



UNIVERSIDADE FEDERAL DO CEARÁ
CAMPUS QUIXADÁ
BACHARELADO EM ENGENHARIA DE SOFTWARE

ANDERSON GONÇALVES UCHÔA

**REMINDER: UMA ABORDAGEM PARA MODELAGEM DE PROPRIEDADES
NÃO-FUNCIONAIS EM LINHAS DE PRODUTO DE SOFTWARE DINÂMICAS**

QUIXADÁ – CEARÁ

2016

ANDERSON GONÇALVES UCHÔA

REMINDER: UMA ABORDAGEM PARA MODELAGEM DE PROPRIEDADES
NÃO-FUNCIONAIS EM LINHAS DE PRODUTO DE SOFTWARE DINÂMICAS

Monografia apresentada no curso de Engenharia de Software da Universidade Federal do Ceará, como requisito parcial à obtenção do título de bacharel em Engenharia de Software. Área de concentração: Computação.

Orientadora: Prof^ª. Dra. Carla Ilane
Moreira Bezerra

QUIXADÁ – CEARÁ

2016

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Biblioteca Universitária
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

- U19r Uchôa, Anderson Gonçalves.
ReMINDER: Uma Abordagem para Modelagem de Propriedades Não-Funcionais em Linhas de Produto de Software Dinâmicas / Anderson Gonçalves Uchôa. – 2016.
136 f. : il. color.
- Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Campus de Quixadá, Curso de Engenharia de Software, Quixadá, 2016.
Orientação: Profa. Dra. Carla Ilane Moreira Bezerra.
1. Engenharia de Linha de Produto de Software. 2. Non-Functional Requirements. 3. Modelo de Características. I. Título.

CDD 005.1

ANDERSON GONÇALVES UCHÔA

REMINDER: UMA ABORDAGEM PARA MODELAGEM DE PROPRIEDADES
NÃO-FUNCIONAIS EM LINHAS DE PRODUTO DE SOFTWARE DINÂMICAS

Monografia apresentada no curso de Engenharia de Software da Universidade Federal do Ceará, como requisito parcial à obtenção do título de bacharel em Engenharia de Software. Área de concentração: Computação.

Aprovada em: __/__/____

BANCA EXAMINADORA

Prof^ª. Dra. Carla Ilane Moreira Bezerra (Orientadora)
Campus Quixadá
Universidade Federal do Ceará – UFC

Prof. Dr. Marcos Antonio de Oliveira
Campus Quixadá
Universidade Federal do Ceará - UFC

Prof. Msc. Camilo Camilo Almendra
Campus Quixadá
Universidade Federal do Ceará - UFC

A Deus.

Aos meus pais, irmãos e amigos.

AGRADECIMENTOS

Agradeço primeiramente a Deus, por me permitir sonhar e tornar todos estes sonhos realidade. À meus pais pela educação e pelo carinho, que apesar de todas as dificuldades e desafios, que os foram impostos, sempre se mostraram confiantes e crentes na minha vontade de vencer e seguir meus sonhos.

À Prof^a. Dra. Carla Ilane Moreira Bezerra, por sua dedicação, pela confiança, pela oportunidades oferecidas, pelo excelente trabalho como orientadora e principalmente de sua sinceridade em parabenizar pelos acertos e em exigir qualidade e comprometimento quando erros foram cometidos. Este trabalho sem a sua orientação não teria a mesma qualidade e resultados positivos.

Aos professores participantes da banca examinadora, Prof. Msc. Camilo Almendra e Prof. Dr Marcos Oliveira, pelo tempo, pelas valiosas colaborações e sugestões.

À Universidade Federal do Ceará - Campus Quixadá, por fornecer um ensino de alta qualidade com professores excelentes em ensino e pesquisa, que me proporcionaram uma educação de excelência.

Aos meus colegas e amigos, que mesmo presentes ou distantes, motivaram-me na execução do trabalho e alegraram-se com mais esta conquista em minha vida. Em especial Luan, Amanda, Robson, Kadu, Cayk, Alexsandro, Állef, Randerson, Ítalos e Mauro pela amizade durante o último ano da graduação.

“Eu tentei 99 vezes e falhei. Mas na centésima tentativa eu consegui. Nunca Desista de seus objetivos, mesmo que eles pareçam impossíveis. A próxima tentativa pode ser a vitoriosa.”

(Albert Einstein)

RESUMO

Linhas de Produto de Software (LPS) representam um conjunto de sistemas de software que compartilham de um conjunto de *features* comuns e gerenciadas, que satisfazem as necessidades de um segmento de mercado particular ou missão. No entanto, a representação de variabilidade dos produtos gerados por LPSs é feita apenas de forma estática, ou seja, a adaptação ocorre em tempo de projeto. Contudo, com o surgimento de sistemas sensíveis ao contexto, a construção de sistemas tem se tornado mais complexa, dado as mudanças de requisitos e restrições conforme o ambiente no qual está inserido, exigindo um alto grau de adaptação em tempo de execução. Esses sistemas podem ser conceitualizados como Linhas de Produtos de Software Dinâmica (LPSD). Um dos desafios para construir um LPSD é desenvolver um mecanismo para incorporar cenários com adaptações de contexto e propriedades não-funcionais (PNFs). Para incorporar esses mecanismo ao modelo de *features* de LPSDs, uma boa estratégia consiste em utilizar uma abordagem para guiar a identificação e representação destes mecanismos. Neste cenário, o objetivo deste trabalho é representar PNFs e cenários de adaptações de contexto em modelos de *features* de LPSDs. Para isso, foi elaborada uma abordagem capaz de identificar PNFs, restrições e cenários de adaptações de contexto, ativação e desativação de *features*, denominada ReMINDER. Também foi desenvolvida uma extensão da ferramenta DyMMer, para suportar a abordagem. Para avaliação da abordagem e da ferramenta, foram realizados dois estudos. O primeiro foi um experimento controlado, para analisar o processo definido na abordagem ReMINDER, com a finalidade de caracterizar a sua aplicação, em relação à identificação de PNFs e cenários de adaptações de contexto, com suas respectivas restrições, para apoiar a modelagem de *features* e representação de PNFs em modelos de *features* de LPSDs. O segundo estudo foi a aplicação de um teste de usabilidade. Esse teste teve como objetivo avaliar a usabilidade da ferramenta estendida por meio da análise de quatro fatores: i) satisfação geral; ii) usabilidade da ferramenta; iii) qualidade da informação; e iv) qualidade da interface. De modo geral, após a análise e interpretação de todos os resultados obtidos, conclui-se que a abordagem auxiliou na identificação das PNFs e a sua relação com o comportamento do modelo de *features* por meio de restrições de interdependência. Além disso, também foi verificado que a ferramenta estendida possui uma boa usabilidade, sendo útil para operacionalização da abordagem.

Palavras-chave: Linhas de Produtos de Software Dinâmicas. Propriedades Não-Funcionais. Modelo de *Features* de LPSD.

ABSTRACT

Software Product Lines (SPL) represent a set of software systems that share a set of common features and managed to satisfy the needs of a particular market segment or mission. However, the representation of variability of products generated by SPL is made only of static way, that is, the adaptation occurs at design time. However, with the advent of context-sensitive systems, the development of systems has become more complex, given the changes of requirements and restrictions according to the environment to which it is inserted, requiring a high degree of adaptation. These systems can be conceptualized as Dynamic Software Product Lines (DSPL). One of the major challenges to build a DSPL is to develop a mechanism to incorporate scenarios with context adaptation and non-functional properties (NFP). To incorporate these mechanisms into the DSPL features model, a good strategy is to use an approach to guide the identification and representation of these mechanisms. In this scenario, the objective of this work is to represent NFPs and context adaptations scenarios in DSPL feature models. For this, an approach was developed to identify NFPs, constraints and context adaptations scenarios, activation and deactivation of features. The approach is called ReMINDER. An extension of the DyMMer tool was also developed to support the approach. Two studies were carried out to evaluate this work. The first one was a controlled experiment, to analyze the process defined in the ReMINDER approach, with the purpose of characterizing its application, in relation to the identification of NFPs and context adaptation scenarios, with their respective constraints, to support feature modeling and representation of constraints in DSPL feature models. The second study was the application of a usability test. This test was conducted to evaluate the usability of the tool extend through the analysis of four factors: (i) general satisfaction; ii) usability of the tool; (iii) the quality of information; and (iv) the quality of the interface. In general, after analysis and interpretation of all results obtained from studies, it was concluded that the approach has assisted in the identification of NFPs and its relations with the behavior of the feature models through interdependencies constraints and that the extended tool has good usability and is useful for operationalizing the approach.

