mesmo resultado esperado.

Ao todo foram 8 resultados esperados que obtiveram sim de todos os especialistas, totalizando 34,78% dos RE analisados. Já as respostas que foram categorizadas como parcialmente ou parcialmente e sim, tiveram um total de 10 RE, representando 43,47 %. Portanto, se analisarmos os resultados totalmente positivos e os que tiveram respostas parcialmente positivas têm-se um total de 18 RE, o que representa que 78,75% dos RE que foram analisados do MPS.BR são antecipados, mesmo que de forma parcial, pelos artefatos e conceitos da instância do MDA. Cabe ressaltar que não foram contabilizadas as respostas que obtiveram parcialmente e não para o mesmo RE.

A Tabela 9 apresenta uma síntese dos RE que foram listados e classificados como antecipados, integralmente ou parcialmente. Nesta tabela, está exposta a porcentagem e a quantidade total dos RE por nível de maturidade. Assim, poderá ter-se uma caracterização de qual nível de maturidade obteve mais RE classificados como antecipados pela proposta.

Tabela 9 – Porcentagem de RE por Nível de Maturidade

Nível de Maturidade	% dos RE	Quantidade de RE
G	17,39%	4
F	4,34%	1
E	8,65%	2
D	47,82%	11
C	0 %	0

Nota-se que os níveis de maturidade que obtiveram mais RE listados considerados antecipados, foram os níveis G e D. O nível C não teve nenhum RE antecipado. Os níveis A e B não foram considerados, pois eles tratam de melhoria contínua.

Ainda na Parte II do questionário, buscou-se analisar a opinião dos especialistas sobre a proposta apresentada, sendo que as respostas para as questões realizadas estão separadas por tabelas expostas a seguir.

1) Em sua opinião, o MDA apresenta artefatos/modelos que atendam resultados esperados no nível G do MPS.BR? Quais?

Para esta questão, os especialistas afirmam que os resultados esperados GRE1, GRE3 e GRE 5 são atendidos com o MDA. Um especialista ainda destaca que o GRE4 pode ser atendido. E outro o GPR3 que trata do ciclo de vida.

2) Em sua opinião, com a inserção do MDA no nível G do MPS.BR, irá ou não apresentar alguns resultados esperados do MPS.BR de níveis superiores? Justifique.

Esta questão pretende saber dos especialistas se o MDA inserido no nível G, apresentará alguns resultados esperados de níveis de maturidade superiores. Todos os especialista afirmam que sim, que alguns resultados esperados já serão alcançados só com a inserção do MDA no nível G. Destacam-se os processos citados de engenharia, no nível D de maturidade.

3) Em sua opinião, uma empresa que já possui o MDA como modelo de desenvolvimento, teria maior facilidade para implementar o MPS.BR do que uma empresa que execute o desenvolvimento orientado a código?

A última pergunta do questionário é a respeito da opinião dos especialistas sobre se uma organização que já executa o MDA possui maior facilidade para implementar um MPS.BR do que uma empresa que faça código. As repostas para estas questões foram que uma organização que possua um MDA terá sim mais facilidade, pois ela já possui certa metodologia, um vez que a estruturação de pontos do MDA são parecidos com o do MPS.BR, além de facilitar na implementação dos processos de engenharia.

5.3 Análises das Hipóteses

Para analisar se a hipótese H1 é verdadeira, foi necessário o desenvolvimento de algumas atividades. Primeiro foi elaborada uma instância de MDA com a definição dos seus artefatos. Após isso, foram adotados processos já certificados MPS.BR nível G. Esses processos foram analisados para verificar se haveria a necessidade de modificações para a adoção do MDA. Após essa análise realizada nos processos, foram sugeridas as modificações necessárias, baseada na instância e seus artefatos. Como meio de averiguar se essas modificações não infringiram o MPS.BR, foi realizada uma aplicação de questionário com especialistas.

Além das atividades desenvolvidas, deve-se considerar a seguinte condição:

"A hipótese H1 será considerada verdadeira se os especialistas confirmarem que um ou mais artefatos do MDA podem ser utilizados como evidências de resultados esperados do MPS.BR nível G."

Portanto, de acordo com as respostas obtidas com os especialistas, pode-se concluir que esta hipótese é considerada verdadeira, pois há resultados esperados do nível G que são antecipados com o MDA. Neste caso foram quatro. Isso aponta que o processo de implementação do nível G do MPS.BR é auxiliado com a adoção do MDA como método de desenvolvimento. Outro ponto importante, é que de acordo com as respostas do questionário, as modificações realizadas nos processos não violaram os princípios do MPS.BR, sendo que alguns especialistas ressaltaram que alguns artefatos novos trouxeram uma melhora na documentação.

Para analisar se a hipótese H2 é verdadeira, foi necessário o desenvolvimento de algumas atividades. Primeiro foi elaborada uma instância de MDA com a definição dos seus artefatos. Em seguida, foram estudados todos os processos e seus respectivos resultados esperados do MPS.BR. Após este estudo, foi realizada a seleção dos possíveis resultados esperados que poderiam ser antecipados pelo MDA. Esta seleção foi baseada na instância elaborada, seus artefatos e nos conceitos do MDA. Para analisar se estes resultados esperados eram realmente antecipados pelo MDA, realizou-se uma aplicação de questionário com especialista.

Além das atividades desenvolvidas, deve-se considerar a seguinte condição:

"A hipótese H2 será considerada verdadeira se existir resultados esperados alcançados de níveis superiores do nível G. "

A hipótese H2 pode ser considerada verdadeira a partir dos resultados esperados listados nas Tabelas 8 e 9. Como já apresentado anteriormente, mais de 78% dos resultados esperado do MPS.BR levantados são antecipados de forma integral ou parcial com a inserção do MDA, sendo que mais de 60% dos RE são de níveis de maturidade superiores ao G. Os especialistas também opinaram que acreditam que o MDA pode antecipar RE do MPS.BR. Portanto, pode-se concluir que a adoção do MDA como método de desenvolvimento contribui para antecipar resultados esperados de determinados processos do MPS.BR.

5.4 Considerações Finais

Neste capítulo foram apresentadas as respostas de cada especialista para o questionário aplicado. Este tinha como objetivo avaliar duas propostas: a primeira é sobre as modificações sugeridas nos processos MPS.BR realizadas no Capítulo 4, já a segunda foi sobre a possibilidade do MDA como método de desenvolvimento antecipar certos resultados esperados de processos do MPS.BR. Também foi apresentado neste capítulo a análise sobre as hipóteses levantadas neste trabalho.

Pode-se analisar que as considerações realizadas sobre os processos do Capítulo 4 foram positivas, sendo que as modificações realizadas continuam seguindo as premissas estabelecidas pelo MPS.BR. Outro ponto que pode ser destacado, é que os especialistas ressaltaram que os artefatos inseridos melhoraram o conteúdo dos documentos. Isso indica que a instância criada e os conceitos do MDA agregaram positivamente ao processo. Porém, destaca-se também que existe certa dificuldade em implantar um MDA, foi citado como exemplo, a burocratização e a falta de nivelamento, o que pode dificultar a adoção do mesmo.

Sobre os resultados esperados antecipados pela adoção da instância do MDA como método de desenvolvimento obteve-se um bom resultado. Mais de 78% dos RE listados

são antecipados de forma integral ou parcial. Isso apresenta que a adoção da instância contribui para a antecipação de determinados RE de diversos níveis de maturidade. Pode-se notar que houve mais RE com opiniões de antecipação parcial, porém como foi julgado por pelo menos um especialista que este RE não era antecipado, este RE não foi avaliado como antecipado.

Outra análise realizada foi sobre os RE antecipados e os níveis de maturidade em que eles se encontram. Os níveis que obtiveram maior número de RE categorizados como antecipados foram os níveis G e D. Como já comentado pelos especialistas, no nível D está o processos de engenharia, por isso têm diversos RE antecipados, já o nível G possui o processo de gerência de requisitos, este por sua vez é alcançado pelos artefatos gerados e conceitos do MDA.

Os especialistas também julgaram que a instância apresentada e os conceitos do MDA contribuem para os MPS.BR, podendo sim antecipar determinados RE. Também foi colocada uma questão hipotética sobre duas empresas, indagando qual teria maior facilidade de implementar o MPS.BR, sendo que uma organização implementa MDA e outra que desenvolva código. Foi pressuposto que a empresa que possui o MDA teria mais facilidade para implementar o MPS.BR. Portanto, conclui-se que a instância apresentada e os conceitos do MDA auxiliam na implementação do MPS.BR, antecipando certos resultados esperados.

Cabe ressaltar que os resultados obtidos foram embasados na instância apresentada no Capítulo 3. Portanto, caso a organização elabore sua própria instância, com artefatos gerados diferentes, pode haver diferenciação nos resultados.

6 CONCLUSÕES

Este trabalho teve por objetivo analisar o MDA como método de desenvolvimento e suas implicações nos processos do MPS.BR.

A partir das respostas obtidas e das análises realizadas das hipóteses pode-se concluir que a instância elaborada e os conceitos do MDA facilitam a implementação do MPS.BR, visto que os documentos/artefatos gerados no MDA são compatíveis com os sugeridos nos resultados esperado do modelo de referência. Isso exprime a relevância da elaboração dos mesmo, sendo que para o MDA esta documentação é necessária para geração do código.

Outro ponto analisado é sobre a antecipação de resultados esperado do MPS.BR. Pode-se concluir que a partir dos artefatos gerados da instância e dos conceitos do MDA, pode-se antecipar certos resultados esperados de diversos níveis de maturidade. Os níveis G e D obtiveram maior quantidade de RE antecipados. Portanto, uma organização terá uma maturidade maior e, consequentemente, a evolução do nível de maturidade da empresa será mais facilmente alcançado com a adoção do MDA.

Conforme a análise dos especialistas, os artefatos e conceitos do MDA podem ser utilizados para compor os resultados esperados do MPS.BR. Assim, espera-se que o MDA contribua da seguinte maneira:

- a implantação de um MDA, que é uma abordagem de MDD, pode oferecer para uma organização as vantagens descritas de um MDD (produtividade, portabilidade, manutenção, reutilização e corretude), assim como auxiliar na certificação dos seus processos, visto que os artefatos poderão ser empregados para trazer as evidências do MPS.BR. Portanto, a adoção do MPS.BR será facilitada se, a implementação, tanto do modelo de referência quanto do MDA, ocorrerem em conjunto;

- uma empresa que implemente um MDA antes da implantação do MPS.BR possuirá maior facilidade futura de conquistar a certificação, pois os processos já apresentam artefatos úteis ao MPS.BR;

- uma organização que opte por implementar o MDA após o MPS.BR, terá que adaptar seus processos para esta nova metodologia.

A elaboração de uma instância de MDA foi uma atividade complexa. Assim a adoção de um MDA em uma empresa precisará de esforços iniciais. Devido a este fato, a organização deve cogitar se, a implementação de um MDA é adequada a sua realidade.

Porém, destaca-se que existem diversas ferramentas que dão suporte MDA e podem auxiliar neste esforço inicial. Como alternativa, ao invés de criar sua própria instância, as organizações podem adotar instâncias pré-definidas.

Os trabalhos de Valenzuela e Pavlich-Mariscal [63], Vasconcelos et al. [64], Li et al. [65], Rios et al [66] e Vasconcelos et al. [67] apresentam o MDD/ MDA inseridos na definição de processos ou na elaboração de modelos de maturidade, baseados no modelo de referência CMMI. Já neste trabalho buscou-se analisar se a instância elaborada teria implicações sobre os processos do MPS.BR. Portanto, pode-se destacar que, além de empregar um modelo de referência diferente dos artigos apresentados buscou-se demonstrar se com a implantação de uma instância MDA, haveria antecipação de certos resultados de determinados processos do MPS.BR.

Há artigos que utilizaram o SCAMPI [63], ou uma variação dele [64], para avaliar se os processos propostos estavam em conformidade com o modelo de referência. Neste trabalho, o método de avaliação foi por meio da análise do questionário com especialistas que implantam o modelo de referência. No entanto, foram usados processos neste trabalho que já possuíam certificação. O que foi avaliado pelo questionário, foram as modificações realizadas nos processos para a inserção dos artefatos e conceitos do MDA.

Os trabalhos de Vasconcelos et al. [64] e [67] apontam a necessidade de analisar a conformidade de abordagens MDD com qualquer modelo de referência. Pode-se concluir que este apontamento foi realizado neste trabalho, visto que foi utilizado uma abordagem de MDD, o MDA, e o modelo de qualidade empregado foi diferente dos utilizados nestes artigos, no caso o MPS.BR.

Como contribuição desta dissertação destaca-se a elaboração de um CIM, com a definição dos artefatos e de uma sintaxe BNF, sendo que esta sintaxe possibilita uma estruturação e uma normatização dos modelos de cenários de caso uso. A partir da BNF também foi possível à criação de regras de mapeamento dos modelos, viabilizando assim as transformações dos modelos CIM para PIM.

Outra contribuição foi a análise realizada sobre os processos de desenvolvimento de software e qualidade. Com isso, pode-se analisar quais mudanças foram necessárias sobre um processo já certificado MPS.BR nível G, para a adoção da instância MDA elaborada. Essas modificações foram analisadas pelos especialistas de MPS.BR, sendo julgado que estes processos, mesmo após as modificações, continuam atendendo o MPS.BR. Desse modo, pode-se concluir que os processos de MPS.BR podem ser adaptados e empregados artefatos de uma instâncias de MDA, por consequência englobará os benefícios do MDA.

Pode-se concluir que a instância elaborada contribui para a implementação do MPS.BR, visto que após a análise houve a constatação que atende os requisitos do MPS.BR nível G e antecipam resultados esperados do modelo de referência brasileiro.

Portanto, a instância elaborada contribui para antecipar quesitos de qualidade de diversos níveis de maturidade do MPS.BR.

No Capítulo 2 foram relatadas algumas dificuldades com relação à implantação e manutenção do MPS.BR. Com a proposta desta dissertação e as análises realizadas, pode-se averiguar que haverá contribuição para algumas dessas dificuldades pelas seguintes razões:

- documentação excessiva [13] e [9]: a documentação ainda continuará sendo excessiva, porém como ela resultará em código, esta não será vista como uma dificuldade, mas sim como um meio de desenvolvimento;

- dificuldade em executar a rastreabilidade bidirecional [16]: a rastreabilidade com o MDA é possibilitada através das transformações entre as fases (CIM-PIM-PSM-Código), sendo que algumas ferramentas de MDA proporcionam esta funcionalidade;

- dificuldade no gerenciamento das mudanças nos requisitos – GRE5 [16]: como pode ser analisado no Capítulo 5, foi considerado que o gerenciamento de mudanças é antecipado de forma parcial pelo MDA. Portanto, pode-se concluir que o MDA auxilia no gerenciamento de mudanças, pois estas devem ser executadas no CIM e serão propagadas para as próximas fases. Isso implica em ter sempre uma documentação em conformidade com o código;

- falta de ferramentas [16], [13] e [17]: como já citado anteriormente, o MDA possui mais de 50 ferramentas, sendo que estas podem conter diversas funcionalidades, além de somente modelar e executar transformações. Alguns exemplos de possíveis funcionalidades são: testes, métricas, validação e verificação de modelos, matriz de rastreabilidade e etc [49]. Portanto, algumas possíveis dificuldades do MPS.BR em relação às ferramentas podem ser supridas pelas ferramentas do MDA;

- subjetividade do modelo de referência [13]: essa subjetividade é atenuada quando se utiliza o MDA, visto que o mesmo é composto por etapas muito bem definidas e regras para viabilizar as transformações automatizada.

Apesar das vantagens discorridas sobre o MDA como método de desenvolvimento pode-se destacar desvantagens ocorridas por uma implementação do MPS.BR com o MDA. Um exemplo seria uma burocratização maior do processo e a dificuldade de pequenas empresas adotarem o MDA, pois ele não possui um nivelamento.

Outra desvantagem identificada foi a dificuldade em realizar as transformações automatizadas de modelo para modelo, como por exemplo, os diagramas comportamentais para código, requisitando muitas vezes um mapeamento manual ou o desenvolvimento das

próprias ferramentas. A ferramenta utilizada não possibilitava esta conversão. Pode-se notar que este tipo de transformação não é usual nas ferramentas.

Vale ressaltar que o MDA produz vários modelos até concluir em código. Em muitos casos se faz necessário a complementação do código fonte. Este trabalho não buscou a geração de um executável a partir de um MDA, mas sim contribuir nos desenvolvimento dos processos para alcançar a qualidade com o MPS.BR. Desta marneira, pode-se concluir que instância de MDA desenvolvida auxiliará na implementação dos processos do MPS.BR.

6.1 Trabalhos Futuros

Como trabalhos futuros e complementares a este trabalho propõe-se a implementação de uma ferramenta, a qual viabilize a concepção do CIM proposto nesta dissertação, sendo possível realizar a conversão automatizada de CIM para PIM.

Sugere-se também a elaboração de uma ferramenta que viabilize o mapeamento de diagramas comportamentais em código.

Objetiva-se ainda como trabalho futuro, a implantação de um MDA juntamente com o MPS.BR em uma organização, e a constatação sobre as dificuldades e benefícios alcançados.

REFERÊNCIAS

[1] KLEPPE, A. G.; WARMER, J. B.; BAST, W. *MDA explained: the model driven architecture: practice and promise*. [S.l.]: Addison-Wesley Professional, 2003.

[2] UML, O. *2.4. 1 superstructure specification*. [S.l.], 2011.

[3] SOFTEX. *MPS.BR - Melhoria de Processo do Software Brasileiro- Guia Geral MPS de Software*. [S.l.: s.n.], 2016.

[4] SEI. *CMMI for Development, Version 1.3, Technical Report*. [S.l.: s.n.], 2010.

[5] AYDAN, U.; YILMAZ, M.; O'CONNOR, R. V. Towards a serious game to teach iso/iec 12207 software lifecycle process: An interactive learning approach. In: *Software Process Improvement and Capability Determination*. [S.l.]: Springer, 2015. p. 217–229.

[6] WAZLAWICK, R. S. *Engenharia de software: conceitos e práticas*. [S.l.: s.n.], 2013.

[7] WEBER, K. et al. Melhoria de processo do software brasileiro (mps. br): um programa mobilizador. In: *Proceedings of the XXXI Conferencia Latinoamericana de Informatica (CLEI 2006). Santiago, Chile: agosto*. [S.l.: s.n.], 2006.

[8] LEAL, G. C. L. et al. Empirical study about the evaluation of the implantation of mps. br in enterprises of paraná. In: IEEE. *Informatica (CLEI), 2012 XXXVIII Conferencia Latinoamericana En*. [S.l.], 2012. p. 1–9.

[9] RODRIGUES, J. F.; KIRNER, T. G. Benefícios, fatores de sucesso e dificuldades da implantação do modelo mps. br. *IX Simpósio Brasileiro de Qualidade de Software*, 2010.

[10] KALINOWSKI, M. et al. From software engineering research to brazilian software quality improvement. In: IEEE. *Software Engineering (SBES), 2011 25th Brazilian Symposium on*. [S.l.], 2011. p. 120–125.

[11] KALINOWSKI, M. et al. Results of 10 years of software process improvement in brazil based on the mps-sw model. In: IEEE. *Quality of Information and Communications Technology (QUATIC), 2014 9th International Conference on the*. [S.l.], 2014. p. 28–37.

[12] FERREIRA, A. I. F. et al. Applying iso 9001: 2000, mps. br and cmmi to achieve software process maturity: Bl informatica's pathway. In: IEEE. *29th International Conference on Software Engineering (ICSE'07)*. [S.l.], 2007. p. 642–651.

[13] CUNHA, I. B. de A.; DIAS, K. J. A. N.; JÚNIOR, J. H. C. R. Dificuldades encontradas na implementação mps. br nível g: Estudo de caso. *e-xacta*, v. 4, n. 3, 2011.

[14] DUARTE, C. H. C. On the relationship between quality assurance and productivity in software companies. In: ACM. *Proceedings of the 2nd International Workshop on Conducting Empirical Studies in Industry*. [S.l.], 2014. p. 31–38.

[15] SANTANA, A. F. L. *Problemas em Iniciativas de Melhoria de Processos de Software sob a Ótica de uma Teoria de Intervenção*. Tese (Doutorado) — Dissertação de mestrado. Centro de Informática da Universidade Federal de Pernambuco (CIn-UFPE), 2007.

[16] ALMEIDA, C. D. A. de. Continuidade da execução dos processos de software em empresas avaliadas no mps. br. 2011.

[17] ROCHA, A. R. et al. Fatores de sucesso e dificuldades na implementação de processos de software utilizando o mr-mps e o cmmi. *PROQUALITY–Qualidade na Produção de Software, apresentado no I Encontro de Implementadores de MPS. BR*, p. 13–18, 2005.

[18] VÖLTER, M. et al. *Model-driven software development: technology, engineering, management*. [S.l.]: John Wiley & Sons, 2013.

[19] MERNIK, M.; HEERING, J.; SLOANE, A. M. When and how to develop domain-specific languages. *ACM computing surveys (CSUR)*, ACM, v. 37, n. 4, p. 316–344, 2005.

[20] BHANOT, V. et al. Using domain-specific modeling to develop software defined radio components and applications. In: *5th OOPSLA Workshop on Domain-Specific Modeling (DSM'05), San Diego, California, USA*. [S.l.: s.n.], 2005.

[21] PARVIAINEN, P. et al. Model-driven development processes and practices. *VTT Technical Research Centre of Finland*, 2009.

[22] OMG. *MDA Guide version 2.0*. 2014.

[23] FUGGETTA, A. Software process: a roadmap. In: ACM. *Proceedings of the Conference on the Future of Software Engineering*. [S.l.], 2000. p. 25–34.

[24] UNTERKALMSTEINER, M. et al. Evaluation and measurement of software process improvement—a systematic literature review. *IEEE Transactions on Software Engineering*, IEEE, v. 38, n. 2, p. 398–424, 2012.

[25] TSUI, F.; KARAM, O.; BERNAL, B. *Essentials of software engineering*. [S.l.]: Jones & Bartlett Publishers, 2013.

[26] VASCONCELOS, A. M. L. de et al. Introdução à engenharia de software e à qualidade de software. 2006.

[27] SILVA, F. S. et al. Using cmmi together with agile software development: A systematic review. *Information and Software Technology*, Elsevier, v. 58, p. 20–43, 2015.

[28] SANTOS, G. et al. Indicadores da implementação do nível e do mr-mps em uma instituição de pesquisa. *VIII Simpósio Brasileiro de Qualidade de Software*, p. 382–389, 2009.

[29] FILHO, R. C. S. et al. A experiência na implantação do processo de gerência de reutilização no laboratório de engenharia de software da coppe/ufrj. *ProQuality (UFLA)*, v. 4, p. 21–26, 2008.

[30] WEBER, K. et al. Mps. br-melhoria de processo do software brasileiro: resultados alcançados e lições aprendidas (2004-2008). *Santa Fé, Argentina*, v. 8, 2008.

[31] SCHOTS, N. C. L. et al. Lições aprendidas em implementações de melhoria de processos em organizações com diferentes características. In: *Anais do VII Workshop Anual do MPS. BR–WAMPS, Campinas–SP*. [S.l.: s.n.], 2011.

[32] LEAL, G. C. L. et al. Empirical study about the evaluation of the implantation of mps. br in enterprises of paraná. In: IEEE. *Informatica (CLEI), 2012 XXXVIII Conferencia Latinoamericana En*. [S.l.], 2012. p. 1–9.

[33] CORGOSINHO, C. C. Como iniciar e acompanhar um programa de implantação do mps. br. *ProQualiti–Qualidade na Produção de Software*, p. 23, 2006.

[34] TSUKUMO, A. N.; MARTINO, W. R. D.; PASSAGNOLO, M. Lições aprendidas na aplicação do método coop-mps para projetos cooperativos de melhoria de processo de software com mps. br. *ProQualiti–Qualidade na Produção de Software*, v. 2, p. 51–56, 2006.

[35] TORCHIANO, M. et al. Relevance, benefits, and problems of software modelling and model driven techniques—a survey in the italian industry. *Journal of Systems and Software*, Elsevier, v. 86, n. 8, p. 2110–2126, 2013.

[36] AMBLER, S. W. Agile model driven development is good enough. *Software, IEEE*, IEEE, v. 20, n. 5, p. 71–73, 2003.

[37] HAILPERN, B.; TARR, P. Model-driven development: The good, the bad, and the ugly. *IBM systems journal*, International Business Machines Corporation, v. 45, n. 3, p. 451, 2006.

[38] THOMAS, D. Mda: Revenge of the modelers or uml utopia? *Software, IEEE*, IEEE, v. 21, n. 3, p. 15–17, 2004.

[39] KONTIO, M. Architectural manifesto: Choosing mda tools. *IBM.[Online]. Available: http://www. ibm. com/developerworks/library/wiarch18. html.[Accessed: visited October 2009]*, 2005.

[40] MAGRI, J. A. Arquitetura dirigida a modelos (mda): Utilizando modelos no desenvolvimento de sistemas. *Augusto Guzzo Revista Acadêmica*, n. 8, p. 29–43, 2008.

[41] OMG. *MDA Guide version 1.0*. 2003.

[42] MELLOR, S. J. *MDA distilled: principles of model-driven architecture*. [S.l.]: Addison-Wesley Professional, 2004.

[43] CALIARI, G. L. P. *Transformações e mapeamentos da MDA e sua implementação em três ferramentas.* Tese (Doutorado) — Universidade de São Paulo, 2007.

[44] SILVA, M. B. R. D. *Uma Avaliação da Abordagem MDA Através de um Estudo de Caso*. Tese (Doutorado) — Universidade Federal Fluminense, 2005.

[45] OMG. Meta object facility (mof). 2002.

[46] GUEDES, G. T. Uml 2-uma abordagem prática-1ª edição. 2008.

[47] LARMAN, C. *Utilizando UML e padrões*. [S.l.]: Bookman Editora, 2002.

[48] LI, X.; LIU, Z.; JIFENG, H. A formal semantics of uml sequence diagram. In: IEEE. *Software Engineering Conference, 2004. Proceedings. 2004 Australian*. [S.l.], 2004. p. 168–177.

[49] CALIC, T.; DASCALU, S.; EGBERT, D. Tools for mda software development: Evaluation criteria and set of desirable features. In: IEEE. *Information Technology: New Generations, 2008. ITNG 2008. Fifth International Conference on*. [S.l.], 2008. p. 44–50.

[50] OMG. *Committed Companies and Their Products*. 2014. Disponível em: <http://www.omg.org/mda/committed-products.htm>. Acesso em: 02.4.2016.

[51] E, R. L. *MODA-TEL (Deliverable 3.4) MDA Foundations and Key Technologie*. 2004.

[52] KITCHENHAM, B. Desmet: A method for evaluating software engineering methods and tools technical report tr96-09. *Department of Computer Science, University of Keele, Staffordshire*, 1996.

[53] NORMAN, D. A. The design of everyday things. Basic Books, 2002.

[54] ROGERS, Y.; SHARP, H.; PREECE, J. Interaction design. Wiley, 2002.

[55] LONDON, K. *An Evaluation of Compuware OptimalJ Professional Edition as an MDA Tool.'*. [S.l.], 2003.

[56] KATEROS, D. A. et al. A methodology for model-driven web application composition. In: *2008 IEEE International Conference on Services Computing*. [S.l.: s.n.], 2008. v. 2, p. 489–492.

[57] ROUBI, S.; ERRAMDANI, M.; MBARKI, S. Modeling and generating graphical user interface for mvc rich internet application using a model driven approach. In: *2016 International Conference on Information Technology for Organizations Development (IT4OD)*. [S.l.: s.n.], 2016. p. 1–6.

[58] DISTANTE, D. et al. Model-driven development of web applications with uwa, mvc and javaserver faces. In: _____. *Web Engineering: 7th International Conference, ICWE 2007 Como, Italy, July 16-20, 2007 Proceedings*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007. p. 457–472. ISBN 978-3-540-73597-7. Disponível em: <http://dx.doi.org/10.1007/978-3-540-73597-7_38>.

[59] KAPITSAKI, G. M. et al. Model-driven development of composite web applications. In: ACM. *Proceedings of the 10th International Conference on Information Integration and Web-based Applications & Services*. [S.l.], 2008. p. 399–402.

[60] KRASNER, G. E.; POPE, S. T. et al. A description of the model-view-controller user interface paradigm in the smalltalk-80 system. *Journal of object oriented programming*, v. 1, n. 3, p. 26–49, 1988.

[61] ISIKDAG, U.; UNDERWOOD, J. Two design patterns for facilitating building information model-based synchronous collaboration. *Automation in Construction*, Elsevier, v. 19, n. 5, p. 544–553, 2010.

[62] PETERSEN, K. et al. Systematic mapping studies in software engineering. In: SN. *12th international conference on evaluation and assessment in software engineering*. [S.l.], 2008. v. 17, n. 1.

[63] VALENZUELA, J.; PAVLICH-MARISCAL, J. A. Hacia un modelo de madurez para apoyar el desarrollo de software dirigido por modelos. In: *2014 9th Computing Colombian Conference (9CCC)*. [S.l.: s.n.], 2014. p. 162–167.

[64] VASCONCELOS, A. M. de et al. Towards cmmi-compliant mdd software processes. In: CITESEER. *ICSEA 2011, The Sixth International Conference on Software Engineering Advances*. [S.l.], 2011.

[65] LI, J. et al. A metamodel for the cmm software process. In: _____. *Parallel and Distributed Processing and Applications: Second International Symposium, ISPA 2004, Hong Kong, China, December 13-15, 2004. Proceedings*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005. p. 446–450. ISBN 978-3-540-30566-8. Disponível em: <http://dx.doi.org/10.1007/978-3-540-30566-8_53>.

[66] RIOS, E. et al. Mdd maturity model: A roadmap for introducing model-driven development. In: SPRINGER. *European Conference on Model Driven Architecture-Foundations and Applications*. [S.l.], 2006. p. 78–89.

[67] VASCONCELOS, A. M. L. de et al. Towards a cmmi-compliant goal-oriented software process through model-driven development. In: SPRINGER. *IFIP Working Conference on The Practice of Enterprise Modeling*. [S.l.], 2011. p. 253–267.

[68] GAILLIARD, G. et al. Transaction level modelling of sca compliant software defined radio waveforms and platforms pim/psm. In: *2007 Design, Automation Test in Europe Conference Exhibition*. [S.l.: s.n.], 2007. p. 1–6. ISSN 1530-1591.

[69] GUTTMAN, M.; PARODI, J. Real-life mda: Solving business problems with model driven architecture. Morgan Kaufmann Publishers Inc., 2006.

[70] ALVES, E. L.; MACHADO, P. D.; RAMALHO, F. Automatic generation of built-in contract test drivers. *Software & Systems Modeling*, Springer, v. 13, n. 3, p. 1141–1165, 2014.

[71] RODRÍGUEZ, A. et al. Secure business process model specification through a uml 2.0 activity diagram profile. *Decision Support Systems*, Elsevier, v. 51, n. 3, p. 446–465, 2011.

[72] RODRÍGUEZ, A. et al. Semi-formal transformation of secure business processes into analysis class and use case models: An mda approach. *Information and Software Technology*, Elsevier, v. 52, n. 9, p. 945–971, 2010.

[73] CASTRO, V. D.; MARCOS, E.; VARA, J. M. Applying cim-to-pim model transformations for the service-oriented development of information systems. *Information and Software Technology*, Elsevier, v. 53, n. 1, p. 87–105, 2011.

[74] RODRÍGUEZ, A.; FERNÁNDEZ-MEDINA, E.; PIATTINI, M. Towards cim to pim transformation: From secure business processes defined in bpmn to use-cases. In: _____. *Business Process Management: 5th International Conference, BPM 2007, Brisbane, Australia, September 24-28, 2007. Proceedings*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007. p. 408–415. ISBN 978-3-540-75183-0. Disponível em: <http://dx.doi.org/10.1007/978-3-540-75183-0_30>.

[75] OSIS, J.; ASNINA, E.; GRAVE, A. Computation independent modeling within the mda. In: *Software-Science, Technology Engineering, 2007. SwSTE 2007. IEEE International Conference on*. [S.l.: s.n.], 2007. p. 22–34.

[76] KHERRAF, S.; LEFEBVRE, .; SURYN, W. Transformation from cim to pim using patterns and archetypes. In: *19th Australian Conference on Software Engineering (aswec 2008)*. [S.l.: s.n.], 2008. p. 338–346. ISSN 1530-0803.

[77] KOCH, N. Transformation techniques in the model-driven development process of uwe. In: *Workshop Proceedings of the Sixth International Conference on Web Engineering*. New York, NY, USA: ACM, 2006. (ICWE '06). ISBN 1-59593-435-9. Disponível em: <http://doi.acm.org/10.1145/1149993.1149997>.

[78] SHARIFI, H. R.; MOHSENZADEH, M. A new method for generating cim using business and requirement models. *World of Computer Science and Information Technology Journal (WCSIT)*, v. 2, n. 1, p. 8–12, 2012.

[79] GUTIÉRREZ, J. J. et al. Visualization of use cases through automatically generated activity diagrams. In: _____. *Model Driven Engineering Languages and Systems: 11th International Conference, MoDELS 2008, Toulouse, France, September 28 - October 3, 2008. Proceedings*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008. p. 83–96. ISBN 978-3-540-87875-9. Disponível em: <http://dx.doi.org/10.1007/978-3-540-87875-9_6>.

[80] WU, J.-H. et al. An extended mda method for user interface modeling and transformation. In: *ECIS*. [S.l.: s.n.], 2007. p. 1632–1642.

[81] RYAN, C.; COLLINS, J.; NEILL, M. O. Grammatical evolution: Evolving programs for an arbitrary language. In: SPRINGER. *European Conference on Genetic Programming*. [S.l.], 1998. p. 83–96.

[82] OMG, O. Unified modeling language. *Superstructure*, 2014.

[83] SOFTEX. *Guia de Implementação – Parte 1: Fundamentação para Implementação do Nível G*. [S.l.: s.n.], 2012.

[84] WAINER, J. et al. Métodos de pesquisa quantitativa e qualitativa para a ciência da computação. *Atualização em informática*, v. 1, p. 221–262, 2007.

[85] SHIRADO, W. H. *Desenvolvimento de Sistema de Automação de Testes Funcionais para Sistemas Embarcados Dirigidos por Modelos*. 2016.

APÊNDICE A – DIAGRAMAS DESENVOLVIDOS PARA O ESTUDO DE CASO

Primeiramente será apresentado a continuação do problema “Internar Paciente no Leito ‘’, que foi iniciado no Capítulo 3. Esta continuação inclui a evolução do PIM, a transformação do PIM para o PSM e a transformação do PSM para o código.

Além desta continuação, neste apêndice os diagramas expostos são correspondentes as funcionalidades “Cadastrar Paciente”, “ Vincular acompanhante ao paciente” e “Gerar Relatório”. Para estas funcionalidades, serão apresentados os cenários de casos de uso e as interfaces de tela, referentes a fase CIM, e os dois diagrama de sequência, referentes a fase PIM. Os diagramas de caso de uso e de classe já foram expostos completos no Capítulo 3 ou neste Apêndice. O código completo gerado a partir do diagrama de classe, também será apresentado neste apêndice.