Keywords: Dynamic Software Product Lines. Non-Functional Properties. DSPL Feature Models

LISTA DE FIGURAS

Figura 1 – Ciclo de vida de atividades de uma LPSD.	23
Figura 2 – Taxonomia para RNFs em linhas de produtos de software.	26
Figura 3 – Modelo de <i>features</i> básico.	30
Figura 4 – Modelo de <i>features</i> dinâmico em termos de ativação e desativação de <i>features</i>	32
Figura 5 – Modelo de <i>features</i> com notação da abordagem (Ext-FM).	34
Figura 6 – Modelo de <i>features</i> com notação da abordagem (ISPLM).	35
Figura 7 – Modelo de <i>features</i> com notação da abordagem (FM-QC).	36
Figura 8 – Representação arquitetural da DyMMer e S.P.L.A.R.	37
Figura 9 – Visualização do modelo de <i>features</i>	38
Figura 10 – Edição do modelo de <i>features</i>	39
Figura 11 – Ativação e desativação de <i>features</i>	39
Figura 12 – Procedimentos realizados durante a execução deste trabalho.	42
Figura 13 – Domínio exemplo.	50
Figura 14 – Visão geral da abordagem.	52
Figura 15 – Catálogo de PNFs de acordo com as características e subcaracterísticas de qualidade da norma ISO/IEC (2011).	54
Figura 16 – Exemplo de Modelo de <i>features</i> de qualidade	56
Figura 17 – Exemplo da representação de restrições de interdependências entre <i>Features</i> e PNFs	56
Figura 18 – Exemplo de regras de contexto	60
Figura 19 – Exemplo da representação de restrições de interdependências entre <i>Features</i> de Contexto e PNFs	64
Figura 20 – Visão arquitetural em camadas da ferramenta	76
Figura 21 – Diagrama de pacotes.	77
Figura 22 – Diagrama de classes.	78
Figura 23 – Adicionando característica de qualidade	80
Figura 24 – Selecionando característica de qualidade	80
Figura 25 – Característica de qualidade adicionada	81
Figura 26 – Adicionando subcaracterística de qualidade	82
Figura 27 – Selecionando subcaracterística de qualidade	82
Figura 28 – Subcaracterística de qualidade adicionada	83

Figura 29 – Adicionando propriedades não-funcionais	84
Figura 30 – Selecionando quantificação de uma propriedade não-funcional	85
Figura 31 – Propriedade não-funcional adicionada	85
Figura 32 – Adicionando informação de contexto	86
Figura 33 – Informação de contexto adicionada	87
Figura 34 – Adicionando quantificadores	88
Figura 35 – Especificando restrições de integridade de contexto	89
Figura 36 – Selecionando tipo de restrição de interdependência	90
Figura 37 – Selecionando PNF impactada pela restrição e adicionando restrição	90
Figura 38 – Adicionando múltiplos cenários de adaptações de contexto	91
Figura 39 – Visualizando a configuração do modelo de <i>features</i>	92

LISTA DE TABELAS

Tabela 1 – Respostas do questionário.	94
Tabela 2 – Dados referente a utilidade da ferramenta.	96
Tabela 3 – Dados referente a qualidade da informação.	96
Tabela 4 – Dados referente a qualidade da interface.	97

LISTA DE QUADROS

Quadro 1 – Qualidade do produto características e subcaracterísticas	28
Quadro 2 – PNFs que emergem em tempo de execução	29
Quadro 3 – Comparação de abordagens que representam PNFs	44
Quadro 4 – Classificação de tipos de contexto em categorias	58
Quadro 5 – Modelo para identificação de cenários de adaptações de contexto	59
Quadro 6 – Mapeamento de restrições de ativação e desativação de modelos de <i>features</i> de LPSD	61
Quadro 7 – Modelo para identificação de restrições de ativação e desativação de <i>features</i> de acordo com cada cenário de adaptação de contexto	62
Quadro 8 – Modelo para identificação de restrições de interdependências entre <i>features</i> e PNFs em um determinado cenário de adoção de contexto.	63
Quadro 9 – Descrição do objetivo do estudo de observação.	66
Quadro 10 – Resultados esperados para a primeira tarefa.	67
Quadro 11 – Resumo da caracterização dos participantes.	69
Quadro 12 – Tempo de cada participante.	69
Quadro 13 – Respostas do P1 apresentadas no modelo de <i>features</i> de qualidade.	70
Quadro 14 – Respostas do P1 das restrições de interdependências do cenário 1	71
Quadro 15 – Respostas do P1 das restrições de interdependências do cenário 2	71
Quadro 16 – Respostas do P2 apresentadas no modelo de <i>features</i> de qualidade	72
Quadro 17 – Respostas do P2 das restrições de interdependências do cenário 1.	73
Quadro 18 – Respostas do P2 das restrições de interdependências do cenário 2.	73

LISTA DE ABREVIATURAS E SIGLAS

LPS	Linhas de Produto de Software
RF	Requisito Funcional
LPSD	Linhas de Produto de Software Dinâmicas
FODA	Análise de Domínio orientada à Feature
RNF	Requisitos Não-Funcionais
ELPS	Engenharia de Linha de Produtos de Software
PNF	Propriedades Não-Funcionais
DyMMer	<i>Dynamic Feature Model Tool Based on Measures</i>

SUMÁRIO

1	INTRODUÇÃO	17
2	FUNDAMENTAÇÃO TEÓRICA	21
2.1	Linha de Produto de Software Dinâmicas	21
2.2	Requisitos de Linha de Produto de Software Dinâmica	24
2.2.1	Requisitos Não-Funcionais	24
2.3	Modelo de <i>Features</i>	29
2.3.1	Notações do Modelo de <i>Features</i> com RNFs	33
2.3.1.1	Extended Feature Model (Ext-FM)	33
2.3.1.2	Integrated Software Product Line Model (ISPLM)	34
2.3.1.3	Feature Models and Quality Criteria (FM-QC)	35
2.4	Ferramenta DyMMer	36
3	TRABALHOS RELACIONADOS	40
4	PROCEDIMENTOS METODOLÓGICOS	42
4.1	Identificação e comparação de abordagens existentes para PNFs em LPSs e LPSDs	43
4.2	Identificação de elementos do modelo de <i>features</i> de LPSDs para representar PNFs e adaptações de contexto	44
4.3	Definição da abordagem para acomodar os elementos identificados no modelo de <i>features</i> de LPSDs	45
4.4	Estudo e análise da ferramenta DyMMer	46
4.5	Definição de mecanismos de extensão da ferramenta DyMMer	46
4.6	Validação da abordagem	47
4.7	Validação da usabilidade da ferramenta estendida	47
4.8	Análise dos resultados	47
5	REMINDER: UMA ABORDAGEM PARA MODELAGEM DE PROPRIEDADES NÃO-FUNCIONAIS EM LINHAS DE PRODUTOS DE SOFTWARE DINÂMICAS	49
5.1	Exemplo para Guiar a Abordagem	50
5.2	ReMINDER: Visão Geral	50
5.3	Fase I - Identificação e Representação de PNFs no Modelo de <i>Features</i>	53
5.3.1	Passo I - Identificação de Características de Qualidade	54

5.3.2	<i>Passo II - Identificação de Subcaracterísticas de Qualidade</i>	55
5.3.3	<i>Passo III - Identificação de PNFs</i>	55
5.3.4	<i>Passo IV - Representação das PNFs</i>	55
5.3.5	<i>Passo V - Modelagem de Restrições de Interdependências entre Features e PNFs</i>	56
5.4	Fase II - Identificação de Restrições e Cenários de Adaptações de Contexto	57
5.4.1	<i>Passo I - Identificação de Cenários de Adaptações de Contexto</i>	57
5.4.2	<i>Passo II - Definição das Regras de Contexto</i>	59
5.4.3	<i>Passo III - Identificação das Restrições entre Features e Cenários de Adaptações de Contexto</i>	60
5.4.4	<i>Passo IV - Definição das Restrições de Interdependências entre PNFs e Features de Contexto</i>	62
5.4.5	<i>Passo V - Modelagem de Restrições de Interdependências entre Features de Contexto e PNFs</i>	64
6	AVALIAÇÃO DA ABORDAGEM REMINDER	65
6.1	Objetivo	65
6.2	Definição	65
6.3	Estudo de observação	67
6.3.1	<i>Descrição</i>	68
6.3.2	<i>Procedimentos</i>	68
6.3.3	<i>Resultados</i>	69
6.3.3.1	<i>Participante P1</i>	70
6.3.3.2	<i>Participante P2</i>	72
6.3.4	<i>Análise geral</i>	73
6.4	Considerações Finais	74
7	EXTENSÃO DA FERRAMENTA DYMMER E TESTE DE USABILIDADE	75
7.1	Visão geral da ferramenta	75
7.1.1	<i>Estrutura em camadas</i>	75
7.1.2	<i>Divisão de pacotes</i>	77
7.1.3	<i>Diagrama de Classes</i>	77
7.2	Requisitos funcionais	78
7.3	Representação das extensões propostas na ferramenta DyMMer	79

7.3.1	Modelo de features de qualidade	79
7.3.1.1	Adicionando característica de qualidade	79
7.3.1.2	Adicionando subcaracterística de qualidade	81
7.3.1.3	Adicionando propriedades não-funcionais	83
7.3.2	Definições de cenários de adaptações de contexto	86
7.3.2.1	Adicionando informação de contexto	86
7.3.2.2	Especificando variações de adaptação	87
7.3.2.3	Especificando restrições de integridade de contexto	88
7.3.3	Restrições de interdependências entre features e PNFs	89
7.3.4	Adição de múltiplos cenários de adaptações de contexto	91
7.3.5	Visualização da configuração do modelo de features	91
7.4	Teste de usabilidade	92
7.4.1	Usuários	93
7.4.2	Tarefas	93
7.4.3	Execução do teste	94
7.4.4	Resultados	94
7.4.4.1	Satisfação Geral	95
7.4.4.2	Utilidade da Ferramenta	96
7.4.4.3	Qualidade da Informação	96
7.4.4.4	Qualidade da Interface	97
7.5	Considerações Finais	97
8	CONSIDERAÇÕES FINAIS	99
8.1	Ameaças à Validade	99
8.2	Trabalhos Futuros	99
8.3	Contribuições	100
8.4	Conclusões	100
	REFERÊNCIAS	102
	APÊNDICE A FORMULÁRIO DE CONSENTIMENTO	106
	APÊNDICE B MODELAGEM DE LINHAS DE PRODUTOS DE SOFTWARE	
	DINÂMICAS - FORMULÁRIO DE BACKGROUND	108
	B.1 PARTE 1 - PERFIL DO ESPECIALISTA	108
	B.2 PARTE 2 - CONHECIMENTO TÉCNICO	109