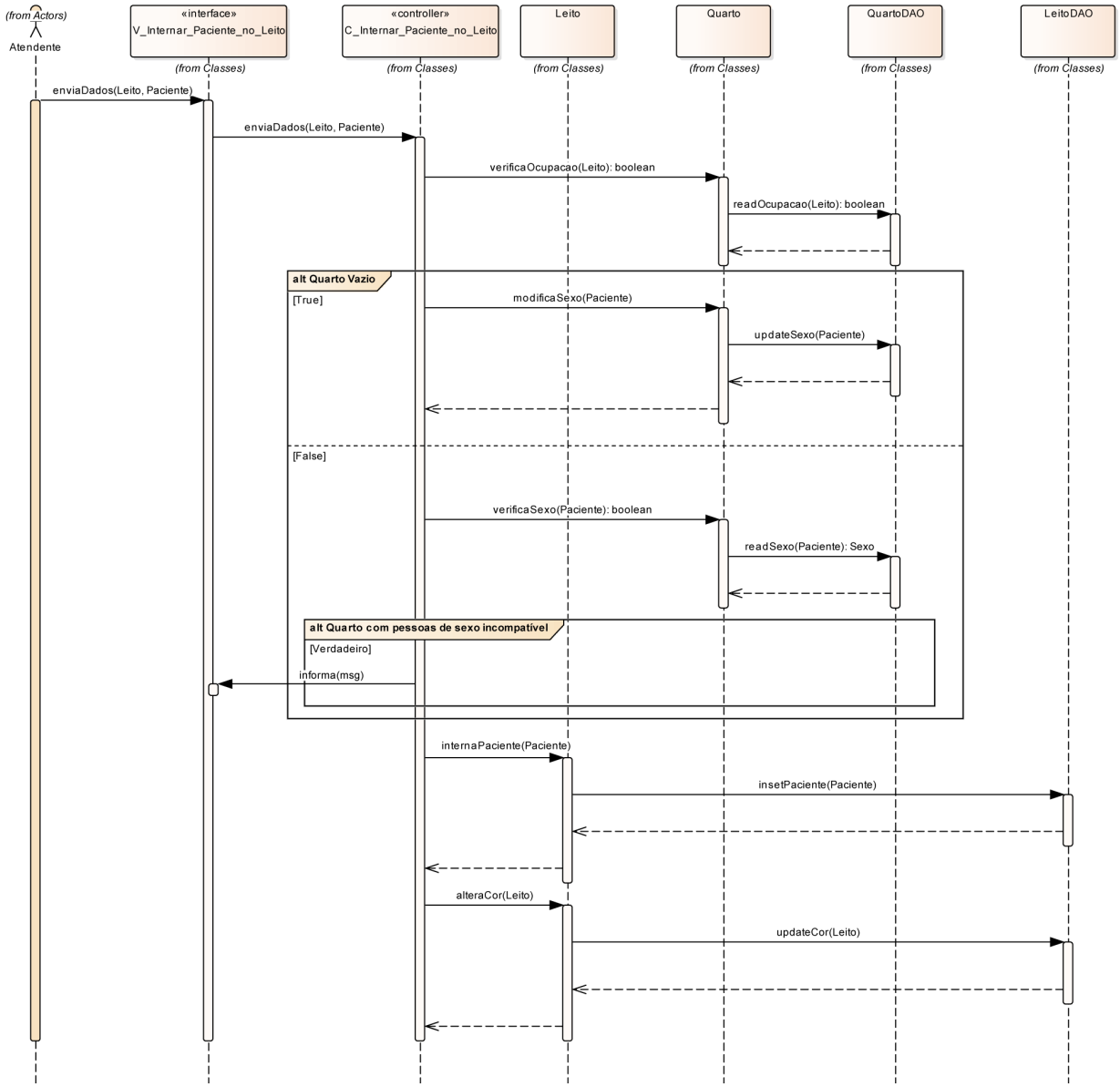
A.1 Continuação do “ Internar Paciente no Leito”

Após os diagramas de PIM derivados dos modelos de CIM, pode-se submeter esses diagramas a uma complementação/evolução. Esta complementação consiste em adicionar elementos que não foram contemplados pelas regras de conversão. Como exemplo, pode-se adicionar elementos do banco de dados.

Neste trabalho será realizada a complementação dos diagramas de PIM pela adição de elementos de bancos de dados, denominadas DAO. Outra complementação realizada será determinar o relacionamento e associação entre as classes do *Model*. Demais complementações podem ser realizadas, no entanto, não serão tratadas neste trabalho.

A Figura 37 apresenta o diagrama de sequência, na fase PIM, após passar pela complementação proposta.

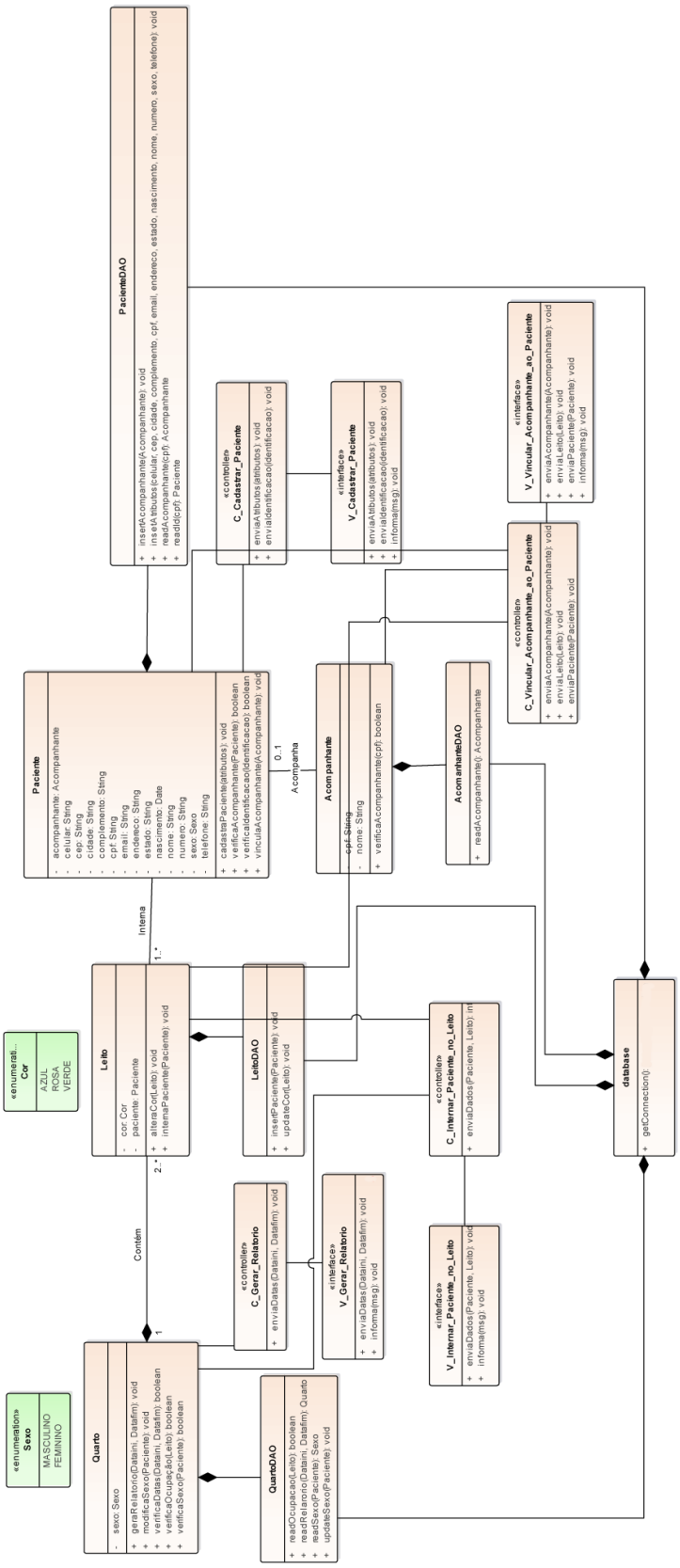
Figura 37 – Diagrama de Sequência depois da Complementação. Fonte: do autor.



A Figura 38 apresenta o digrama de classe na fase PIM, após passar pela complementação proposta.

Após os diagramas do PIM terem passados pela complementação, pode-se realizar a conversão para a fase PSM. Essa conversão é realizada pela ferramenta e ocorre de modo automático para o diagrama de classe. Para os diagramas comportamentais, como é caso do diagrama de sequência, a ferramenta utilizada não possui suporte para conversão. Portanto, para a transformação de PIM para PSM neste trabalho será apresentado o diagrama de classe, que foi realizada a conversão automática pela ferramenta. Para exemplificar como seria um diagrama de sequência transformado de PIM para PSM será apresentado um diagrama elaborado seguindo as premissas da plataforma escolhida .

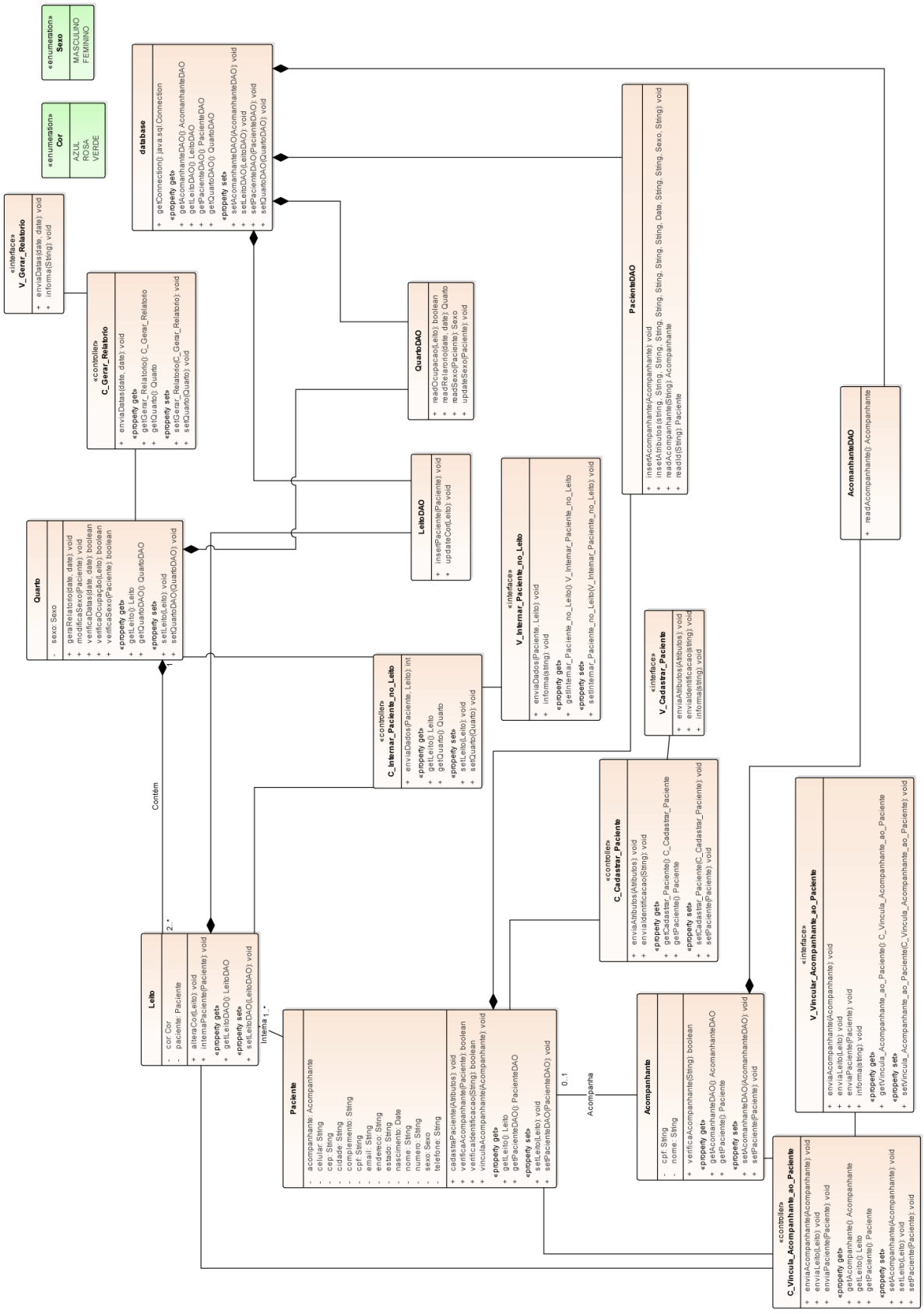
Figura 38 – Diagrama de Classes depois da Complementação. Fonte: do autor.



A Figura 39 apresenta o diagrama de classe após a transformação para o modelo PSM. Como o PSM especifica a plataforma, neste trabalho a linguagem escolhida para a transformação foi a Java.

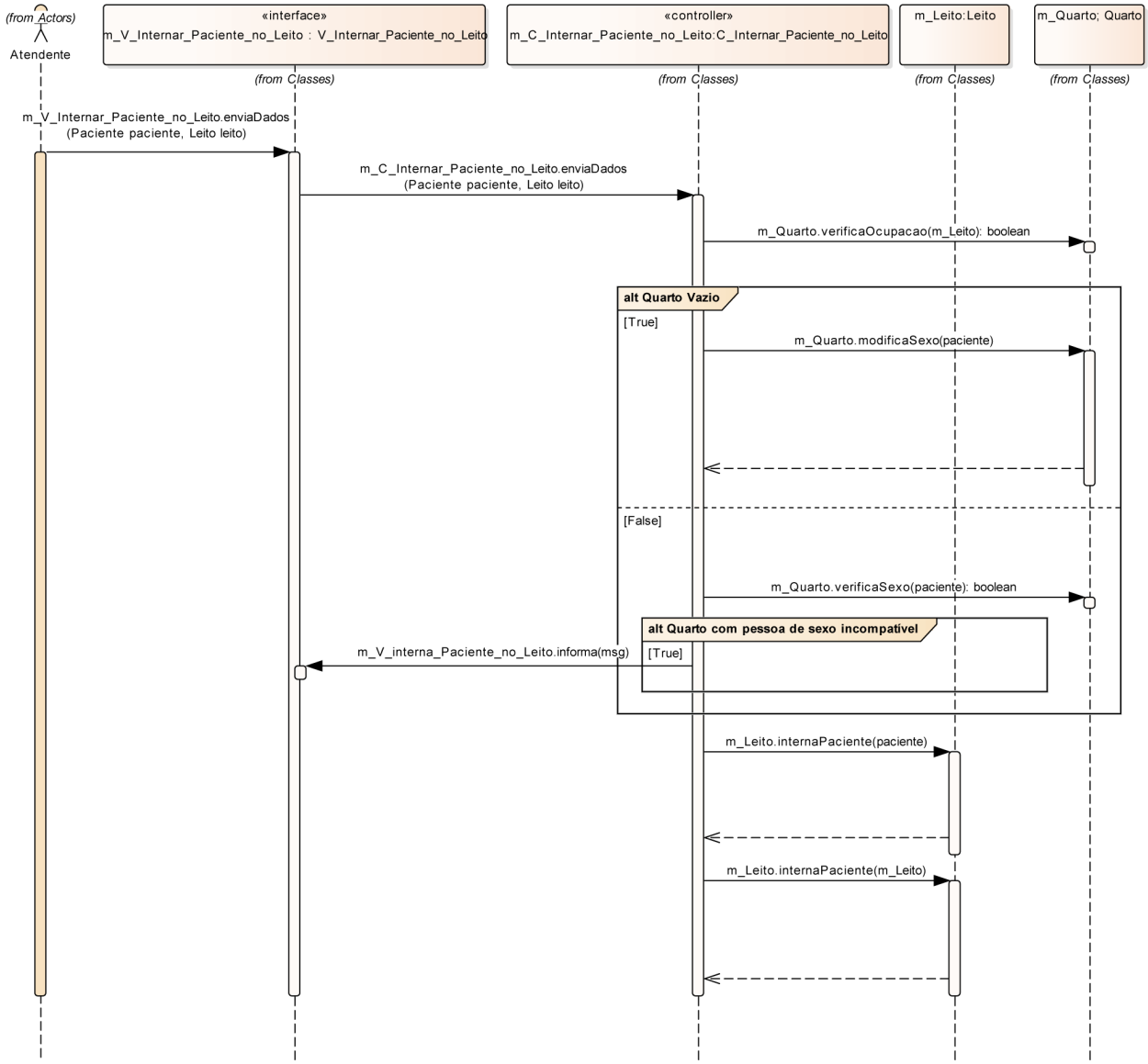
Como pode ser observado na Figura 39, o digrama de classe possui algumas diferenças do diagrama do PIM, representado na Figura 38, o qual foram adicionadas os *gets* e *sets* para as associações entre as classes.

Figura 39 – Modelo PSM do Diagrama de Classe. Fonte: do autor.



A Figura 40 apresenta o diagrama de sequência elaborado para fase PSM. Este diagrama derivou da Figura 28. Assim como no diagrama de classe, a linguagem escolhida para a transformação foi a Java.

Figura 40 – Modelo PSM do Diagrama de Classe. Fonte: do autor.



O último passo é a geração do código. Para o diagrama de classe, o código é gerado de maneira automática do modelo PSM. Já para o diagrama de sequência, essa funcionalidade não é suportada pela ferramenta utilizada. No entanto, a ferramenta disponibiliza a exportação do diagrama para XMI e a partir deste padrão é possível extrair as informações necessárias de forma manual.