APÊNDICE C	DESCRIÇÃO DAS TAREFAS	112
APÊNDICE D	DESCRIÇÃO DO MODELO EXEMPLO	114
APÊNDICE E	DESCRIÇÃO DO CENÁRIO DE USO	115
APÊNDICE F	REPRESENTAÇÃO DO MODELO DE <i>FEATURES</i> DE QUALIDADE	116
F.1	CATÁLOGO DE PROPRIEDADES NÃO-FUNCIONAIS	117
APÊNDICE G	DEFINIÇÕES DE CENÁRIOS DE ADAPTAÇÕES DE CONTEXTO	118
G.1	REGRAS DE CONTEXTO	118
APÊNDICE H	<i>TEMPLATE</i> DE IDENTIFICAÇÃO DE RESTRIÇÕES DE ATIVAÇÃO E DE DESATIVAÇÃO DE ACORDO COM CADA CENÁRIO DE ADAPTAÇÃO DE CONTEXTO	120
APÊNDICE I	<i>TEMPLATE</i> DE IDENTIFICAÇÃO DE RESTRIÇÕES DE INTERDEPENDÊNCIAS ENTRE <i>FEATURES</i> E PNFs EM UM DETERMINADO CENÁRIO DE ADAPTAÇÃO DE CONTEXTO	121
APÊNDICE J	FORMULÁRIO DE AVALIAÇÃO DA ABORDAGEM ReMINDER	122
APÊNDICE K	DESCRIÇÃO DAS TAREFAS - TESTE DE USABILIDADE	125
ANEXO A	CARACTERÍSTICAS E SUBCARACTERÍSTICAS ISO/IEC 25010 (SQUARE)	129
ANEXO B	<i>THE POST-STUDY SYSTEM USABILITY QUESTIONNAIRE</i> (PSSUQ)	132

1 INTRODUÇÃO

A necessidade de desenvolver produtos de software com alto nível de qualidade fez com que houvesse um aumento da complexidade no desenvolvimento destes produtos. Qualidade e complexidade, associadas com baixo custo de desenvolvimento, tempo de entrega do produto e produtividade são elementos fundamentais para a melhoria da competitividade entre empresas de software e diferenciar entre o sucesso do fracasso na indústria de software (PACHECO et al., 2015). Neste sentido, o reuso de software tem sido amplamente reconhecido como uma forma eficaz de melhorar a qualidade e produtividade do software.

O paradigma de Linhas de Produto de Software (LPS) tem emergido como meio para o reuso, quando o reuso é planejado, implementado e gerenciado (NORTHROP, 2002). Uma LPS tem como objetivo identificar *features* comuns e variáveis entre os possíveis produtos gerados a partir de uma linha de produtos. Uma LPS representa um conjunto de sistemas de software que compartilham de um conjunto de *features* comuns e gerenciadas, que satisfazem as necessidades de um segmento de mercado particular ou missão e são desenvolvidas a partir de um conjunto comum de artefatos de uma maneira predeterminada (CLEMENTS; NORTHROP, 2002).

Pohl, Böckle e Linden (2005) verificam que o uso de LPSs traz, dentre outros benefícios, o aumento na qualidade dos produtos de software e uma redução no tempo de desenvolvimento de novos produtos. Assim, LPSs têm se consolidado como um paradigma de desenvolvimento de software que promove o reuso de software de maneira sistematizada (CLEMENTS; NORTHROP, 2002).

Com o advento do uso de LPSs, surgiram métodos, técnicas e ferramentas para apoiar as principais atividades das LPSs, que são a representação da variabilidade e o gerenciamento de *features*, comuns e variáveis, além da promoção do reuso em larga escala. Entre as diversas técnicas utilizadas para tal fim, o modelo de *features* representa um dos artefatos mais utilizados para representação e tratamento da variabilidade em LPSs (ALVES et al., 2010).

O modelo de *features* é uma representação gráfica de similaridades e variabilidades, entre *features* de uma LPS (KANG et al., 1990). O modelo de *features* estrutura hierarquicamente um conjunto de *features* de um sistema. Uma *feature* pode ser decomposta em várias *sub-features* (POHL; BÖCKLE; LINDEN, 2005). As *features* expressam similaridades e variabilidades de produtos de um domínio, podendo significar algo diferente para diferentes linhas de produto e domínio, tal como ser um requisito funcional (RF), uma funcionalidade ou um aspecto de

qualidade (BEUCHE; DALGARNO, 2007)

Hallsteinsen et al. (2008) separam as *features* de uma LPS em três categorias: i) a primeira agrupa o conjunto de *features* comuns a todos os produtos de uma LPS; ii) a segunda, as *features* presentes em uma LPS, mas não em todos os produtos; iii) e a última representam as *features* individuais de um produto, atendendo as suas particularidades.

No entanto, a representação de variabilidade dos produtos gerados por LPSs é feita apenas de forma estática, ou seja, a adaptação ocorre em tempo de desenvolvimento, sendo necessário a geração de diferentes configurações do produto conforme variam a necessidade do cliente e as características do seu ambiente de execução (FERNANDES; WERNER; MURTA, 2008). Contudo, com o surgimento de sistemas autônomicos e de sistemas sensíveis ao contexto, o desenvolvimento de sistemas tem se tornado mais complexo que o comum, dado as mudanças de requisitos e restrições conforme o ambiente a qual está inserido, exigindo um alto grau de adaptação. Estes sistemas podem ser conceitualizados como uma Linha de Produto de Software Dinâmicas (LPSD) em que a ativação/desativação de funcionalidades e de configuração ocorrem em tempo de execução (HALLSTEINSEN et al., 2008).

Bencomo, Hallsteinsen e Almeida (2012) definem uma LPSD como uma abordagem que estende o conceito convencional de LPSs permitindo a geração de variantes do software em tempo de execução. LPSDs e LPSs tradicionais diferenciam-se pelo modo como cada uma trata do gerenciamento de variabilidade. Variabilidade pode ser definida como a capacidade de um sistema ou artefato de software possui de ser alterado, customizado ou configurado para um contexto em particular (BOSCH, 2004).

O gerenciamento de variabilidade em uma LPS ocorre de forma estática, por meio do mapeamento dos requisitos do novo produto a partir dos ativos disponíveis, seleção dos ativos que satisfazem aos requisitos, e por fim a composição do produto através da junção desses ativos (POHL; BÖCKLE; LINDEN, 2005). Entretanto, em LPSD o gerenciamento de variabilidade ocorre de forma dinâmica, em que o produto pode adaptar-se em tempo de execução as mudanças de comportamento do ambiente em execução, resultando na ativação ou desativação das funcionalidades do produto (HINCHEY; PARK; SCHMID, 2012).

A notação para a representação de variabilidade pode ser gráfica, textual ou combinação das duas formas (MASSEN; LICHTER, 2002). Em LPSs e LPSDs, a variabilidade de um produto pode ser representada por meio do modelo de *features*. O modelo de *features* possui duas variações, modelo de *features* de LPS e modelo de *features* de LPSD. O modelo de

feature de LPS não pode modificar as características do sistemas, ou adaptar-se automaticamente, em tempo de execução; já no modelo de *features* de LPSD, as características do sistema podem ser adicionadas, removidas e modificadas em tempo de execução de maneira gerenciada (BENCOMO; HALLSTEINSEN; ALMEIDA, 2012).

Para suportar a modelagem do modelo de *features* de LPSDs, foi desenvolvida a ferramenta DyMMer (BEZERRA et al., 2016). A DyMMer representa as *features* e adaptações de contexto em tempo de projeto em um modelo de *features* único, permitindo a edição dos modelos de *features* para a inserção de regras de adaptação de contexto para ativação e desativação de *features*. Além disso, a ferramenta suporta a avaliação da qualidade estrutural do modelo de *features* em LPSs e LPSDs e representa o modelo de *features* utilizando a notação do método FODA com cardinalidade proposta por Czarnecki, Helsen e Eisenecker (2005b). No entanto, esta notação não suporta a modelagem de propriedades não-funcionais (PNFs) em modelo de *features* de LPS e modelo de *features* de LPSD (CZARNECKI; HELSEN; EISENECKER, 2005b).

Hammani (2014) verifica que apesar da importância de PNFs em linhas de produtos, geralmente as abordagens de modelagem se concentram mais em RFs ao invés de PNFs, tornando a modelagem de PNFs uma deficiência na Engenharia de Linhas de Produtos de Software (ELPS).

Fundamentado na importância de PNFs em linhas de produtos, e verificando a existência da deficiência no processo de ELPS, este trabalho tem como objetivo geral definir uma abordagem e estender o modelo de *features* para LPSDs, por meio da representação de PNFs, restrições e adaptações de contexto em tempo de projeto. Com a execução deste trabalho espera-se alcançar quatro objetivos específicos, que são: (i) identificar formas de representação de PNFs e múltiplos cenários de adaptações de contexto no modelo de *features* de LPSD em tempo de projeto; (ii) identificar regras de contexto, restrições entre *features* e adaptações de contexto; (iii) identificar restrições entre *features* e PNFs em múltiplos cenários de adaptação; (iv) levantar e representar as restrições entre *features* e adaptações de contexto; e (v) estender a ferramenta DyMMer para suportar a operacionalização da abordagem proposta. Como público-alvo deste trabalho, identifica-se engenheiros de domínio, pesquisadores acadêmicos, e quaisquer profissionais que trabalhem com o modelo de *features* de LPSDs.

Este trabalho está organizado em outras sete Seções, além desta Seção de introdução. A Seção 2 apresenta os os conceitos que fundamentam o desenvolvimento deste trabalho. A Seção 3 apresenta os trabalhos relacionados que levam em consideração abordagens existentes

que modelam RNFs. Na Seção 4 são apresentados os passos executados para a realização deste trabalho. A Seção 5 apresenta a abordagem proposta para modelagem de PNFs e adaptações de contexto em modelo de *features* de LPSDs, denominada ReMINDER (An AppRoach to Modeling Non-FunctIoNal Properties in Dynamic SoftwarE PRoduct Lines). A Seção 6 apresenta a avaliação realizada para validação da abordagem. A Seção 7 apresenta uma visão geral das extensões adicionadas a ferramenta DyMMer. Também apresenta o teste de usabilidade realizado na ferramenta. Finalmente, a Seção 8 apresenta algumas conclusões, incluindo as contribuições e as limitações deste trabalho, além das possibilidades de trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Nesta Seção serão apresentados os conceitos que fundamentam o desenvolvimento deste trabalho. Na Seção 2.1 são introduzidos os conceitos referentes à Linha de Produto de Software Dinâmicas (LPSDs), compreendendo seus respectivos componentes, o escopo de atuação e suas particularidades quando comparadas com Linhas de Produto de Software tradicionais (LPSs). Na Seção 2.2, os conceitos referentes a Requisitos de LPSD são abordados; na Subseção 2.2.1 são apresentados os conceitos de requisitos não-funcionais, tratados aqui como propriedades não-funcionais. Na Seção 2.3 são abordados os conceitos de modelo de *features*, no que corresponde a suas características, restrições e notações em LPS e LPSD. Na última Seção 2.4 é apresentada a ferramenta DyMMer, sua arquitetura e funcionalidades.

2.1 Linha de Produto de Software Dinâmicas

Linhas de Produto de Software (LPS) representam um conjunto de sistemas de software que compartilham de um conjunto de *features* comuns e gerenciadas, que satisfazem as necessidades de um segmento de mercado particular ou missão. As LPSs são desenvolvidas a partir de um conjunto comum de artefatos de uma maneira pré-determinada. As LPSs têm se consolidado como um paradigma de desenvolvimento de software que promove o reuso de software de maneira sistematizada (CLEMENTS; NORTHROP, 2002).

A necessidade de desenvolver sistemas capazes de se autoadaptar dinamicamente a variações de ambiente e aos requisitos de usuários, exigindo um alto grau de adaptação, torna o desenvolvimento de tais sistemas uma atividade complexa. A fim de lidar com o surgimento e os desafios desses sistemas, o campo de Linhas de Produto de Software Dinâmicas (LPSDs) tem sido investigado (HALLSTEINSEN et al., 2008).

Bencomo, Hallsteinsen e Almeida (2012) definem uma LPSD como uma abordagem que estende das LPSs convencionais, permitindo a geração de variantes do software em tempo de execução. Estas variantes podem ser definidas como a capacidade de um sistema ou artefato de software ser alterado, customizado ou configurado para um contexto em particular (BOSCH, 2004). Segundo Capilla e Bosch (2011), a maioria dos modelos de gerenciamento de variabilidade lidam somente com a variabilidade em LPS, capturando os pontos de variação, assim como as regras de restrições e dependências entre as variações em tempo de desenvolvimento.

Em LPS convencionais, o gerenciamento de variabilidade ocorre em tempo de

desenvolvimento, através do mapeamento dos requisitos do novo produto a partir dos ativos disponíveis, seleção dos ativos que satisfazem aos requisitos, e por fim, a composição do produto através da junção desses ativos (POHL; BÖCKLE; LINDEN, 2005). Entretanto, em LPSDs, o gerenciamento de variabilidade ocorre de forma dinâmica, em que o produto pode adaptar-se em tempo de execução as mudanças de comportamento do ambiente e resultando na ativação ou desativação das funcionalidades do produto (HINCHEY; PARK; SCHMID, 2012).

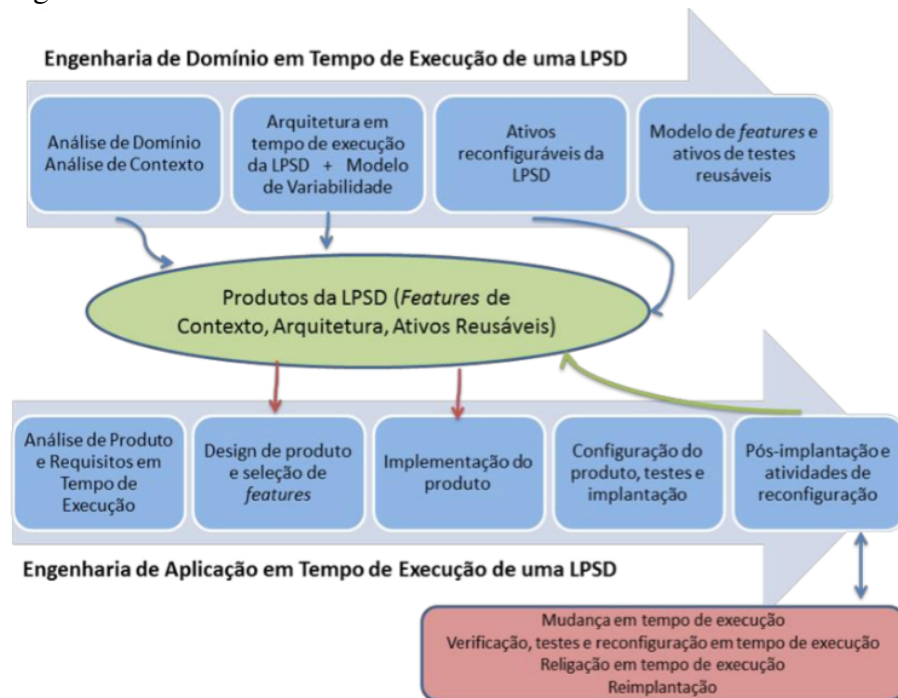
Segundo Bencomo et al. (2008) um sistema dinamicamente adaptativo opera em ambientes que impõem mudanças de contexto e essas mudanças podem ocorrer por dois tipos de variabilidades: i) ambiente ou variabilidade de contexto, onde a evolução do ambiente não pode ser prevista em tempo de projeto e ii) variabilidade estrutural que abrange a variabilidade dos componentes ou *features* e de suas configurações. A fim de satisfazer os requisitos do novo contexto, o sistema pode adicionar novos componentes ou organizar a atual configuração (BENCOMO et al., 2008).

Hallsteinsen et al. (2008) enumeram cinco principais propriedades presentes em uma LPSD: i) Variabilidade dinâmica, configuração e vinculação em tempo de execução; ii) Adição de *features* variáveis em tempo de execução; iii) Adaptar-se diversas vezes durante todo seu ciclo de vida; iv) Tratar-se com mudanças inesperadas; e v) Tratar-se com mudanças das necessidades de usuário, tais como requisitos funcionais ou não-funcionais. Além disso, os autores mencionam algumas propriedades opcionais, ou seja, que não estão presentes em todas as LPSD: i) Sensível ao contexto; ii) Propriedades autonômicas ou auto-adaptativas; e iii) Tomada de decisão automática.

Dadas estas propriedades, LPSD podem beneficiar diversas áreas de pesquisas, por exemplo, o desenvolvimento de sistemas autonômicos e sistemas sensíveis ao contexto, que exigem um alto grau de adaptação, dado as mudanças de requisitos e restrições conforme o ambiente no qual está inserido. Hallsteinsen et al. (2008) conceitualizam estes sistemas como LPSD em que a ativação/desativação de funcionalidades e de configurações ocorrem em tempo de execução.

Capilla et al. (2014) apresentam um processo de ciclo de desenvolvimento de uma LPSD, ilustrado na Figura 1. O processo reúne um conjunto de propriedades que caracterizam uma LPSD, que são: gerenciamento e suporte a variabilidade em tempo de execução, conexões dinâmicas e múltiplas, autoadaptação e sensibilidade ao contexto para comportamento autonômico.

Figura 1 – Ciclo de vida de atividades de uma LPSD.



Fonte – Adaptada de Capilla et al. (2014).

Na fase de Engenharia de Domínio em tempo de execução, Capilla et al. (2014) estende a atividade de análise de domínio com a inclusão da atividade de análise de contexto, responsável por identificar as *features* de contexto que são relevantes para os produtos de LPSDs. Dessa forma, quando um produto altera seu comportamento de acordo com o contexto de uso, é possível identificar quais *features* devem ser adicionadas à nova configuração do produto. A ativação e desativação de *features* são exemplos de propriedades que podem ser identificadas e definidas no modelo de variabilidade. A arquitetura em tempo de execução inclui todos os mecanismos capazes de lidar com as mudanças dinâmicas de LPSDs, tais como, ativação e desativação de *features* e as possíveis modificações na variabilidade estrutural em tempo de execução. Essa arquitetura é responsável por liderar a implementação dos ativos reconfiguráveis. Por fim, todos esses artefatos são adicionados ao repositório da LPSD para a fase subsequente (CAPILLA et al., 2014).

A análise de domínio é um processo de identificação, elicitação e modelagem de requisitos de uma família de produtos (NOORIAN; BAGHERI; DU, 2012). As principais tarefas da análise de domínio podem ser categorizadas em: identificação e elicitação de propriedades funcionais e não-funcionais; modelagem, que tem como objetivo modelar e representar propriedades não-funcionais de forma estruturada; integração de modelos funcionais e não-funcionais, auxiliando na compreensão e identificação dos impactos mútuos entre eles; e

avaliação do modelo, em que o modelo precisa ser avaliado em termos de alguns atributos de qualidades (NOORIAN; BAGHERI; DU, 2012).

Na fase de Engenharia de Aplicação em tempo de execução, os requisitos estáticos e em tempo de execução, conduzem ao desenvolvimento dos produtos de uma LPSD. A arquitetura em tempo de execução é personalizada juntamente com o modelo de *features* para produzir os produtos da LPSD, que exibem as características reconfiguráveis necessárias. Em seguida, os produtos são testados, configurados e implantados, e se uma nova configuração é necessária em tempo de execução, o ciclo de vida de LPSD suporta as tarefas de pós-implantação e de reconfiguração para lidar com as novas mudanças (CAPILLA et al., 2014).

2.2 Requisitos de Linha de Produto de Software Dinâmica

Os requisitos são descrições do que o sistema deve implementar, como ele deve se comportar, as restrições que o sistema ou o seu desenvolvimento deve satisfazer, as propriedades e atributos de qualidade que o sistema deve possuir (SOMMERVILLE; KOTONYA, 1998). Requisitos podem ser categorizados em funcionais (RFs) e não-funcionais (RNFs). RFs correspondem a funcionalidades que devem ser realizadas pelo sistema. Os RNFs, também chamados de requisitos extra-funcionais ou características de qualidade, referem-se a restrições sobre as funcionalidades do sistema, tais como, desempenho, segurança, confiabilidade, dentre outros (HAMMANI, 2014).