O código gerado pelo diagrama de classe do modelo PSM, para o caso de uso " Internar Paciente no Leito" e para a classe Leito, está apresentado nas Figuras 41 e 42 .

A Figura 41 está exposta o código da *View* do " Internar Paciente no Leito".

Figura 41 – Código da Classe View de Internar Paciente no Leito. Fonte: do autor.

```
package Classes;

public class V_Internar_Paciente_no_Leito {

    private C_Internar_Paciente_no_Leito m_C_Internar_Paciente_no_Leito;

    public V_Internar_Paciente_no_Leito(){

    }

    public void finalize() throws Throwable {

    }

    /**
     *
     * @param paciente
     * @param leito
     */
    public void enviaDados(Paciente paciente, Leito leito){

    }

    public C_Internar_Paciente_no_Leito getC_Internar_Paciente_no_Leito(){
        return m_C_Internar_Paciente_no_Leito;
    }

    public V_Internar_Paciente_no_Leito getInternar_Paciente_no_Leito(){
        return m_Internar_Paciente_no_Leito;
    }

    /**
     *
     * @param msg
     */
    public void informa(string msg){

    }

    /**
     *
     * @param newVal
     */
    public void setC_Internar_Paciente_no_Leito(C_Internar_Paciente_no_Leito newVal){
        m_C_Internar_Paciente_no_Leito = newVal;
    }

    /**
     *
     * @param newVal
     */
    public void setInternar_Paciente_no_Leito(V_Internar_Paciente_no_Leito newVal){
        m_Internar_Paciente_no_Leito = newVal;
    }

}
```

A Figura 42 possui dois códigos: em (a) está o código do *Controller* do caso de uso e em (b) está o código da classe Leito.

Figura 42 – Códigos das classes: (a) Controller de Internar Paciente no Leito (b) Leito. Fonte: do autor.

```java
package Classes;

public class C_Internar_Paciente_no_Leito {

    private Leito m_Leito;
    private Quarto m_Quarto;

    public C_Internar_Paciente_no_Leito(){

    }

    public void finalize() throws Throwable {

    }

    /**
     *
     * @param paciente
     * @param leito
     */
    public int enviaDados(Paciente paciente, Leito leito){
        return 0;
    }


    public Leito getLeito(){
        return m_Leito;
    }

    public Quarto getQuarto(){
        return m_Quarto;
    }


    /**
     *
     * @param newVal
     */
    public void setLeito(Leito newVal){
        m_Leito = newVal;
    }

    /**
     *
     * @param newVal
     */


    public void setQuarto(Quarto newVal){
        m_Quarto = newVal;
    }

    /**
     *
     * @param newVal
     */

}
```

(a)

```java
package Classes;

public class Leito {

    private Cor cor;
    private Paciente paciente;
    private LeitoDAO m_LeitoDAO;

    public Leito(){

    }

    public void finalize() throws Throwable {

    }

    /**
     *
     * @param leito
     */
    public void alteraCor(Leito leito){

    }

    public LeitoDAO getLeitoDAO(){
        return m_LeitoDAO;
    }


    /**
     *
     * @param Paciente
     */
    public void internaPaciente(Paciente Paciente){

    }

    /**
     *
     * @param newVal
     */
    public void setLeitoDAO(LeitoDAO newVal){
        m_LeitoDAO = newVal;
    }


}
```

(b)

Para extrair o código do diagrama de sequência é necessário o padrão em XMI do mesmo. Com o XMI é possível realizar o mapeamento completo dos elementos do diagrama de sequência para o código. Para exemplificar este mapeamento será realizada uma transformação utilizando o XMI do diagrama de sequência da Figura 37. Para este mapeamento é necessário identificar certos elementos presentes no padrão XMI, que são elementos e atributos de relevância para este mapeamento. São esses [85]:

- **<UML: ClassifierRole>** : este é um elemento que simboliza uma *lifeline* no

diagramas, sendo o atributo *"name"* utilizado para referenciar qual é esta *lifeline*.

- **<UML:Message** name = " nome do método"/> : O elemento *Message* é utilizado para assinalar a presença de uma mensagem entre dois elementos. O atributo *"name"*, neste caso, contém o nome do método invocado. Para especificar os elementos fonte e alvo são utilizados os atributo *" source"* e *" target"*, respectivamente.

- **<UML:TaggedValue>**: este elemento é empregado para armazenar valores rotulados. Esses valores servem como meios de descrição das características de um *ClassifierRole*, de uma mensagem ou outros elementos.

- **<UML: CombinedFragment>**: este elemento apresenta o fragmento combinado. O atributo *"name"* recebe a especificação dada para aquele fragmento. O elemento *"interactionOperator"* indica qual tipo de fragmento combinado está sendo utilizado, se é alternativo, *loop* e etc .

A Figura 43 apresenta um fragmento do XMI do diagrama de sequência da Figura 40.

Figura 43 – XMI referente ao Diagrama de Sequência. Fonte: do autor.

```
<UML:Message xmi.id= ...      name="m_Quarto.verificaOcupacao"
   <UML:TaggedValue value="paramvalues=m_Leito;"
  <UML:TaggedValue value="m_C_Internar_Paciente_no_Leito:C_Internar_Paciente_no_Leito"
      tag="ea_sourceName"/>
  <UML:TaggedValue value="m_Quarto:Quarto" tag="ea_targetName"/>
```

Pode-se notar que este fragmento fornece a mensagem enviada, bem como o parâmetro enviado com a mensagem e os elementos fonte e alvo desta mensagem .

Como citado anteriormente, pode-se extrair o código do XMI e a Figura 43 apresenta elementos que possibilitam esta extração. Portanto, foi elaborado o código para todo o diagrama de sequência da Figura 40, porém na Figura 44 será apresentado o código da classe C_Internar_Paciente_no_Leito, visto que esta classe possui uma quantidade maior de código gerado a partir do diagrama de sequência.

Figura 44 – Código do Modelo PSM do Diagrama de Sequência. Fonte do autor.

```
public class C_Internar_Paciente_no_Leito {
 private Leito m_Leito;
 private Quarto m_Quarto;
 private V_Internar_Paciente_no_Leito m_V_Internar_Paciente_no_Leito;

 public C_Internar_Paciente_no_Leito(){
 }
 public void finalize() throws Throwable {
 }
 /**
  *
  * @param paciente
  * @param leito
  */
 public int enviaDados(Paciente paciente, Leito leito){
   boolean Quarto_Vazio = m_Quarto.verifica_Ocupacao(m_Leito);

       if (Quarto_Vazio) {
              m_Quarto.modificaSexo(paciente);
       }
       else {
           boolean Quarto_com_pessoas_de_sexo_incompativel = m_Quarto.verificaSexo(paciente);
         if  (Quarto_com_pessoas_de_sexo_incompativel){
             m_V_interna_Paciente_no_Leito.informa("msg");
         }
       }
       m_Leito.internaPaciente(paciente);
       m_Leito.alteraCor(m_Leito);


 return 0;
 }

 public Leito getLeito(){
  return m_Leito;
 }

 public Quarto getQuarto(){
  return m_Quarto;
 }

 /**
  *
  * @param newVal
  */
 public void setLeito(Leito newVal){
  m_Leito = newVal;
 }

 /**
  *
  * @param newVal
  */
 public void setQuarto(Quarto newVal){
  m_Quarto = newVal;
 }

 }
}
```

Com o código apresentado na Figura 44, pode-se concluir que o código gerado pelo diagrama de sequência é a lógica do sistema, porém ele ainda não é completo, portanto, existe a necessidade da complementação deste código de forma manual.

A Figura 44 é o código completo da classe C_Internar_Paciente_no_Leito, portanto já inclui os códigos gerados pelo o diagrama de classe e pelo diagrama de sequência.

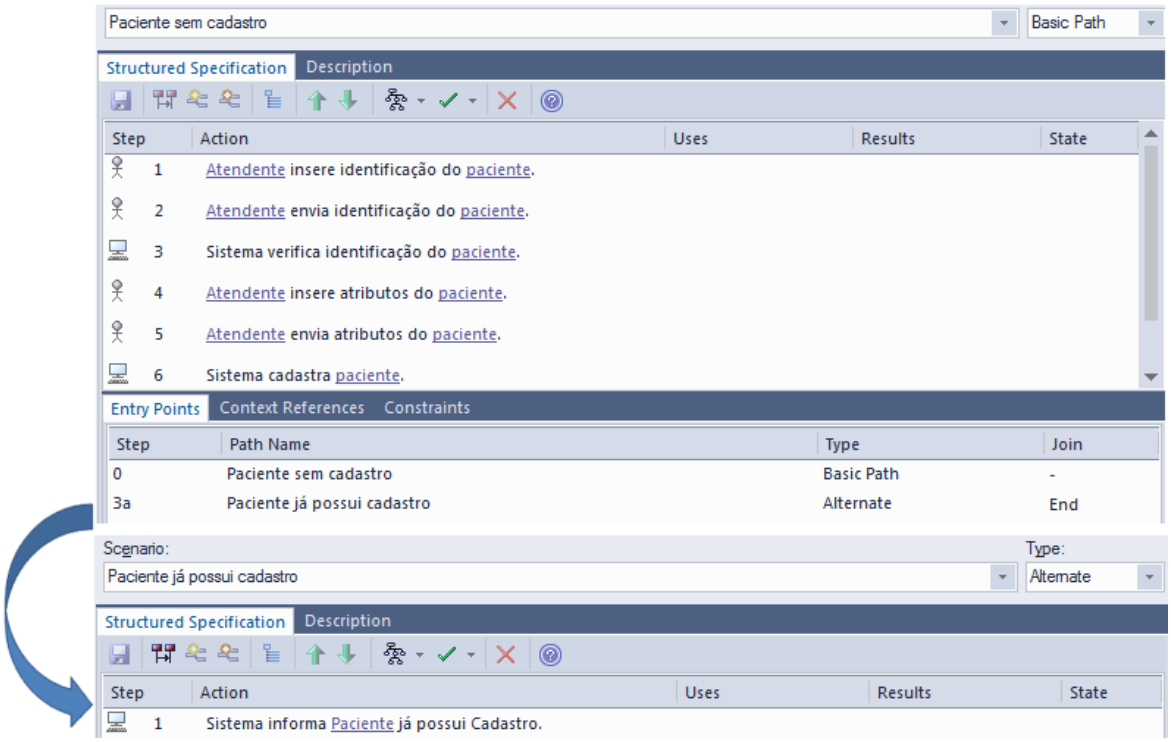
A.2 Demais Funcionalidades do Sistema da Clínica

Nesta seção serão apresentados os modelos das funcionalidades: “Cadastrar Paciente”, “ Vincular acompanhante ao paciente” e “Gerar Relatório”. Para estas funcionalidades serão apresentados os cenários de caso de uso, interface de tela, diagramas de sequência e os códigos gerados à partir do diagrama de classe da Figura 39 . Primeiro serão apresentados todos os modelos de CIM, depois os diagramas de sequência de PIM e por último os códigos.

Vale ressaltar que os diagramas de CIM seguem as premissas estabelecidas no Capítulo 3, portanto todas as orações elaboradas nos cenários de casos de uso seguem o BNF estabelecido.

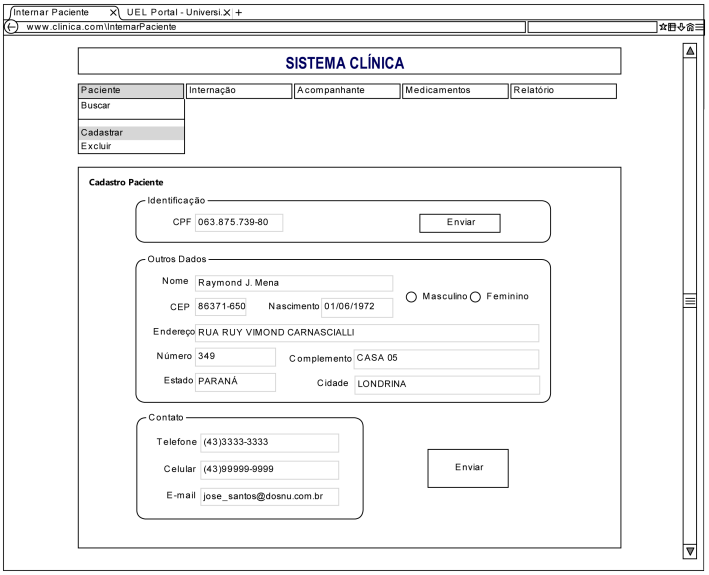
A Figura 45 apresenta o cenário de caso de uso para “ Cadastrar Paciente ”.

Figura 45 – Cenário de Caso de Uso de Cadastrar Paciente. Fonte: do autor



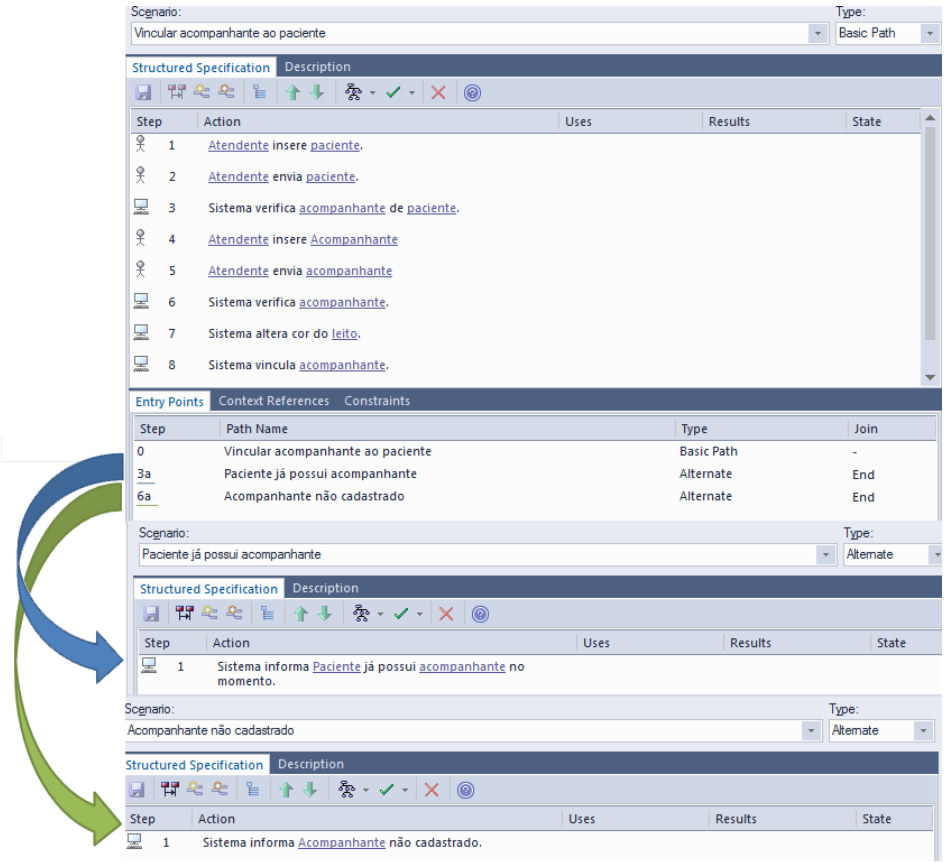
A interface de tela correspondente de “ Cadastrar Paciente ” está presente na Figura 46.

Figura 46 – Interface de Tela do Cadastrar Paciente. Fonte: do autor.



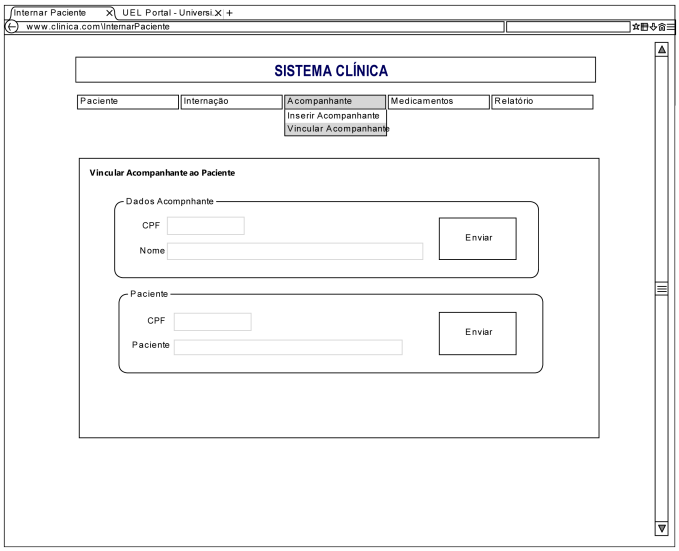
A Figura 47 apresenta o cenário de caso de uso para “ Vincular acompanhante ao paciente ”.

Figura 47 – Cenário de Caso de Uso de Vincular acompanhente ao Paciente. Fonte: do autor.