Em LPSDs a modelagem de RFs e RNFs representam um importante artefato gerado na Engenharia de Domínio. Estes podem ser representados em LPSDs por meio de *features*, uma vez que *features* correspondem a diferentes níveis de abstração: RFs e RNFs, ambiente e restrições de contexto, componentes de tempo de execução, módulos de implementação, entre outros (SANCHEZ et al., 2015).

2.2.1 Requisitos Não-Funcionais

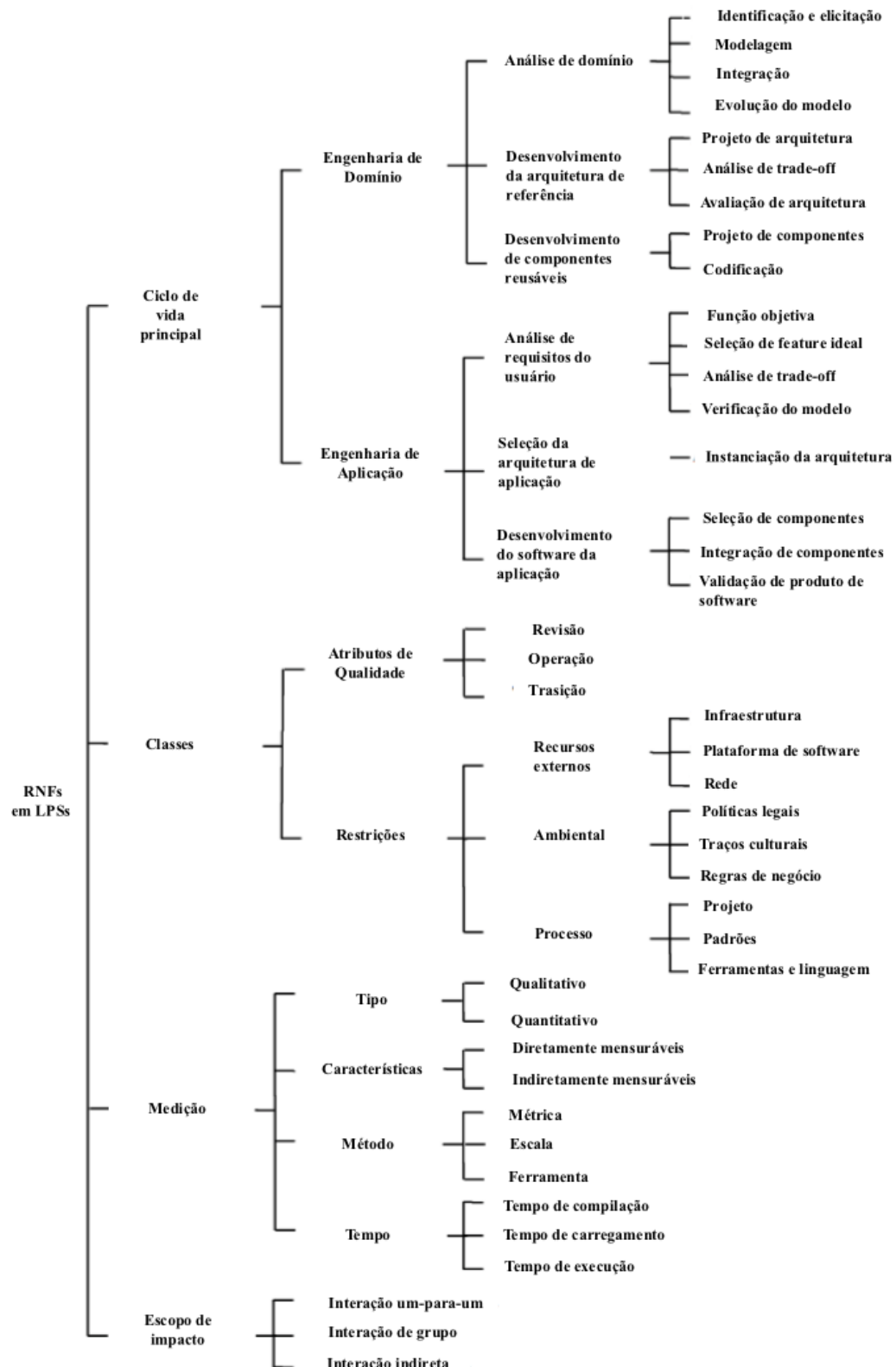
A complexidade de um sistema é determinada tanto por suas funcionalidades quanto por suas características não-funcionais que impactam na qualidade do sistema, como *performance*, usabilidade e modificabilidade (CHUNG; LEITE, 2009). RNFs ou propriedades não-funcionais, também são conhecidos como atributos de qualidade (BOEHM; BROWN; KASPAR, 1978).

Na Engenharia de Linha de Produtos de Software (ELPS) existem diversos tipos de propriedades de RNFs que são classificados de diferentes formas. Etxeberria e Sagardui (2005) classificam em dois tipos diferentes os atributos de qualidade em LPSs: i) atributos de qualidade de LPSs, considerados atributos de desenvolvimento ou não observáveis em tempo de execução; e ii) atributos de qualidade de domínio, considerados operacionais e observáveis através da execução (ETXEVERRIA; SAGARDUI; BELATEGI, 2008). Por exemplo, atributos de qualidade não observáveis em tempo de execução são: portabilidade, reusabilidade, integridade, dentre outros, ao contrário de *performance*, segurança, disponibilidade e usabilidade, que são observáveis em tempo de execução e operacionais (CLEMENTS, 2002).

No âmbito de LPSs, Noorian, Bagheri e Du (2012) apresentam uma taxonomia para a caracterização e a classificação dos principais aspectos de RNFs, esta taxonomia também pode ser aplicada a LPSDs. Esta taxonomia é dividida em quatro dimensões (ver Figura 2):

- **Ciclo de vida principal** – refere-se ao impacto dos RNFs nos dois principais ciclos de vida no desenvolvimento de linhas de produtos, a Engenharia de Domínio e Engenharia de Aplicação. Esta dimensão auxilia no entendimento e categorização dos passos ou tarefas que precisam ser realizados para desenvolver uma linha de produtos ciente de qualidade.
- **Classes** – investiga as diferentes classes de RNFs, atributos de qualidade e restrições, que tem sido identificadas por engenheiros de software, além disso esta dimensão auxilia na identificação de RNFs que foram aplicados no desenvolvimento de linhas de produtos.
- **Medição** – divide os RNFs em relação a características de medição, identificando tipos, características, métodos e tempo de medição que são exigidos para a medição de RNFs.
- **Escopo de impacto** – busca identificar os diversos tipos de interações existentes entre um RF ou RNF sobre outro RF ou RNF, levando em consideração o escopo de impacto de acordo com as suas interações (NOORIAN; BAGHERI; DU, 2012).

Figura 2 – Taxonomia para RNFs em linhas de produtos de software.



Fonte – Adaptada de Noorian, Bagheri e Du (2012).

Segundo Noorian, Bagheri e Du (2012), em relação às características de medição. Os RNFs podem ser classificados em: tipo, características, métodos e tempo. Com relação ao

tipo de medição, elas podem ser quantitativas e qualitativas. A quantitativa, preocupa-se com medições numéricas, com valores contínuos para os RNFs, enquanto que medidas qualitativas se preocupam com medições não-numéricas, por exemplo, o nível de qualidade sonora, pode ser medido como, baixo, médio ou alto. As características de medição podem ser classificadas como: diretamente mensuráveis e indiretamente mensuráveis. As diretamente mensuráveis preocupam-se com a medição de atributos base, ou seja, que não necessitam ser divididos em outros atributos, enquanto que as características indiretamente mensuráveis dependem de outros atributos base e necessitam ser divididos em vários atributos mensuráveis. A classificação por métodos, referem-se a métricas, escalas e ferramentas que suportem a medição de RNFs. A classificação por tempo de medição, é categorizada em tempo de compilação, tempo de carga e tempo de execução, por exemplo, o RNF de segurança deve ser medido em tempo de execução.

A norma SQuaRE 25010 da ISO/IEC (2011), define um modelo de qualidade em uso e um modelo de qualidade do produto. Estes modelos de qualidade são compostos por características, subcaracterísticas e atributos de qualidade. Características de qualidade são um conjunto de propriedades de um produto de software pela qual sua qualidade pode ser definida e avaliada (ISO/IEC, 2001). Uma característica pode ser dividida em várias subcaracterísticas. Atributos são propriedades que podem ser mensuráveis, abstratas ou físicas.

O modelo de qualidade em uso é dividido em cinco categorias: Eficácia, Eficiência, Satisfação, Segurança e Usabilidade; enquanto que o modelo de qualidade do produto é dividido em oito categorias: Adequação funcional, Eficiência de desempenho, Compatibilidade, Usabilidade, Confiabilidade, Segurança, Manutenibilidade e Portabilidade. Cada uma destas características é composta por um conjunto de subcaracterísticas relacionadas (ISO/IEC, 2011). Cada característica e subcaracterística do modelo de qualidade do produto pode ser observada no Quadro 1.

Quadro 1 – Qualidade do produto características e subcaracterísticas

Características	Descrição	Subcaracterísticas
Adequação funcional	Grau em que um produto ou sistema fornece funções que correspondam às necessidades explícitas e implícitas quando utilizado sob condições especificadas	Completude funcional Corretude funcional Adequação funcional
Eficiência de desempenho	O desempenho em relação à quantidade dos recursos utilizados sob condições estabelecidas	Comportamento de tempo Utilização de recurso Capacidade
Compatibilidade	Grau em que um produto, sistema ou componente pode trocar informações com outros produtos, sistemas ou componentes, e/ou realizar suas funções necessárias, ao compartilhar o mesmo ambiente de hardware ou software	Interoperabilidade Coexistência
Usabilidade	Grau em que um produto ou sistema pode ser usado por usuários específicos para alcançar objetivos específicos com eficácia, eficiência e satisfação em um contexto de uso especificado	Adequação de identificabilidade Aprendibilidade Operabilidade Proteção de erro do usuário Estética da interface de usuário Acessibilidade
Confiabilidade	Grau em que um sistema, produto ou componente executa funções específicas sob condições especificadas por um período de tempo especificado	Maturidade Disponibilidade Tolerância a falhas Recuperabilidade
Segurança	Grau ao qual um produto ou sistema protege informações e dados, de modo que as pessoas ou outros produtos ou sistemas têm o grau de acesso de dados apropriado para os seus tipos e níveis de autorização	Confidencialidade Integridade Não-repúdio Prestação de contas Autenticidade
Manutenibilidade	Grau de eficácia e eficiência com que um produto ou sistema pode ser modificado por a equipa de manutenção destinados	Modularidade Reusabilidade Analisabilidade Modificabilidade Testabilidade
Portabilidade	Grau de eficácia e eficiência com que um sistema, o produto ou componente pode ser transferido de um hardware, software ou outro ambiente operacional ou para outro uso.	Adaptabilidade Instabilidade Substituibilidade

Fonte – Adaptado de ISO/IEC (2011).

Mari e Eila (2003) classificam os atributos de qualidade nos tipos, de execução e de evolução. Os atributos de qualidade em execução, podem ser observáveis em tempo de execução, ou seja, de acordo com o comportamento do sistema. Por exemplo, desempenho, segurança, disponibilidade, usabilidade, escalabilidade, confiabilidade, interoperabilidade e adaptabilidade. Ao contrário dos atributos de qualidade de evolução, que são observáveis durante o ciclo de vida do software, caracterizando as diferentes fases de vida de um processo de desenvolvimento e manutenção de um sistema. Por exemplo, manutenibilidade, flexibilidade, modificabilidade, extensibilidade, portabilidade, reusabilidade, integridade e testabilidade.

Soares et al. (2014) identifica por meio de uma revisão sistemática da literatura, 52 propriedades não-funcionais (PNFs) que são comumente tratadas em tempo de execução (ver Quadro 2). A revisão sistemática foi concentrada em sete domínios de aplicação: automotivo, sistemas de gerenciamento de banco de dados, baseados na web, embarcados, móveis, sistemas operacionais e domínio específico dos usuários finais.

Quadro 2 – PNFs que emergem em tempo de execução

1. Precisão	14. Tempo de resposta da expedição	27. Consumo de memória	40. Escalabilidade
2. Disponibilidade	15. Facilidade de uso	28. Tamanho da mensagem	41. <i>Security</i>
3. CPU disponíveis	16. Esforço	29. Tempo mínimo de espera	42. Tempo de serviço
4. Capacidade de canal/latência	17. Consumo de energia	30. Desempenho	43. Gravidade de falhas do produto
5. Conforto	18. Eficiência energética	31. Frequência de imagem	44. Espaço - memória principal
6. Desempenho da comunicação	19. Tempo de execução	32. Precisão da posição	45. Espaço - memória secundária
7. Completude	20. Tolerância a falhas	33. Privacidade	46. Velocidade
8. Confidencialidade	21. Tamanho do arquivo	34. <i>Safety</i>	47. Time - taxa de transferência
9. Uso conveniente do sistema	22. <i>Footprint</i>	35. Tempo de reconhecimento	48. Usabilidade
10. Consumo de CPU	23. Interface de usuário amigável	36. Confiabilidade	49. Usabilidade para residentes APT
11. Satisfação do cliente	24. Integridade	37. Tempo de resposta	50. Usabilidade para VIPs
12. Ciclo	25. Usabilidade para deficientes	38. <i>Safety</i>	51. Peso
13. Segurança de acesso a dados	26. Latência	39. Economia de energia	52. Largura de banda WLAN

Fonte – Adaptado de Soares et al. (2014).

Dado este conjunto de PNFs que são tratadas em tempo de execução, por domínios de aplicações que sofrem adaptações em tempo de execução, tais propriedades também podem emergir em sistemas autônômicos e sistemas sensíveis ao contexto, que podem ser conceitualizados como LPSDs.

2.3 Modelo de *Features*

O modelo de *features* é um dos mais importantes artefatos de uma LPS. Originalmente o modelo de *features* foi proposto por (KANG et al., 1990) como parte do seu método FODA (do inglês, *Feature-Oriented Domain Analysis*). O modelo de *features* é uma representação gráfica de similaridades e variabilidades entre as *features* de uma Linha de Produtos de Software (KANG et al., 1990). As *features* expressam similaridades e variabilidades de produtos de um domínio, podendo significar algo diferente para diferentes linhas de produto e domínio, tal como ser um requisito, uma funcionalidade ou um aspecto de qualidade (BEUCHE; DALGARNO, 2007). O modelo de *features* estrutura hierarquicamente um conjunto de *features* de um sistema. Uma *feature* pode ser decomposta em várias *subfeatures* (POHL; BÖCKLE; LINDEN, 2005).

Um modelo de *feature* é uma representação de todas as configurações possível de um produto. Este modelo é visualmente representado com *features* e os relacionamentos entre elas (KANG et al., 1990). Segundo Benavides, Segura e Ruiz-Cortés (2010) são um tipo especial de modelo de informação amplamente utilizado em engenharia de linhas de produto de software. O modelo de *features* é representado como um conjunto de *features* hierarquizadas compostas por:

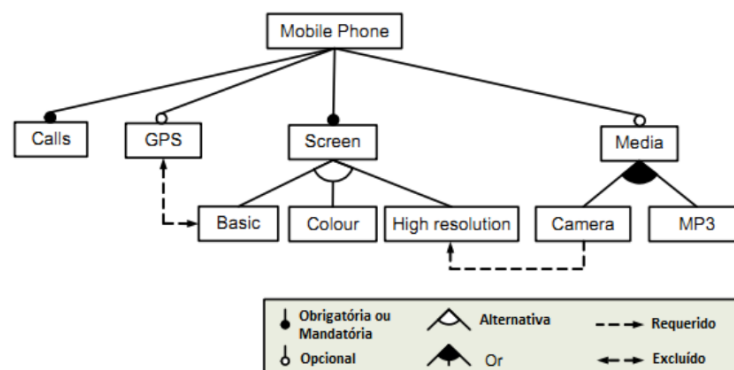
- Relacionamento entre uma *feature* pai (ou compostos) e suas *features* filhas (ou *subfeatures*);
- Restrições *Cross-tree* que são tipicamente declaradas como inclusão ou exclusão

de forma que: se uma *feature* F é incluída, então *features* A e B podem ser incluídas (ou excluídas).

A Figura 3 ilustra um modelo de *features* básicos. O modelo de *features* básico permite as seguintes relações entre as *features* (BENAVIDES; SEGURA; RUIZ-CORTÉS, 2010):

- **Obrigatórias ou mandatórias** – A *feature* filha tem um relacionamento mandatório com sua *feature* pai quando a *feature* filha é incluída em todos os produtos em que a sua *feature* pai aparece. Por exemplo, no modelo de *features* da Figura 3 todo o sistema *Mobile phone* deve fornecer suporte a *feature* *Call*;
- **Opcionais** – Uma *feature* filha tem um relacionamento opcional com seu pai quando o filho pode ser opcionalmente incluído em todos os produtos em que a sua *feature* pai aparece. Por exemplo, no modelo de *features* da Figura 3 todo o sistema *Mobile phone* pode opcionalmente incluir suporte para a *feature* *GPS*;
- **Alternativas** – Um conjunto de *features* filhas apresenta uma relacionamento alternativo com seu pai quando apenas uma de suas *features* filhas podem ser selecionadas quando a sua *feature* pai é parte do produto. Como por exemplo, no modelo de *features* da Figura 3 *Mobile phones* pode incluir suporte para as *features* *basic*, *colour* ou *high resolution screen*, mas apenas para uma destas *features*; e
- **Or** – Um conjunto de *features* filhas tem um relacionamento "OR" com seus pais quando um ou mais entre eles podem ser incluídos nos produtos em que a sua *feature* pai aparece. Como por exemplo, no modelo de *features* da Figura 3 sempre a *feature* *Media* é selecionada, as *features* *Camera*, *MP3* ou ambos podem ser selecionadas.

Figura 3 – Modelo de *features* básico.



Fonte – Adaptada de Benavides, Segura e Ruiz-Cortés (2010).

Benavides, Segura e Ruiz-Cortés (2010) determinam que uma *feature* filha somente pode aparecer em um produto se a sua *feature* pai faz parte do produto. A *feature* raiz é uma parte de todos os produtos dentro da linha de produtos de software. Além das relações de parentesco entre as *features*, o modelo de *features* pode também conter restrições *cross-tree* entre as suas *features*. Estas restrições são tipicamente na forma:

- **Requerido** – Se uma *feature* A requer uma *feature* B, a inclusão de A em um produto implica a inclusão de B. Exemplo da Figura 3, em que *Mobile Phones* incluem uma *feature* *Camera* para poder incluir suporte para a *Feature High Resolution Screen*;
- **Excluído** – Se uma *feature* A exclui uma *feature* B, ambas as *features* não podem ser parte do mesmo produto. Exemplo da Figura 3: *GPS* e *Basic Screen* são *features* incompatíveis.

Desde a introdução do modelo de *features* proposto por Kang et al. (1990), diversas notações foram desenvolvidas e novas características ao modelo de *features* foram adicionadas, como o conceito de cardinalidade para representar *features* proposto por Czarnecki, Helsen e Eisenecker (2005b). A cardinalidade é utilizada para definir o número mínimo e máximo de *features* que podem ser escolhidas a partir de um conjunto de alternativas de um ponto de variação (FERNANDES, 2009).