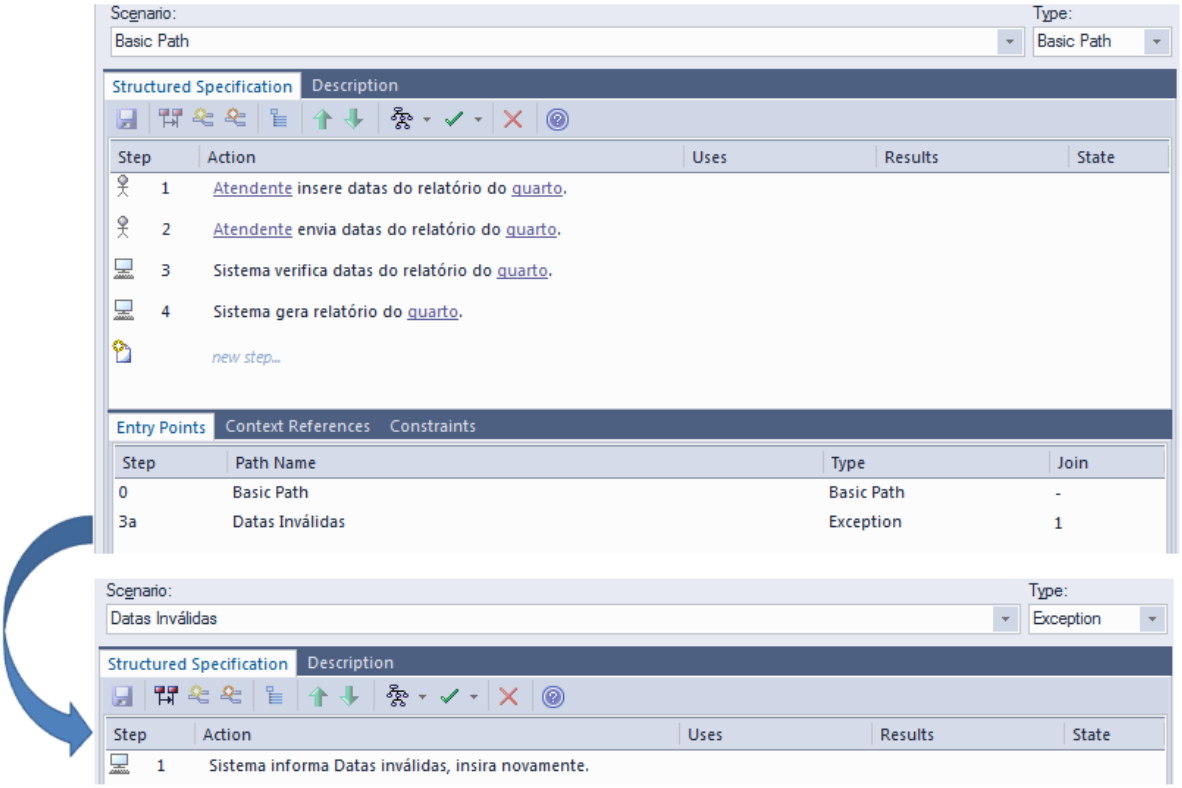
A interface de tela que corresponde do cenário “ Vincular acompanhante ao paciente ” está presenta na Figura 48.

Figura 48 – Interface de tela de Vincular Acompanhante ao Paciente. Fonte: do autor.



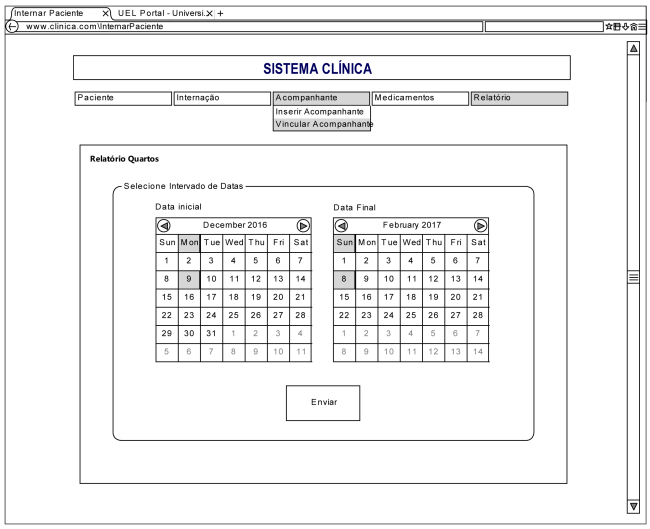
A Figura 49 apresenta o cenário de caso de uso para “ Gerar relatório”.

Figura 49 – Cenário de Caso de Uso para Gerar Relatório. Fonte: do autor.



A interface de tela correspondente a “Gerar relatório” está apresentada na Figura 50.

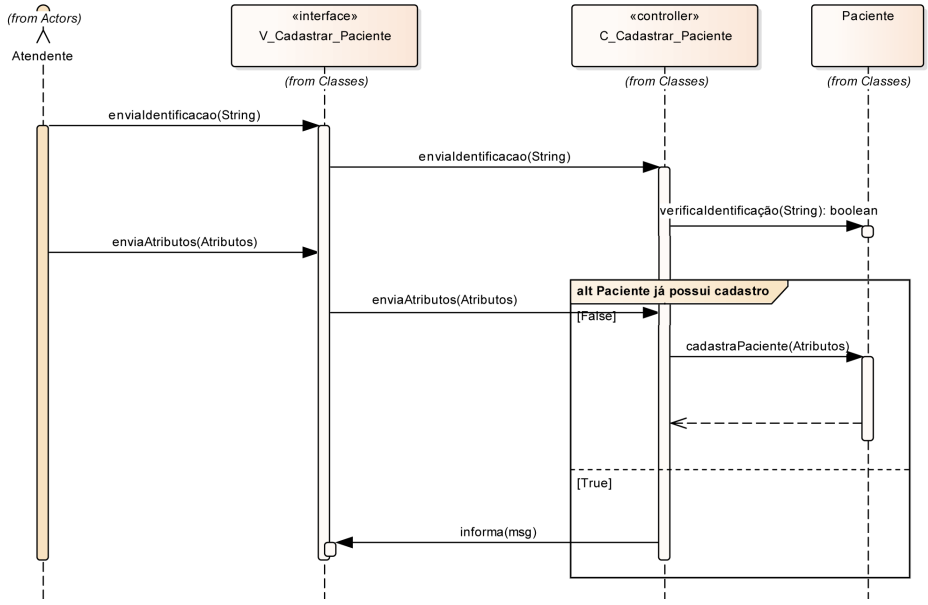
Figura 50 – Interface de Tela de Gerar Relatório. Fonte: do autor.



Esses diagramas apresentados são referentes a fase CIM. Após esses diagramas deve-se realizar as conversões para os diagramas de PIM. Essas conversões são realizadas através das regras apresentadas no Capítulo 3. O diagrama de PIM apresentado nesta seção será o diagrama de sequência, visto que o diagrama de classe já foi apresentado anteriormente. Portanto, serão apresentado dois "grupos" de diagramas de sequência. O primeiro grupo são os diagramas extraídos a partir das regras de conversão apresentado no Capítulo 3, logo esses diagramas foram transformados a partir dos modelos de CIM utilizando as regras de conversão. Já o segundo grupo serão os diagramas de sequência que passaram por uma complementação.

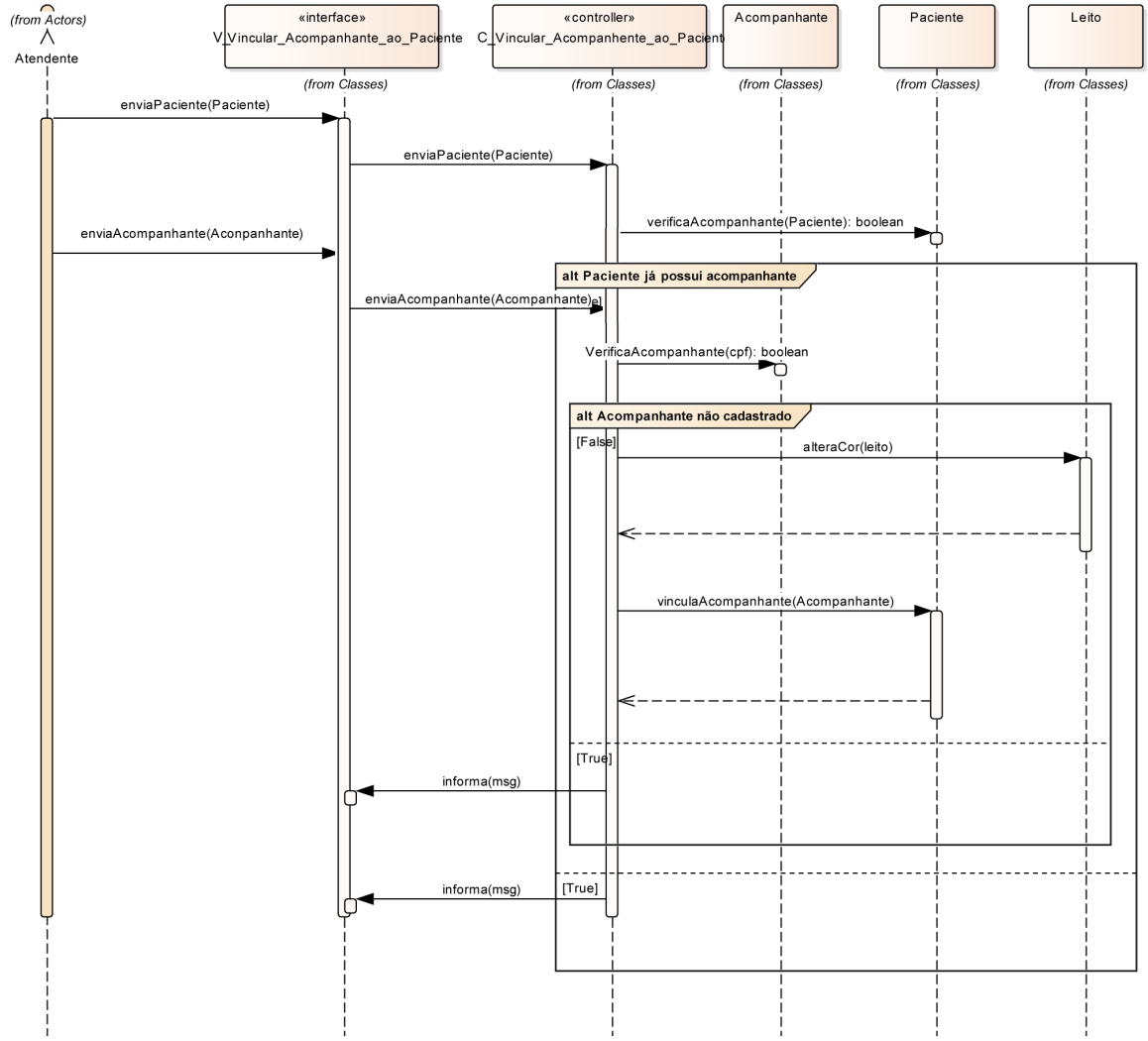
A Figura 51 apresenta o diagrama de sequência para "Cadastrar Paciente" extraído do cenário de caso de uso da Figura 45.

Figura 51 – Diagrama de Sequência para Cadastrar Paciente. Fonte: do autor.



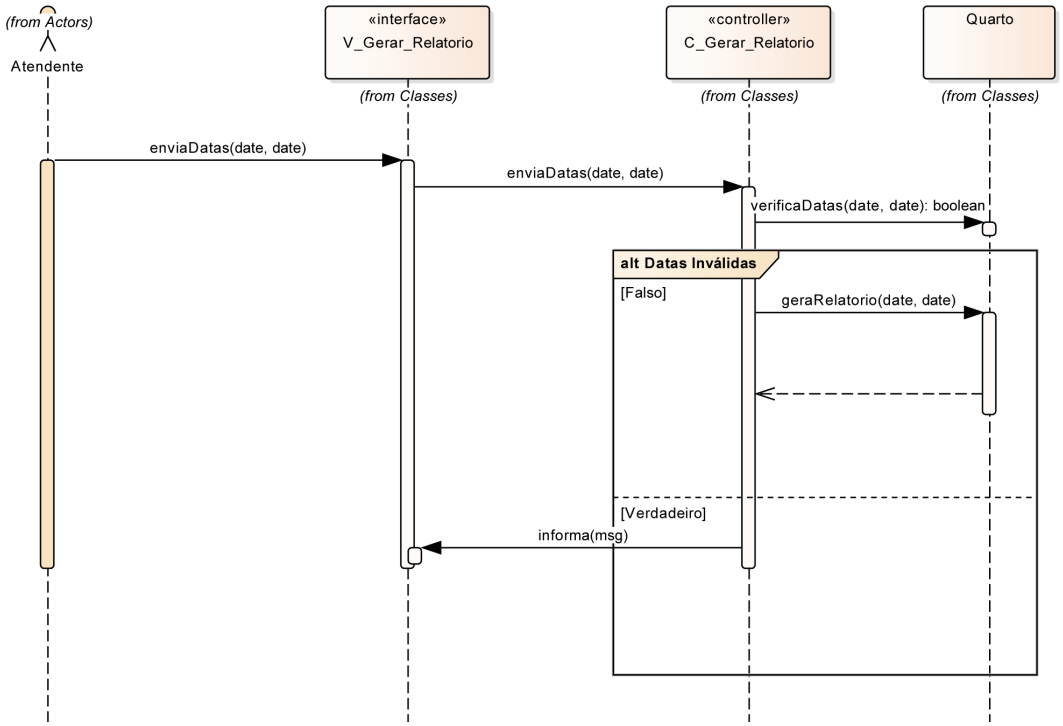
A Figura 52 apresenta o diagrama de sequência para “Vincular Acompanhante ao Paciente”extraído do cenário de caso de uso da Figura 47.

Figura 52 – Diagrama de Sequência para o Vincular Acompanhante ao Paciente. Fonte: do autor.



A Figura 53 apresenta o diagrama de sequência para “Gerar Relatório” extraído do cenário de caso de uso da Figura 49.

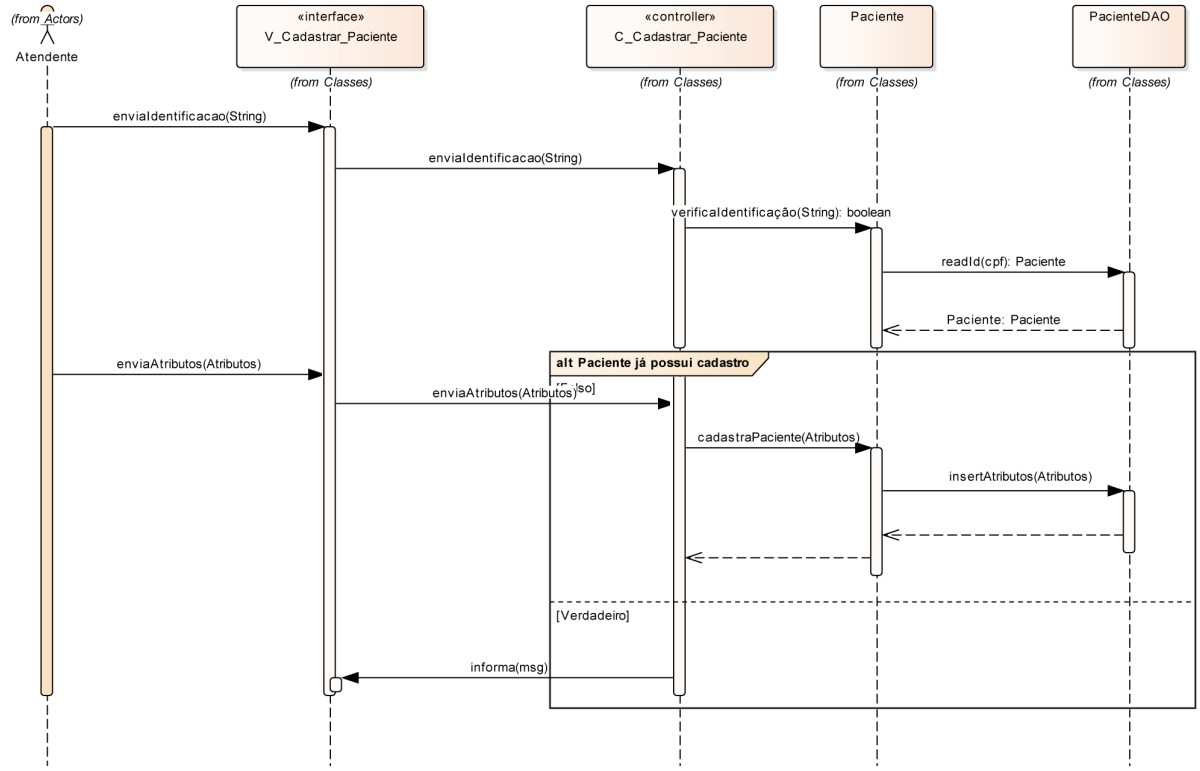
Figura 53 – Diagrama de Sequência para o Gerar Relatório. Fonte: do autor.



Os diagramas de sequência apresentados nas Figuras 51, 52 e 53 foram derivados dos diagramas de CIM. Os próximos diagramas apresentados serão uma complementação dos modelos já apresentados. Essa complementação foi realizada adicionando elementos referentes a banco de dados, denominados DAO.