Essa nova relação introduzida no modelo de *features* são definidas por Benavides, Segura e Ruiz-Cortés (2010) como:

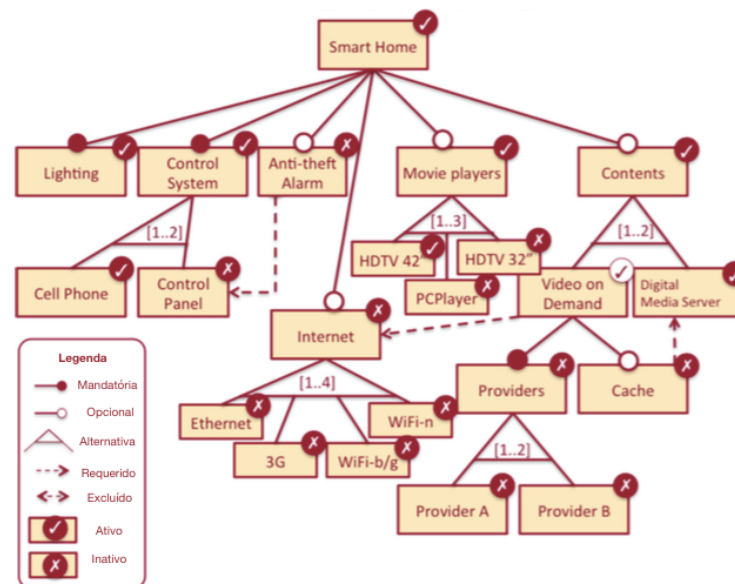
- **Cardinalidade de Feature** – é uma sequência de intervalos indicados $[n..m]$ com n como limite inferior e m como limite superior. Estes intervalos determinam o número de instâncias da *feature* que pode ser parte de um produto. Esse relacionamento pode ser usado como uma generalização do obrigatório original $[1,1]$ e opcionais $[0,1]$ relações definidas na notação FODA; e
- **Grupo de Cardinalidade** – é um intervalo denotado por $\langle n..m \rangle$, onde n representa o limite inferior e m o limite superior limitando o número de *features* filhas que podem ser parte de um produto quando a *feature* pai é selecionada. Assim, um relacionamento alternativo é o equivalente a $\langle 1..1 \rangle$ e um Or-relacionamento é equivalente a $\langle 1..n \rangle$, sendo n o número de *features* no relacionamento.

O modelo de *features* apresenta variações conforme se tratar de LPS tradicional

ou LPS dinâmica. O modelo de *feature* de LPS, não podem modificar as características do sistemas, ou adaptar-se automaticamente, uma vez que as variações são criadas durante o ciclo de desenvolvimento, enquanto no modelo de *features* de LPSD, as características do sistema podem ser adicionadas, removidas e modificadas em tempo de execução de maneira gerenciada (BENCOMO; HALLSTEINSEN; ALMEIDA, 2012).

Restrições de inclusão e exclusão, de cardinalidade de *features* e de grupos de cardinalidade representam uma importante fonte de informação para o modelo de *features* de LPSD. Por exemplo, restrições de inclusão e exclusão fornecem informações para a ativação/desativação de *features* em tempo de execução. Dessa forma um conjunto de restrições definem o conjunto de *features* a ser adicionada ou excluída na configuração atual do produto, conforme ilustrado na Figura 4.

Figura 4 – Modelo de *features* dinâmico em termos de ativação e desativação de *features*.



Fonte – Adaptada de Capilla et al. (2014).

A Figura 4 exemplifica um modelo de *features* de LPSD de um sistema de casa inteligente, ilustrando o estado atual de cada *feature* em termos de ativação e desativação. Neste exemplo, as *features* *Lighting* e *Control System* podem ser ativadas através do *Cell Phone*. Percebe-se também que, caso o usuário final deseje utilizar o recurso de *Video on-demand*, a configuração atual não é válida até que a *feature* *Video on-demand* ative a conexão com a internet (atualmente desativada) (CAPILLA et al., 2014).

A *feature* internet possui quatro reconfigurações possíveis uma para cada tipo de

conexão com a internet: *Ethernet*, *3G*, *Wi-Fi-b/g* e *Wi-Fi/n*. Estas devem mudar seu estado para alcançar uma configuração válida do produto em tempo de execução. Segundo Capilla et al. (2014) um dos critérios para escolher a melhor configuração pode ser baseado em termos de atributos de qualidade, ou RNFs exigidos pelos *stakeholders* ou do sistema, por exemplo, a taxa de largura de banda disponível.

2.3.1 Notações do Modelo de Features com RNFs

A modelagem de *features* é a atividade de identificação de aspectos visíveis e externos aos produtos de uma linha e a sua organização em um modelo de *features* (LEE; MUTHIG, 2006). As principais tarefas com base na qualidade podem ser categorizadas em: identificação e elicitación de propriedades funcionais e não-funcionais; modelagem, que tem como objetivo modelar e representar propriedades não-funcionais de forma estruturada; integração de modelos funcionais e não-funcionais, auxiliando na compreensão e identificação dos impactos mútuos entre eles; e avaliação do modelo, em que o modelo precisa ser avaliado em termos de alguns atributos de qualidades (NOORIAN; BAGHERI; DU, 2012).

O modelo de *features* tem sido extensivamente estudado para representação de RNFs. Recentemente, extensões para o modelo de *features* tem sido propostas, por exemplo, *Extended Feature Model* (Ext-FM) (BENAVIDES; TRINIDAD; RUIZ-CORTÉS, 2005) e *Integrated Software Product Line Model* (ISPLM) (SIEGMUND et al., 2008) para a inclusão de RNFs em LPS, dentre estas foi identificado *Feature Models and Quality Criteria* (FM-QC) (SANCHEZ et al., 2015) para LPSDs. Nesta Seção, serão apresentadas algumas notações utilizadas para a modelagem do modelo de *features*.

2.3.1.1 Extended Feature Model (Ext-FM)

Benavides, Trinidad e Ruiz-Cortés (2005), apresentam extensões no modelo de *features* proposto por Czarnecki, Helsen e Eisenecker (2005b), introduzindo os conceitos de:

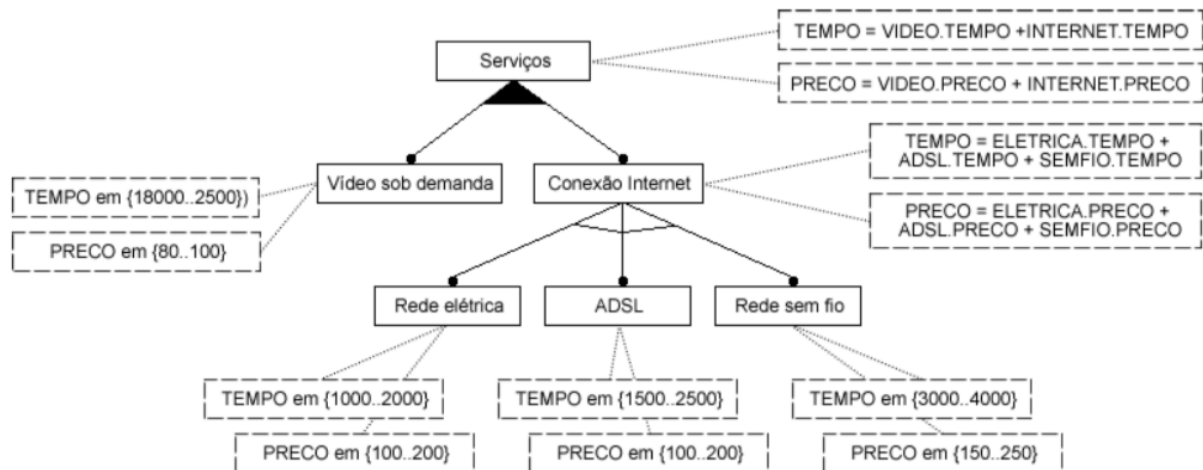
- **Atributos** – propriedades de uma *feature* que podem ser medidas, por exemplo, latência e disponibilidade podem ser atributos da *feature* Conexão de Internet;
- **Domínio de um atributo** – representa o espaço de valores que ele pode assumir, podendo ser discreto (inteiros, booleanos) ou contínuo (reais); e
- **Features extra-funcionais** – correspondem aos relacionamentos entre um ou mais atributos de uma *feature*, por exemplo, latência/disponibilidade > 50 e

largura de banda = 256. Essas relações são associadas a uma *feature*.

As restrições de obrigatoriedade, opcionalidade e de grupos de cardinalidade são herdadas da modelo de *features* com cardinalidade proposto por Czarnecki, Helsen e Eisenecker (2005b).

A Figura 5 ilustra o modelo de *features* estendido, neste exemplo as extensões propostas por Benavides, Trinidad e Ruiz-Cortés (2005) foram utilizados com o propósito de representar o tempo e o preço para o desenvolvimento de cada uma das *features*. As *features* extra-funcionais são representadas por retângulos pontilhados, contento os relacionamentos entre um ou mais atributos, enquanto que as *features* funcionais são representados por retângulos não pontilhados.

Figura 5 – Modelo de *features* com notação da abordagem (Ext-FM).



Fonte – Adaptada de Benavides, Trinidad e Ruiz-Cortés (2005).

Considerando a *feature* funcional Serviço, as *features* extras-funcionais relacionadas a ela, com suas relações entre os atributos, por exemplo, disponibilidade, latência, confiabilidade, tempo de desenvolvimento, dentre outros, são consideradas representações de RNFs.