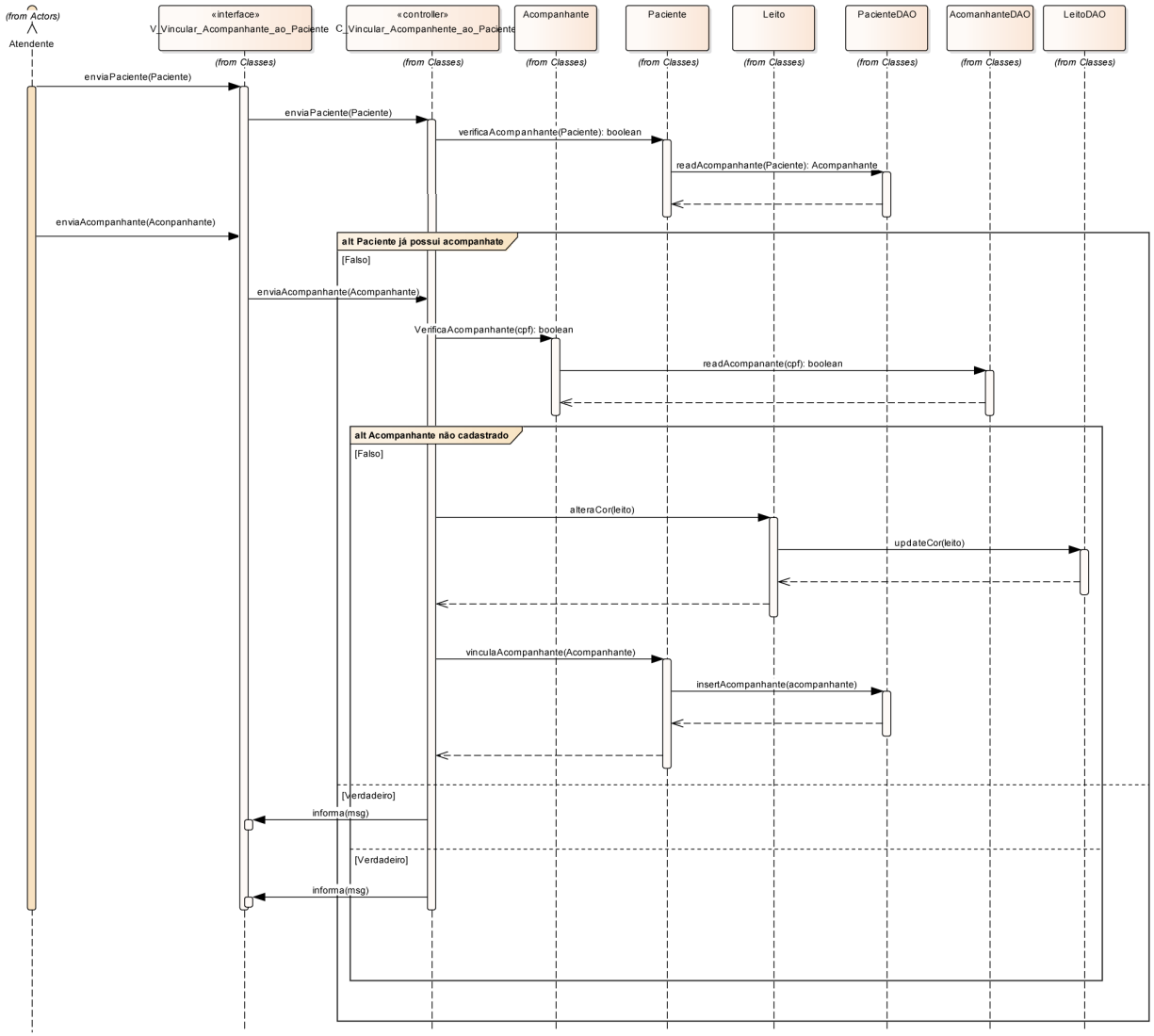
A Figura 54 apresenta o diagrama de sequência para “ Cadastrar Paciente”, na fase PIM após passar pela complementação.

Figura 54 – Diagrama de Sequência para Cadastrar Paciente após a Complementação. Fonte: do autor.



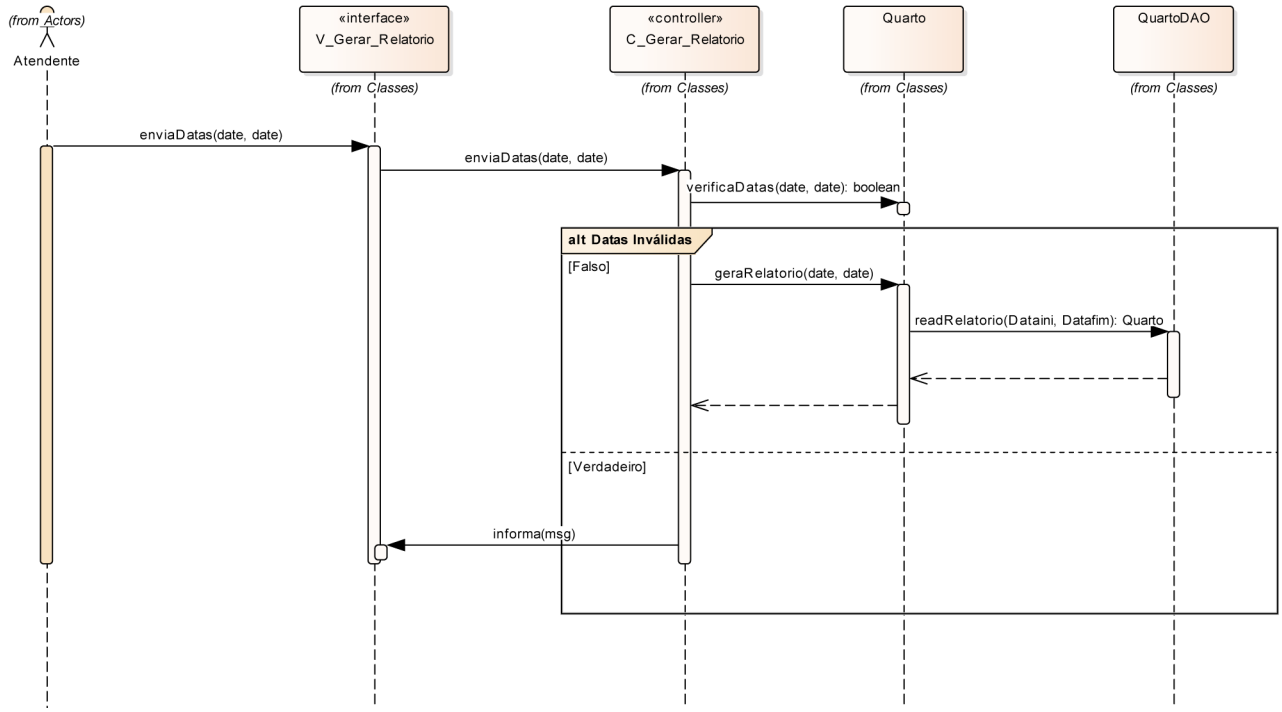
A Figura 55 apresenta o diagrama de sequência para “ Vincular Acompanhante ao Paciente”, na fase PIM após passar pela complementação.

Figura 55 – Diagrama de Sequência para Vincular Acompanhante ao Paciente após a Complementação. Fonte: do autor.



A Figura 56 apresenta o diagrama de sequência para “ Gerar Relatório”, na fase PIM após passar pela complementação.

Figura 56 – Diagrama de Sequência para Gerar Relatório após a Complementação. Fonte: do autor.



Como já discorrido anteriormente, não foi possível gerar automaticamente o código fonte para os diagramas de sequência a partir da ferramenta utilizada. Portanto, nesta seção só serão apresentados os códigos fontes gerados a partir do diagrama de classe PSM da Figura 39.

Na Figura 57 estão presentes os códigos fontes das Classes *View* e *Controller* de Cadastrar Paciente.

Figura 57 – Código Fonte de Cadastrar Paciente: (a) Classe *Controller* e (b) Classe *View*. Fonte: do autor.

```
package Classes;

public class C_Cadastrar_Paciente {

    private Paciente m_Paciente;
    private V_Cadastrar_Paciente m_V_Cadastrar_Paciente;

    public C_Cadastrar_Paciente(){

    }

    public void finalize() throws Throwable {

    }

    public void enviaAtributos(Atributos atributos){

    }

    public void enviaIdentificacao(String identificacao){

    }

    public C_Cadastrar_Paciente getCadastrar_Paciente(){
        return m_Cadastrar_Paciente;
    }

    public Paciente getPaciente(){
        return m_Paciente;
    }


    public V_Cadastrar_Paciente getV_Cadastrar_Paciente(){
        return m_V_Cadastrar_Paciente;
    }


    public void setCadastrar_Paciente(C_Cadastrar_Paciente newVal){
        m_Cadastrar_Paciente = newVal;
    }

    public void setPaciente(Paciente newVal){
        m_Paciente = newVal;
    }



    public void setV_Cadastrar_Paciente(V_Cadastrar_Paciente newVal){
        m_V_Cadastrar_Paciente = newVal;
    }

}
```

(a)

```
package Classes;

public class V_Cadastrar_Paciente {

    public V_Cadastrar_Paciente(){

    }

    public void finalize() throws Throwable {

    }


    public void enviaAtributos(Atributos atributos){

    }


    public void enviaIdentificacao(string identificacao){

    }


    public void informa(string msg){

    }
}
```

(b)

Na Figura 58 estão apresentados dois códigos fontes. O (a) é o da Classe Paciente e o (b) é o da classe Quarto.

Figura 58 – Código fonte das Classes: (a) Paciente e (b) Quarto. Fonte: do autor.

```java
package Classes;

public class Paciente {

    private Acompanhante acompanhante;
    private String celular;
    private String cep;
    private String cidade;
    private String complemento;
    private String cpf;
    private String email;
    private String endereco;
    private String estado;
    private Date nascimento;
    private String nome;
    private String numero;
    private Sexo sexo;
    private String telefone;
    private PacienteDAO m_PacienteDAO;
    private Leito m_Leito;

    public Paciente(){

    }

    public void finalize() throws Throwable {

    }


    public void cadastraPaciente(Atributos atributos){

    }

    public Leito getLeito(){
        return m_Leito;
    }


    public PacienteDAO getPacienteDAO(){
        return m_PacienteDAO;
    }



    public void setLeito(Leito newVal){
        m_Leito = newVal;
    }



    public void setPacienteDAO(PacienteDAO newVal){
        m_PacienteDAO = newVal;
    }



    public boolean verificaAcompanhante(Paciente paciente){
        return false;
    }


    public boolean verificaIdentificacao(String Identificacao)
        return false;
    }


    public void vinculaAcompanhante(Acompanhante acompanhante)

    }

}
```

(a)

```java
package Classes;

public class Quarto {

    private Sexo sexo;
    private Leito m_Leito;
    private QuartoDAO m_QuartoDAO;

    public Quarto(){

    }

    public void finalize() throws Throwable {

    }


    public void geraRelatorio(date Dataini, date Datafim){

    }

    public Leito getLeito(){
        return m_Leito;
    }


    public QuartoDAO getQuartoDAO(){
        return m_QuartoDAO;
    }



    public void modificaSexo(Paciente paciente){

    }


    public void setLeito(Leito newVal){
        m_Leito = newVal;
    }



    public void setQuartoDAO(QuartoDAO newVal){
        m_QuartoDAO = newVal;
    }



    public boolean verificaDatas(date Dataini, date Datafim){
        return false;
    }


    public boolean verificaOcupação(Leito leito){
        return false;
    }


    public boolean verificaSexo(Paciente paciente){
        return false;
    }

}
```

(b)

A Figura 59 apresenta o código da classe View da Classe Vincular Acompanhante ao Paciente.

Figura 59 – Classe *View* de Vincular Acompanhante ao Paciente. Fonte: do autor.

```
package Classes;

public class V_Vincular_Acompanhante_ao_Paciente {

    private C_Vincula_Acompanhante_ao_Paciente m_C_Vincula_Acompanhante_ao_Paciente;

    public V_Vincular_Acompanhante_ao_Paciente(){

    }

    public void finalize() throws Throwable {

    }


    public void enviaAcompanhante(Acompanhante acompanhante){

    }

    public void enviaLeito(Leito leito){

    }


    public void enviaPaciente(Paciente paciente){

    }

    public C_Vincula_Acompanhante_ao_Paciente getC_Vincula_Acompanhante_ao_Paciente(){
        return m_C_Vincula_Acompanhante_ao_Paciente;
    }

    public C_Vincula_Acompanhante_ao_Paciente getVincula_Acompanhante_ao_Paciente(){
        return m_Vincula_Acompanhante_ao_Paciente;
    }

    public void informa(string msg){

    }


    public void setC_Vincula_Acompanhante_ao_Paciente(C_Vincula_Acompanhante_ao_Paciente
newVal){
        m_C_Vincula_Acompanhante_ao_Paciente = newVal;
    }


    public void setVincula_Acompanhante_ao_Paciente(C_Vincula_Acompanhante_ao_Paciente
newVal){
        m_Vincula_Acompanhante_ao_Paciente = newVal;
    }

}
```

A Figura 60 expõe a classe *Controller* de Vincular Acompanhante ao Paciente

Figura 60 – Classe *Controller* de Vincular Acompanhante ao Paciente. Fonte: do autor.

```
package Classes;

public class C_Vincula_Acompanhante_ao_Paciente {

    private Acompanhante m_Acompanhante;
    private Paciente m_Paciente;
    private Leito m_Leito;

    public C_Vincula_Acompanhante_ao_Paciente(){

    }

    public void finalize() throws Throwable {

    }

    public void enviaAcompanhante(Acompanhante acompanhante){

    }


    public void enviaLeito(Leito leito){

    }

    public void enviaPaciente(Paciente paciente){

    }

    public Acompanhante getAcompanhante(){
        return m_Acompanhante;
    }


    public Leito getLeito(){
        return m_Leito;
    }


    public Paciente getPaciente(){
        return m_Paciente;
    }


    public void setAcompanhante(Acompanhante newVal){
        m_Acompanhante = newVal;
    }



    public void setLeito(Leito newVal){
        m_Leito = newVal;
    }


    public void setPaciente(Paciente newVal){
        m_Paciente = newVal;
    }

}
```

A Figura 61 apresenta a Classe Acompanhante.

Figura 61 – Código da Classe Acompanhante. Fonte: do autor.

```
package Classes;


/**
 * @author Vanessa
 * @version 1.0
 * @created 23-fev-2017 16:57:27
 */
public class Acompanhante {

    private String cpf;
    private String nome;
    private AcomanhanteDAO m_AcomanhanteDAO;
    private Paciente m_Paciente;

    public Acompanhante(){

    }

    public void finalize() throws Throwable {

    }

    public AcomanhanteDAO getAcomanhanteDAO(){
        return m_AcomanhanteDAO;
    }


    public Paciente getPaciente(){
        return m_Paciente;
    }


    /**
     *
     * @param newVal
     */
    public void setAcomanhanteDAO(AcomanhanteDAO newVal){
        m_AcomanhanteDAO = newVal;
    }




    /**
     *
     * @param newVal
     */
    public void setPaciente(Paciente newVal){
        m_Paciente = newVal;
    }



    /**
     *
     * @param cpf
     */
    public boolean verificaAcompanhante(String cpf){
        return false;
    }

}
```

Figura 62 – Código Fonte de Gerar Relatório: (a) Classe *Controller* e (b) Classe *View*. Fonte: do autor.

```
package Classes;

public class C_Gerar_Relatorio {

    private V_Gerar_Relatorio m_V_Gerar_Relatorio;
    private Quarto m_Quarto;

    public C_Gerar_Relatorio(){

    }

    public void finalize() throws Throwable {

    }

    public void enviaDatas(date Dataini, date Datafim){

    }

    public C_Gerar_Relatorio getGerar_Relatorio(){
        return m_Gerar_Relatorio;
    }


    public Quarto getQuarto(){
        return m_Quarto;
    }

    public V_Gerar_Relatorio getV_Gerar_Relatorio(){
        return m_V_Gerar_Relatorio;
    }

    public void setGerar_Relatorio(C_Gerar_Relatorio newVal){
        m_Gerar_Relatorio = newVal;
    }



    public void setQuarto(Quarto newVal){
        m_Quarto = newVal;
    }


    public void setV_Gerar_Relatorio(V_Gerar_Relatorio newVal){
        m_V_Gerar_Relatorio = newVal;
    }

}
```

(a)

```
package Classes;

public class V_Gerar_Relatorio {

    public V_Gerar_Relatorio(){

    }

    public void finalize() throws Throwable {

    }


    public void enviaDatas(date Dataini, date Datafim){

    }


    public void informa(String msg){

    }

}
```

(b)

Na figura 62 estão apresentados dois códigos fontes. O (a) é o da Classe *Controller* de Gerar Relatório e o (b) é o da classe *View* de Gerar Relatório.

Na Figura 63 está apresentado o código fonte para a classes PacienteDAO.

Figura 63 – Código Fonte da Classe PacienteDAO. Fonte: do autor.

```
package Classes;

public class PacienteDAO {

    public PacienteDAO(){

    }

    public void finalize() throws Throwable {

    }


    public void insertAcompanhante(Acompanhante acompanhante){

    }

    /**
     *
     * @param celular
     * @param cep
     * @param cidade
     * @param complemento
     * @param cpf
     * @param email
     * @param endereco
     * @param estado
     * @param nascimento
     * @param nome
     * @param numero
     * @param sexo
     * @param telefone
     */
    public void insetAtributos(string celular, String cep, String cidade, String
complemento, String cpf, String email, String endereco, String estado, Date nascimento,
String nome, String numero, Sexo sexo, String telefone){

    }

    /**
     *
     * @param cpf
     */
    public Acompanhante readAcompanhante(String cpf){
        return null;
    }

    /**
     *
     * @param cpf
     */
    public Paciente readId(String cpf){
        return null;
    }

}
```

Na Figura 64 está apresentado o código fonte para a classes QuartoDAO.

Figura 64 – Código Fonte da Classes QuartoDAO. Fonte: do autor.

```
package Classes;

public class QuartoDAO {

    public QuartoDAO(){

    }

    public void finalize() throws Throwable {

    }


    public boolean readOcupacao(Leito leito){
        return false;
    }


    public Quarto readRelarorio(date Dataini, date Datafim){
        return null;
    }


    public Sexo readSexo(Paciente paciente){
        return null;
    }


    public void updateSexo(Paciente paciente){

    }

}
```

Na Figura 65 estão expostos os códigos fontes para as classes: (a) LeitoDAO e (b) AcompanhanteDAO.

Figura 65 – Código Fonte das Classes: (a) LeitoDAO e (b) AcompanhanteDAO. Fonte: do autor.

```java
package Classes;

public class LeitoDAO {

    public LeitoDAO(){

    }

    public void finalize() throws Throwable {

    }

    /**
     *
     * @param paciente
     */
    public void insertPaciente(Paciente paciente){

    }

    /**
     *
     * @param leito
     */
    public void updateCor(Leito leito){

    }

}
```

(a)

```java
package Classes;

public class AcomanhanteDAO {

    public AcomanhanteDAO(){

    }

    public void finalize() throws Throwable {

    }

    public Acompanhante readAcompanhante(){
        return null;
    }

}
```

(b)

A Figura 66 apresenta o código da classe database.

Figura 66 – Código Fonte da classe database. Fonte: do autor.

```
package Classes;


public class database {

    private AcomanhanteDAO m_AcomanhanteDAO;
    private PacienteDAO m_PacienteDAO;
    private QuartoDAO m_QuartoDAO;
    private LeitoDAO m_LeitoDAO;

    public database(){

    }

    public void finalize() throws Throwable {

    }



    public AcomanhanteDAO getAcomanhanteDAO(){
        return m_AcomanhanteDAO;
    }

    public java.sql.Connection getConnection(){
        return null;
    }



    public LeitoDAO getLeitoDAO(){
        return m_LeitoDAO;
    }



    public PacienteDAO getPacienteDAO(){
        return m_PacienteDAO;
    }



    public QuartoDAO getQuartoDAO(){
        return m_QuartoDAO;
    }



    public void setAcomanhanteDAO(AcomanhanteDAO newVal){
        m_AcomanhanteDAO = newVal;
    }



    public void setLeitoDAO(LeitoDAO newVal){
        m_LeitoDAO = newVal;
    }



    public void setPacienteDAO(PacienteDAO newVal){
        m_PacienteDAO = newVal;
    }



    public void setQuartoDAO(QuartoDAO newVal){
        m_QuartoDAO = newVal;
    }

}
```

Por fim, a Figura 67 apresenta os códigos das classes *Enumeration* cor e sexo respectivamente.

Figura 67 – Código Fonte das Classes: (a) Cor e (b) Sexo. Fonte: do autor.

```java
package Classes;

public enum Cor {
    AZUL,
    ROSA,
    VERDE
}
```

(a)

```java
package Classes;

public enum Sexo {
    MASCULINO,
    FEMININO
}
```

(b)

APÊNDICE B – QUESTIONÁRIO

Parte I

Faça uma apresentação sobre sua qualificação profissional voltado para o MPS.BR (qualidade de software)

1. As modificações apresentadas no processo GRE atendem o MPS.BR?
2. As modificações realizadas no ciclo de vida apresentado continuam atendendo os quesitos do MPS.BR?
3. Em sua opinião, quais as desvantagens da inserção do MDA em um processo MPS.BR?
4. Em sua opinião, a inserção do MDA em um processo MPS.BR já instaurado, ofereceria benefícios ao processo? Se sim quais?

Outras Considerações:

Parte II

Nesta fase, na questão 1, serão apresentados slides com as respectivas explanações dos artefatos propostos, para o avaliador verificar se o resultado esperado do MPS.BR atende o quesito em questão.

1. Sobre os artefatos e conceitos mostrados do MDA, quais os resultados esperados do MPS.BR são cumpridos por esses artefatos/conceitos (totalmente, parcialmente ou não é abordado). Caso julgue necessário, comente sua reposta.

   a) GPR3 (nível G) – (Slide do Ciclo de Vida)
   ( ) Sim ( ) Parcialmente ( ) Não

   b) GRE1 (nível G) – (Slide do Documentos de caso Uso)
   ( ) Sim ( ) Parcialmente ( ) Não

   c) GRE3 (Nível G) –(Slide do Conceito de Mapeamento e Matriz de Relacionamento)
   ( ) Sim ( ) Parcialmente ( ) Não

   d) GRE5 (Nível G) –(Slide Conceito de transformação e mapeamento)
   ( ) Sim ( ) Parcialmente ( ) Não

e) MED2 (Nível F) – (Slide de Métricas pela ferramenta)
( ) Sim ( ) Parcialmente ( ) Não

f) MED3 (Nível F) – (Slide da Métricas executadas pela ferramenta.)
( ) Sim ( ) Parcialmente ( ) Não

g) MED4 (Nível F) – Slide do Procedimento executados pela ferramenta.)
( ) Sim ( ) Parcialmente ( ) Não

h) MED5 (Nível F) – (Slide da Análise de métricas realizada pela ferramenta.)
( ) Sim ( ) Parcialmente ( ) Não

i) AMP7 (Nível E) – (Slide dos Modelos gerados.)
( ) Sim ( ) Parcialmente ( ) Não

j) DFP4 (Nível E) – (Slide do Ciclo de Vida.)
( ) Sim ( ) Parcialmente ( ) Não

k) DRE1 (Nível D) – (Slide dos Documentos de Casos de Uso.)
( ) Sim ( ) Parcialmente ( ) Não

l) DRE4 (Nível D) –(Slide dos Documentos de Casos de Usos.)
( ) Sim ( ) Parcialmente ( ) Não

m) DRE6 (Nível D) – (Slides de Cenários de casos de Usos.)
( ) Sim ( ) Parcialmente ( ) Não

n) ITP2 (Nível D) – (Slides de Teste de Integração pela ferramenta.)
( ) Sim ( ) Parcialmente ( ) Não

o) PCP3 (Nível D) – (Slides dos Modelos apresentados .)
( ) Sim ( ) Parcialmente ( ) Não

p) PCP7 (Nível D) –(Slides dos Modelos gerados (regras))
( ) Sim ( ) Parcialmente ( ) Não

q) PCP8 (Nível D) – (Slides do Conceitos do MDA para gerar os modelos e a ferramenta de edição.)
( ) Sim ( ) Parcialmente ( ) Não

r) VAL2 (Nível D) – (Slides da Prototipação e testes.)
( ) Sim ( ) Parcialmente ( ) Não

s) VAL3 (Nível D) – (Slides de Testes pelas ferramentas.)
( ) Sim ( ) Parcialmente ( ) Não

t) VAL4 (Nível D) –(Slides de Testes pelas ferramentas.)
( ) Sim ( ) Parcialmente ( ) Não

u) VER2 (Nível D) –(Slides de Testes pelas Ferramentas)
( ) Sim ( ) Parcialmente ( ) Não

v) VER4 (Nível D) – (Slides de Testes pelas ferramentas.)
( ) Sim ( ) Parcialmente ( ) Não

w) DRU7 (Nível C) –(Slides de Modelos desenvolvidos.)
( ) Sim ( ) Parcialmente ( ) Não

2. Em sua opinião, o MDA apresenta artefatos/modelos que atendam resultados esperados no nível G do MPS.BR? Quais?

3. Em sua opinião, com a inserção do MDA no nível G do MPS.BR, irá ou não apresentar alguns resultados esperados do MPS.BR de níveis superiores? Justifique.

4. Em sua opinião, uma empresa que já possui o MDA como modelo de desenvolvimento, teria maior facilidade para implementar o MPS.BR do que uma empresa que execute o desenvolvimento orientado a código?

Outras considerações:

APÊNDICE C – RESPOSTAS DO QUESTIONÁRIO

Neste apêndice serão apresentadas as repostas de cada especialista para o questionário aplicado.

C.1 Especialista 1

Parte I

Faça uma apresentação sobre sua qualificação profissional voltado para o MPS.BR (qualidade de software)

Resposta: Sou implementador MPS dos 3 modelos MPS.SW/SV/RH. Sou especilista em Gerenciamento de Projetos Senai SC, especialista em Ciência da Computação UEL, Docente nos cursos MBA em Gestão de Projetos e Gestão Industrial, 17 anos na área de TI, 7 anos com Gerência de Projetos e Consultor.

1. As modificações apresentadas no processo GRE atendem o MPS.BR?

   Resposta: Os requisitos documentados traz um nível de entendimento melhor. O modelo pratica detalhamento dos requisitos, em conformidade com as práticas do nível 3 do CMMI e nível D do MPS.

2. As modificações realizadas no ciclo de vida apresentado continuam atendendo os quesitos do MPS.BR?

   Resposta: Sim, não afetam.

3. Em sua opinião, quais as desvantagens da inserção do MDA em um processo MPS.BR?

   Resposta: Para empresas menores o modelo pode burocratizar o processo, porém para sistemas ou projetos novos, a aplicação do modelo pode facilitar a abstração e entrega de requisitos consistentes.

4. Em sua opinião, a inserção do MDA em um processo MPS.BR já instaurado, ofereceria benefícios ao processo? Se sim quais?

   Resposta: Sim, na minha opinião o principal benefício é a definição de cenários.

Outras Considerações:

Parte II

Nesta fase, na questão 1, serão apresentados slides com as respectivas explanações dos artefatos propostos, para o avaliador verificar se o resultado esperado do MPS.BR atende o quesito em questão.

1. Sobre os artefatos e conceitos mostrados do MDA, quais os resultados esperados do MPS.BR são cumpridos por esses artefatos/conceitos (totalmente, parcialmente ou não é abordado). Caso julgue necessário, comente sua reposta.

   a) GPR3 (nível G) – (Slide do Ciclo de Vida)
   ( ) Sim (X) Parcialmente ( ) Não
   Comentário: Faltou o artefato de cronograma para complementar o resultado.

   b) GRE1 (nível G) – (Slide do Documentos de caso Uso)
   (X) Sim ( ) Parcialmente ( ) Não
   Comentário: Um dos artefatos apresentados, faltou a validação (aceitação do requisito)

   c) GRE3 (Nível G) –(Slide do Conceito de Mapeamento e Matriz de Relacionamento)
   (X) Sim ( ) Parcialmente ( ) Não

   d) GRE5 (Nível G) –(Slide Conceito de transformação e mapeamento)
   () Sim (X) Parcialmente ( ) Não
   Comentário: Para a análise da mudança é necessário o registro do impacto (análise) e aprovação da mudança.

   e) MED2 (Nível F) – (Slide de Métricas pela ferramenta)
   ( ) Sim (X) Parcialmente ( ) Não
   Comentário: Os objetivos de medição não está relacionados a métrica, aprovação da medição.

   f) MED3 (Nível F) – (Slide da Métricas executadas pela ferramenta.)
   ( ) Sim (X) Parcialmente ( ) Não
   Comentário: Faltou o procedimento para coleta e armazenamento dos resultados através do uso da ferramenta auxilia o processo.

g) MED4 (Nível F) – Slide do Procedimento executados pela ferramenta.)
( ) Sim ( ) Parcialmente (X) Não
Comentário: Falta definir a escala de como analisar a medição.

h) MED5 (Nível F) – (Slide da Análise de métricas realizada pela ferramenta.)
( ) Sim (X) Parcialmente ( ) Não
Comentário: Faltou a análise dos dados.

i) AMP7 (Nível E) – (Slide dos Modelos gerados.)
( ) Sim (X) Parcialmente ( ) Não
Comentário: Os modelos auxiliam na implementação dos ativos.

j) DFP4 (Nível E) – (Slide do Ciclo de Vida.)
( ) Sim (X) Parcialmente ( ) Não
Comentário: Por problemas da forma da abordagem do MDA, a definição do processo não prevê adaptações, isso precisa estar claro.

k) DRE1 (Nível D) – (Slide dos Documentos de Casos de Uso.)
(X) Sim ( ) Parcialmente ( ) Não

l) DRE4 (Nível D) –(Slide dos Documentos de Casos de Usos.)
(X) Sim ( ) Parcialmente ( ) Não

m) DRE6 (Nível D) – (Slides de Cenários de casos de Usos.)
(X) Sim ( ) Parcialmente ( ) Não

n) ITP2 (Nível D) – (Slides de Teste de Integração pela ferramenta.)
(X) Sim ( ) Parcialmente ( ) Não

o) PCP3 (Nível D) – (Slides dos Modelos apresentados .)
(X) Sim ( ) Parcialmente ( ) Não

p) PCP7 (Nível D) –(Slides dos Modelos gerados (regras))
(X) Sim ( ) Parcialmente ( ) Não

q) PCP8 (Nível D) – (Slides do Conceitos do MDA para gerar os modelos e a ferramenta de edição.)

(X)Sim ( ) Parcialmente ( ) Não

r) VAL2 (Nível D) – (Slides da Prototipação e testes.)
(X) Sim ( ) Parcialmente ( ) Não

s) VAL3 (Nível D) – (Slides de Testes pelas ferramentas.)
(X) Sim ( ) Parcialmente ( ) Não

t) VAL4 (Nível D) –(Slides de Testes pelas ferramentas.)
(X) Sim ( ) Parcialmente ( ) Não

u) VER2 (Nível D) –(Slides de Testes pelas Ferramentas)
( ) Sim (X) Parcialmente ( ) Não
Comentário: Utiliza o processo de revisão por pares.

v) VER4 (Nível D) – (Slides de Testes pelas ferramentas.)
( ) Sim (X) Parcialmente ( ) Não
Comentário: Utiliza o processo de revisão por pares.

w) DRU7 (Nível C) –(Slides de Modelos desenvolvidos.)
( ) Sim (X) Parcialmente ( ) Não
Comentário: No GRU1 o MDA pode fazer parte da estratégia de reutilização.

2. Em sua opinião, o MDA apresenta artefatos/modelos que atendam resultados esperados no nível G do MPS.BR? Quais?

   Resposta: Sim, os artefatos auxiliam no atendimento dos resultados GRE1, GRE3, GRE4, pois trazem informações para validações dos casos de usos e casos de teste, como também ao controle de mudanças (GRE5).

3. Em sua opinião, com a inserção do MDA no nível G do MPS.BR, irá ou não apresentar alguns resultados esperados do MPS.BR de níveis superiores? Justifique.

   Resposta: Sim, a inserção do MDA facilita a implementação de outros resultados, principalmente aos processos de engenharia do nível D.

4. Em sua opinião, uma empresa que já possui o MDA como modelo de desenvolvimento, teria maior facilidade para implementar o MPS.BR do que uma empresa que execute o desenvolvimento orientado a código?

   Resposta: Sim, principalmente na implementação de processos de engenharia, como dito anteriormente.

C.2 Especialista 2

Parte I

Faça uma apresentação sobre sua qualificação profissional voltado para o MPS.BR (qualidade de software)

Resposta:

- Implementador MPS.BR Software e Serviços Credenciado;
- Especialista em gestão de projetos (PMP);
- Trabalhou 12 anos em uma fabrica de software CMMI;
- Implementação em mais de 20 empresas (MPS.BR, CMMI, ITMARK e MOPROSOFT)

1. As modificações apresentadas no processo GRE atendem o MPS.BR?

   Resposta: Sim, as modificações melhoraram o conteúdo dos documentos.

2. As modificações realizadas no ciclo de vida apresentado continuam atendendo os quesitos do MPS.BR?

   Resposta: Sim, vale lembrar que o ciclo de vida deve estar refletido no cronograma.

3. Em sua opinião, quais as desvantagens da inserção do MDA em um processo MPS.BR?

   Resposta: O MDA não possui nivelamento como o MPS.BR, desta forma algumas empresas poderão ter dificuldades na implementação.

4. Em sua opinião, a inserção do MDA em um processo MPS.BR já instaurado, ofereceria benefícios ao processo? Se sim quais?

   Resposta: A vantagem é a automação e estruturação da documentação dos requisitos.

Outras Considerações:

Parte II

Nesta fase, na questão 1, serão apresentados slides com as respectivas explanações dos artefatos propostos, para o avaliador verificar se o resultado esperado do MPS.BR atende o quesito em questão.

1. Sobre os artefatos e conceitos mostrados do MDA, quais os resultados esperados do MPS.BR são cumpridos por esses artefatos/conceitos (totalmente, parcialmente ou não é abordado). Caso julgue necessário, comente sua reposta.

a) GPR3 (nível G) – (Slide do Ciclo de Vida)
(X) Sim () Parcialmente ( ) Não

b) GRE1 (nível G) – (Slide do Documentos de caso Uso)
( ) Sim (X) Parcialmente ( ) Não
Comentário: Não esta explicito o aceite do cliente no modelo.

c) GRE3 (Nível G) –(Slide do Conceito de Mapeamento e Matriz de Relacionamento)
(X) Sim ( ) Parcialmente ( ) Não

d) GRE5 (Nível G) –(Slide Conceito de transformação e mapeamento)
() Sim (X) Parcialmente ( ) Não
Comentário: Não há garantia que as mudanças foram incorporadas no planejamento do projeto.

e) MED2 (Nível F) – (Slide de Métricas pela ferramenta)
( ) Sim ( ) Parcialmente (X) Não
Comentário: Não é o foco do MDA a documentação de procedimentos.

f) MED3 (Nível F) – (Slide da Métricas executadas pela ferramenta.)
( ) Sim ( ) Parcialmente (X) Não
Comentário: Não é o foco do MDA a documentação de procedimentos.

g) MED4 (Nível F) – Slide do Procedimento executados pela ferramenta.)
( ) Sim ( ) Parcialmente (X) Não
Comentário: Não é o foco do MDA a documentação de procedimentos.

h) MED5 (Nível F) – (Slide da Análise de métricas realizada pela ferramenta.)
(X) Sim () Parcialmente ( ) Não

i) AMP7 (Nível E) – (Slide dos Modelos gerados.)
(X) Sim ( ) Parcialmente ( ) Não

j) DFP4 (Nível E) – (Slide do Ciclo de Vida.)
(X) Sim () Parcialmente ( ) Não

k) DRE1 (Nível D) – (Slide dos Documentos de Casos de Uso.)
(X) Sim ( ) Parcialmente ( ) Não

l) DRE4 (Nível D) –(Slide dos Documentos de Casos de Usos.)
(X) Sim ( ) Parcialmente ( ) Não

m) DRE6 (Nível D) – (Slides de Cenários de casos de Usos.)
(X) Sim ( ) Parcialmente ( ) Não

n) ITP2 (Nível D) – (Slides de Teste de Integração pela ferramenta.)
(X) Sim ( ) Parcialmente ( ) Não

o) PCP3 (Nível D) – (Slides dos Modelos apresentados .)
(X) Sim ( ) Parcialmente ( ) Não

p) PCP7 (Nível D) –(Slides dos Modelos gerados (regras))
(X) Sim ( ) Parcialmente ( ) Não

q) PCP8 (Nível D) – (Slides do Conceitos do MDA para gerar os modelos e a ferramenta de edição.)
(X)Sim ( ) Parcialmente ( ) Não

r) VAL2 (Nível D) – (Slides da Prototipação e testes.)
( ) Sim (X) Parcialmente ( ) Não
Comentário: A estratégia de validação não contempla as evidências solicitadas pelo MPS.BR.

s) VAL3 (Nível D) – (Slides de Testes pelas ferramentas.)
(X) Sim ( ) Parcialmente ( ) Não

t) VAL4 (Nível D) –(Slides de Testes pelas ferramentas.)
(X) Sim ( ) Parcialmente ( ) Não

u) VER2 (Nível D) –(Slides de Testes pelas Ferramentas)
( ) Sim ( ) Parcialmente (X) Não
Comentário: Utiliza o processo de revisão por pares.

v) VER4 (Nível D) – (Slides de Testes pelas ferramentas.)
( ) Sim (X) Parcialmente ( ) Não
Comentário: As revisões por pares não fazem parte do escopo do MDA.

w) DRU7 (Nível C) –(Slides de Modelos desenvolvidos.)
( ) Sim ( ) Parcialmente (X) Não

2. Em sua opinião, o MDA apresenta artefatos/modelos que atendam resultados esperados no nível G do MPS.BR? Quais?

   Resposta: O MDA atende os resultados GRE1, GRE3 e GRE5. A gestão de projetos (GPR) não faz parte do escopo do MDA.

3. Em sua opinião, com a inserção do MDA no nível G do MPS.BR, irá ou não apresentar alguns resultados esperados do MPS.BR de níveis superiores? Justifique.

   Resposta: O MDA apresentará resultados esperados em processos de níveis D e C relacionados a Engenharia de Software.

4. Em sua opinião, uma empresa que já possui o MDA como modelo de desenvolvimento, teria maior facilidade para implementar o MPS.BR do que uma empresa que execute o desenvolvimento orientado a código?

   Resposta: Se a empresa usa o MDA ela já possui maturidade para seguir uma metodologia e isso poderá ajudar na implantação do MPS.BR.

Outras Considerações:

O MDA atende o DRE3.

C.3 Especialista 3

Parte I

Faça uma apresentação sobre sua qualificação profissional voltado para o MPS.BR (qualidade de software)

Resposta: Trabalhei por 8 anos na área de qualidade de software, mais especificamente em desenvolvimento e melhoria de processos. Realizei o curso C1 do MPS.BR para conhecer os detalhes sobre o modelo.

1. As modificações apresentadas no processo GRE atendem o MPS.BR?

   Resposta: Sim, atende, pois as modificações buscaram estabelecer um padrão maior mais sofisticado e detalhado de como documentar os requisitos

2. As modificações realizadas no ciclo de vida apresentado continuam atendendo os quesitos do MPS.BR?

   Resposta: Sim, continuaram atendendo, pois os resultados esperados do MPS.BR continuam sendo atendido, com a diferença que são atendidos de forma automatizada.

3. Em sua opinião, quais as desvantagens da inserção do MDA em um processo MPS.BR?

   Resposta: Não vejo desvantagem, apenas a necessidade de realizar treinamentos para a equipe aprender o novo modo de trabalho.

4. Em sua opinião, a inserção do MDA em um processo MPS.BR já instaurado, ofereceria benefícios ao processo? Se sim quais?

   Resposta: Com certeza oferece. O principal é a automatização das tarefas do processo.

Outras Considerações:

Parte II

Nesta fase, na questão 1, serão apresentados slides com as respectivas explanações dos artefatos propostos, para o avaliador verificar se o resultado esperado do MPS.BR atende o quesito em questão.

1. Sobre os artefatos e conceitos mostrados do MDA, quais os resultados esperados do MPS.BR são cumpridos por esses artefatos/conceitos (totalmente, parcialmente ou não é abordado). Caso julgue necessário, comente sua reposta.

a) GPR3 (nível G) – (Slide do Ciclo de Vida)
( ) Sim (X) Parcialmente ( ) Não

b) GRE1 (nível G) – (Slide do Documentos de caso Uso)
(X) Sim ( ) Parcialmente ( ) Não

c) GRE3 (Nível G) –(Slide do Conceito de Mapeamento e Matriz de Relacionamento)
(X) Sim ( ) Parcialmente ( ) Não

d) GRE5 (Nível G) –(Slide Conceito de transformação e mapeamento)
(X) Sim ( ) Parcialmente ( ) Não

e) MED2 (Nível F) – (Slide de Métricas pela ferramenta)
( ) Sim (X) Parcialmente ( ) Não

f) MED3 (Nível F) – (Slide da Métricas executadas pela ferramenta.)
( ) Sim ( ) Parcialmente (X) Não

g) MED4 (Nível F) – Slide do Procedimento executados pela ferramenta.)
( ) Sim ( ) Parcialmente (X) Não

h) MED5 (Nível F) – (Slide da Análise de métricas realizada pela ferramenta.)
( ) Sim (X) Parcialmente ( ) Não

i) AMP7 (Nível E) – (Slide dos Modelos gerados.)
(X) Sim ( ) Parcialmente ( ) Não

j) DFP4 (Nível E) – (Slide do Ciclo de Vida.)
(X) Sim () Parcialmente ( ) Não

k) DRE1 (Nível D) – (Slide dos Documentos de Casos de Uso.)
(X) Sim ( ) Parcialmente ( ) Não

l) DRE4 (Nível D) –(Slide dos Documentos de Casos de Usos.)
(X) Sim ( ) Parcialmente ( ) Não

m) DRE6 (Nível D) – (Slides de Cenários de casos de Usos.)
(X) Sim ( ) Parcialmente ( ) Não

n) ITP2 (Nível D) – (Slides de Teste de Integração pela ferramenta.)
(X) Sim ( ) Parcialmente ( ) Não

o) PCP3 (Nível D) – (Slides dos Modelos apresentados .)
(X) Sim ( ) Parcialmente ( ) Não

p) PCP7 (Nível D) –(Slides dos Modelos gerados (regras))
(X) Sim ( ) Parcialmente ( ) Não

q) PCP8 (Nível D) – (Slides do Conceitos do MDA para gerar os modelos e a ferramenta de edição.)
(X)Sim ( ) Parcialmente ( ) Não

r) VAL2 (Nível D) – (Slides da Prototipação e testes.)
( ) Sim (X) Parcialmente ( ) Não

s) VAL3 (Nível D) – (Slides de Testes pelas ferramentas.)
( ) Sim (X) Parcialmente ( ) Não

t) VAL4 (Nível D) –(Slides de Testes pelas ferramentas.)
( ) Sim (X) Parcialmente ( ) Não

u) VER2 (Nível D) –(Slides de Testes pelas Ferramentas)
( ) Sim (X) Parcialmente ( ) Não

v) VER4 (Nível D) – (Slides de Testes pelas ferramentas.)
( ) Sim (X) Parcialmente ( ) Não

w) DRU7 (Nível C) –(Slides de Modelos desenvolvidos.)
( ) Sim ( ) Parcialmente (X) Não

2. Em sua opinião, o MDA apresenta artefatos/modelos que atendam resultados esperados no nível G do MPS.BR? Quais?

Resposta: Sim, atende aos resultados esperados 1,3 e 5 do GRE e parcialmente do resultado esperado 3 do GPR.

3. Em sua opinião, com a inserção do MDA no nível G do MPS.BR, irá ou não apresentar alguns resultados esperados do MPS.BR de níveis superiores? Justifique.

   Resposta: Alguns resultados esperados em níveis superiores serão atendidos, como por exemplo, o DFP, por meio da definição do que é o próprio MDA.

4. Em sua opinião, uma empresa que já possui o MDA como modelo de desenvolvimento, teria maior facilidade para implementar o MPS.BR do que uma empresa que execute o desenvolvimento orientado a código?

   Resposta: Acredito que teria uma enorme facilidade, pois a estruturação utilizando o MDA é uma prévia da estruturação dos processos do MPS.BR.

APÊNDICE D – EXPLICAÇÃO SOBRE OS RESULTADOS ESPERADOS LISTADOS

A seguir serão apresentadas as explicações sobre os resultados esperados listados no questionário parte II. Será realizada uma breve explicação sobre o que trata este resultado e o porquê é antecipado pela instância desenvolvida e conceitos do MDA. Nesta abordagem será discutido o porquê o RE é considerado antecipado, não sendo julgado se de forma completa ou parcial.

GPR3- No MPS.BR, este RE determina a definição do modelo e das fases do ciclo de vida. Com o MDA este RE pode ser antecipado, pois o MDA já traz fases estabelecidas (CM-PIM-PSM), sendo possível estabelecer mais etapas e atividades caso necessárias. A Figura 2 apresenta um ciclo de vida proposta com as fases do MDA.

GRE1- Este RE visa entender os requisitos junto aos fornecedores , documentá-los e obter o aceite do cliente. O MDA antecipa este RE, pois documenta estes requisitos com os artefatos gerados no CIM.

GRE3- No MPS.BR este RE é sobre a necessidade de estabelecer e manter um mecanismo de rastreabilidade bidirecional entre os requisitos. Com MDA este RE é antecipado, pois a rastreabilidade é possível ser realizada pelos conceitos das transformações entre as fases, além dos recursos de algumas ferramentas.

GRE5- Este RE aborda o gerenciamento de mudanças nos requisitos ao longo do tempo. Diversas atividades são abordas neste RE como, por exemplo, registro da necessidade da mudança, histórico de decisões e análise de impacto . O MDA antecipa esse RE, pois ele gerencia bem as mudanças, visto que as mudanças devem ser executadas na fazem CIM e serem propagadas até o código.

MED 2- Neste RE deve-se selecionar um conjunto de medidas, levando em consideração diversos fatores, como por exemplo, facilidade para a coleta de dados, viabilidade para a coleta de dados e etc. De acordo com [22] e [49], uma vez os modelos capturados como dados semânticos, as análises como métricas devem ser possíveis de ser executadas, sendo que as ferramentas deveriam fornecer suporte. Portanto, como o MDA possui essa proposta, isso viabiliza e facilita a coleta de dados para a métrica.

MED3- Em MED3 os procedimentos para a coleta devem ser documentados, juntamente com os dados que foram coletados. Como o MDA possui a proposta de métricas por ferramentas, essa coleta já seria realizada desta maneira. Logo o procedimento seria este.

MED4- Este RE requisita para especificar os procedimentos para a análise das

medidas. Com o MDA, a análise é realizada por meio das ferramentas.

MED5- Em MED5 os dados são coletados e analisados. Portanto com o MDA, a ferramenta possibilita essa coleta de dados.

AMP7- Este RE objetiva-se a implantar ativos de processos organizacionais na empresa. Os ativos de processo organizacional incluem definição do conjunto de processos padrão da organização, modelos de documentos a serem utilizados nos projetos, diretrizes sobre a execução de tarefas, documentação de projetos passados. Portanto, o MDA tanto na sua definição quanto nos seus artefatos já antecipa diversos ativos de processos organizacionais.

DFP4- O DFP4 define o ciclo de vida para ser utilizado e mantido, sendo que neste caso é previsto para variedade de projetos. Como já foi discutido no GPR3, o MDA aborda suas fases e com isso se pode considerar um ciclo de vida.

DRE1- Neste RE busca-se identificar as necessidades, expectativas e restrições do cliente, sendo que o próprio guia do MPS.BR sugere a utilização do caso de uso. Logo, o MDA neste caso antecipa este RE, pois o caso de uso é um dos artefatos gerados pela a instância proposta.

DRE4- No DRE4 os requisitos funcionais e não-funcionais devem ser documentados. Essa documentação com o MDA ocorre no CIM, portanto este RE é antecipado.

DRE6- Neste RE é definido o desenvolvimento de cenários. Portanto, com o MDA este RE é antecipado, pois os cenários são desenvolvidos no CIM para cada caso de uso.

ITP2- No ITP2 é esperado manter um ambiente de integração do produto, além de ferramentas para testes. De acordo com [49], [22] as ferramentas de MDA deveriam fornecer suporte para testes. Portanto, além de todos os modelos serem elaborados nas mesmas ferramentas, os testes também podem ser executados.

PCP3- Neste RE o produto e o componente do produto são documentados. No MDA, estes são todos os diagramas elaborados, portanto este RE é antecipado.

PCP7- Para este RE é esperado que a documentação seja desenvolvida. Logo com o MDA, a documentação é desenvolvida em todas as suas etapas, sendo que este RE é antecipado.

PCP8- No PCP8 a documentação deve ser mantida de acordo com critérios. No MDA existem regras de elaboração dos modelos, como as utilizadas para o desenvolvimento do CIM e o MOF. Também, pode-se destacar o padrão que é mantido pelas transformações, assim não haverá um modelo com a escrita diferente do outro.

VAL2- Neste RE deve-se estabelecer uma estratégia de validação, sendo que esta validação pode ser realizada através de testes e prototipação de telas. De acordo com, [49] e [22] tanto a validação quanto os testes deveriam ser suportados por ferramentas MDA.

No trabalho a prototipação de tela foi um artefato gerado e alguns tipos de testes podem ser realizados na ferramenta utilizada.

VAL3 – No VAL3 é estabelecido os critérios e procedimentos para a validação dos produtos. Como o MDA, traz em suas ferramentas os testes, isso já é estabelecido pela própria ferramenta, portanto, pode ser antecipado.

VAL4- Neste RE é esperado que fossem executadas as atividades de validações, sendo a principal forma de se executar essa atividade é através de testes. Como já ressaltado anteriormente, as ferramentas de MDA dão (ou pelo menos deveriam) suporte a teste, logo este RE é antecipado.

VER 2- Para o VER2 deve-se estabelecer e implementar uma estratégia de verificação, estabelecendo cronograma, revisores envolvidos, métodos para verificação e qualquer material a ser utilizado na verificação. Neste caso, o MDA só anteciparia o método de verificação, que seria os testes pelas ferramentas.

VER4- Neste RE é executado a atividade de verificação, incluindo testes e revisões por pares. Com o MDA é possível a realização de alguns testes.

DRU7- O DRU é um processo que visa a reutilização. Já o DRU7 objetiva-se no desenvolvimento de um modelo de domínio para potencial reutilização. Na instância MDA desenvolvida, os artefatos gerados podem ser apresentados como possíveis modelos de domínios, antecipando assim este RE.

ANEXO A – CONVENÇÕES BNF

Para anotações textuais, uma variante do formulário de BNF-*Backus-Naur Form* é freqüentemente usada para especificar os formatos legais. As convenções deste BNF são [82]:

- Todos os não-terminais estão em itálico e fechados entre colchetes angulares (por exemplo, <*não-terminal*>) .
- Todos os terminais (palavras-chave, strings, etc.) são colocados entre aspas simples (Por exemplo, 'ou').
- As definições de regra de produção não-terminal são significadas com o operador '::=':
- Repetição de um item é significada por um asterisco colocado após o item: '*'.
- As escolhas alternativas numa produção são separadas pelo símbolo '|' (por exemplo, <Alternativa-A> | <Alternativa-B>).
- Os itens que são opcionais são colocados entre colchetes (por exemplo, [<item-x>]).
- Quando os itens precisam ser agrupados, eles são incluídos em parênteses simples; por exemplo:(<Item-1> | <item-2>)* significa uma seqüência de um ou mais itens, cada um dos quais é <item-1> ou <item-2>.

B TRABALHOS PUBLICADOS PELO AUTOR

1. Vanessa Matias Leite, Emmanuel da Costa Gallo, Jandira Guenka Palma, **Intersection of MPS.BR-E and SPICE Models Focused on Projects for the Automotive Industry. In: ICSEA 2015 : The Tenth International Conference on Software Engineering Advances**,ICSEA- The Tenth International Conference on Software Engineering Advances, 2015. v.1. p. 268-273. (Qualis-CC-2012, B3).

2. Wilson Hissamu Shirado, Vanessa Matias Leite, Jandira Guenka Palma, **Model Driven Architecture for Development Test Automation Tools. In: The World Congress in Computer Science, Computer Engineering and Applied Computing**, SERP- Proceedings of the International Conference on Software Engineering Research and Practice, 2016. p. 167-173. (Qualis-CC-2012, B2).

3. Marcio de Abreu Moreira, Jéssica Tomaz Silva ,Vanessa Matias Leite, Jandira Guenka Palma, **IMPLEMENTATION OF MOPROSOFT IN A SMALL COMPANY WITH MPS.BR-F**, PROCEEDINGS OF THE 8th IADIS INTERNATIONAL CONFERENCE, 2015. v.1. p. 257-260. (Qualis-CC-2012, B5).

4. Vanessa Matias Leite, Jandira Guenka Palma **Definition of a Computing Independent Model and Rules for Transformation Focused on the Model-View-Controller Architecture**, International Journal of Computer, Electrical, Automation, Control and Information Engineering, 2017. v.11. p. 225-232.