2.3.1.2 Integrated Software Product Line Model (ISPLM)

Siegmund et al. (2008) apresentam uma abordagem que integra unidades de código dentro do modelo de *features* para representar a implementação de uma *feature*, assim como suas propriedades não-funcionais. Para representar essas propriedades Siegmund et al. (2008) estende o modelo de *features* com a adição novos elementos:

- **Unidade de código** – representado no modelo de *features* para indicar componentes que implementam uma determinada *feature*. Por exemplo, na

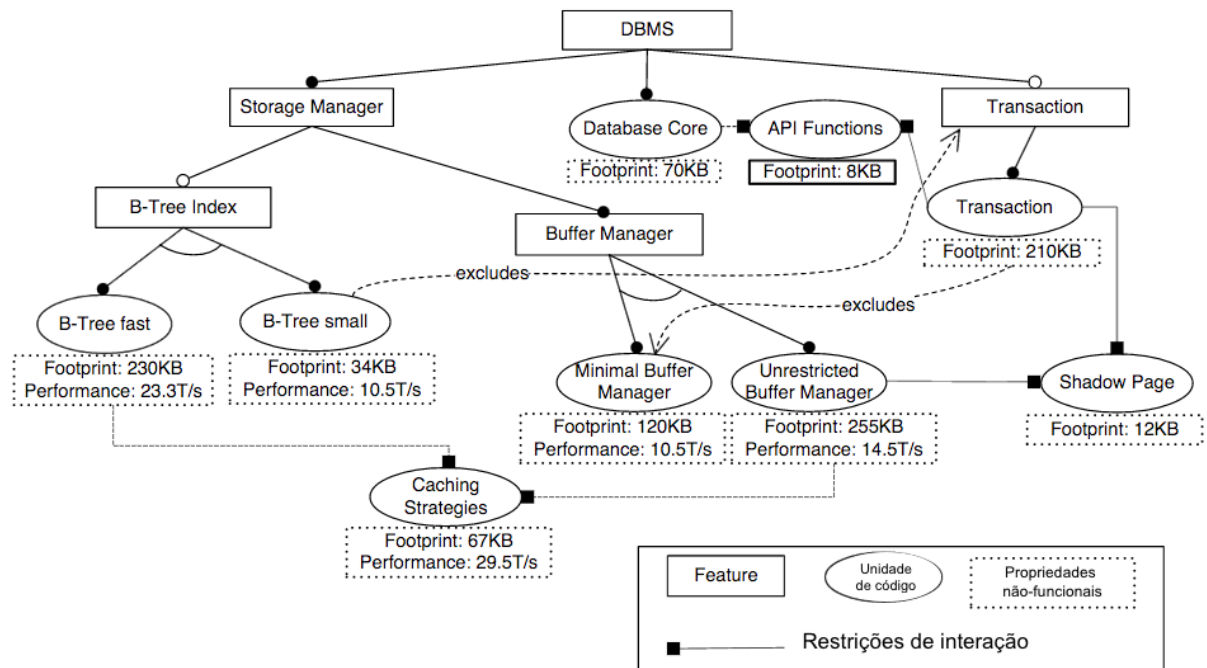
Figura 6, a unidade de código *Database Core* implementa a *feature DBMS*; e

- **Propriedades não-funcionais** – representam propriedades não-funcionais que são associados a uma unidade de código.

Nesta notação uma unidade de código, pode apenas ser elementos filhos de uma *feature* ou de outra unidade de código, ou seja, não é permitido modelar uma unidade de código como um nó raiz ou como um nó pai de uma *feature*, uma vez que as *features* são definidas durante a análise de domínio que antecede a fase de implementação (SIEGMUND et al., 2008). A sintaxe para representar restrições de obrigatoriedade, opcionalidade e de grupos de cardinalidade entre as unidades de código e as *features* são semelhantes as da notação FODA com cardinalidade.

A Figura 6 ilustra o modelo de *features* de uma sistema de gerenciamento de banco de dados, com as extensão propostas por Siegmund et al. (2008). Observa-se que as propriedades não-funcionais representadas nesta notação, correspondem a RNFs no modelo de *features*.

Figura 6 – Modelo de *features* com notação da abordagem (ISPLM).



Fonte – Adaptada de Siegmund et al. (2008).

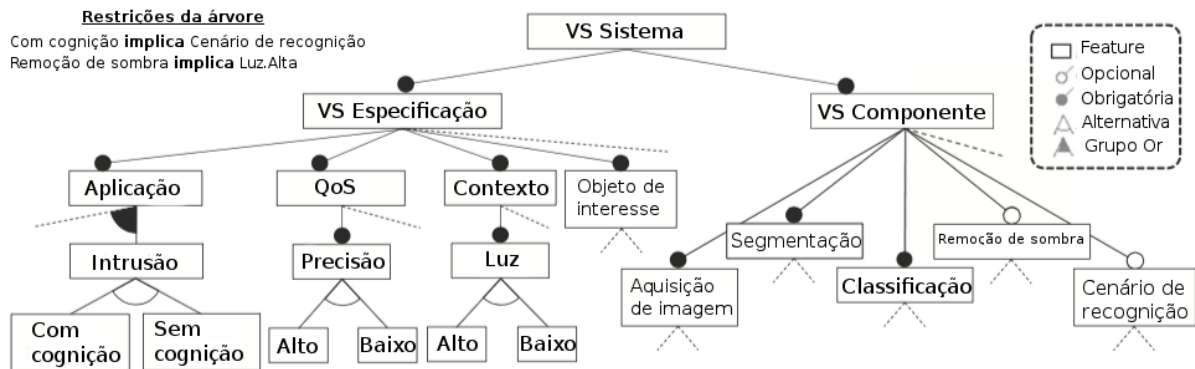
2.3.1.3 Feature Models and Quality Criteria (FM-QC)

Sanchez et al. (2015) apresentam uma abordagem para especificação, medição e otimização de propriedades de atributos de qualidade. Para representar essas propriedades Sanchez et al. (2015) estendem o modelo de *features* com foco na separação de requisitos e de

implementação, dividindo o nó raiz em dois sub-modelos (ver Figura 7):

- **VS Especificação** – representam as especificações da linha em funcionalidades de aplicação, objetos de interesse, atributos de qualidade (QoS) que são decididos pelo usuário e condições de contexto (Contexto);
- **VS Componente** – representam os componentes do sistemas e seus parâmetros, correspondendo a como o software deve fazê-lo, por exemplo, em um sistema de vigilância a classificação de objetos identificados pode ser realizada de duas formas: pela identificação de qualquer objeto ou ser restrita a identificação de pessoas. Dessa forma o tipo de classificação representam variações de como a classificação pode ser realizada.

Figura 7 – Modelo de *features* com notação da abordagem (FM-QC).



Fonte – Adaptada de Sanchez et al. (2015).

As restrições do modelo de *features* são similares aos da notação FODA com cardinalidade. Diferentemente das outras notações apresentadas, a utilizada nesta abordagem, divide o modelo de *features* em sub-módulos, dessa forma os atributos de qualidade ou RNFs são representados no sub-módulo VS Especificação, como as *features* filhas da subárvore *QoS*.

2.4 Ferramenta DyMMer

DyMMer¹ (*Dynamic Feature Model Tool Based on Measures*) é uma ferramenta desenvolvida para extrair medidas de diferentes modelos de *features*, que podem ser descritas usando o formato de arquivo (XML) propostos pelo repositório de modelo de *features* S.P.L.O.T

². A ferramenta foi desenvolvida em Java, utilizando a plataforma J2SE (*Java 2 Standart Edition*)

¹ <https://github.com/DyMMerProject/DyMMerV2>

² <http://www.splot-research.org/>

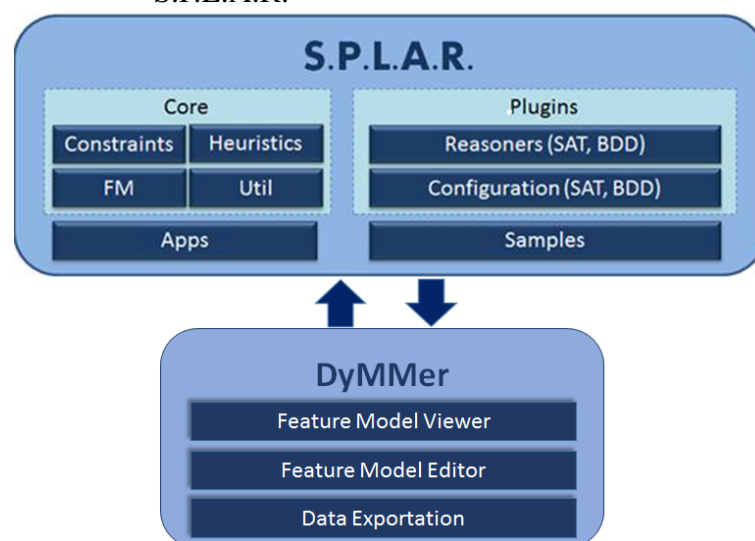
para uso *desktop*, de forma a atender as necessidades do usuário sem o uso da internet (FREIRES JUNIOR, 2014).

Atualmente, a ferramenta suporta as seguintes funcionalidades: i) Importação do modelo de *features* com base em estruturas de arquivos gerados pelo S.P.L.O.T; ii) Visualização do modelo de *features* conforme o contexto selecionado; iii) Análise de diferentes medidas de qualidade sobre o modelo de *features* de LPS e LPSD; iv) Edição dos modelos de *features* para a inserção de adaptações de contexto incluindo regras de adaptação de contexto para ativação e desativação de *features*; e v) Exportação da análise dos resultados das medidas de qualidade, com base em um ou mais modelos de *features* com ou sem contexto (FREIRES JUNIOR, 2014).

Segundo Freires Junior (2014), para suportar as funcionalidades citadas acima, a ferramenta foi desenvolvida com arquitetura baseada em três camadas (ver Figura 8):

- **Camada de visualização de modelos** - possibilita a visualização dos modelos de *features*, assim como o contexto ao qual esta inserido;
- **Camada de edição dos modelos** - permite a modelagem de adaptações de novos contextos e regras de restrições conforma edita-se o modelo; e
- **Camada para exportação dos dados** - permite a exportação dos valores gerados a partir das medidas de qualidade aplicadas aos modelos de *features* com e sem contexto para uma planilha no formato *Microsoft Office Excel*.

Figura 8 – Representação arquitetural da DyMMer e S.P.L.A.R.



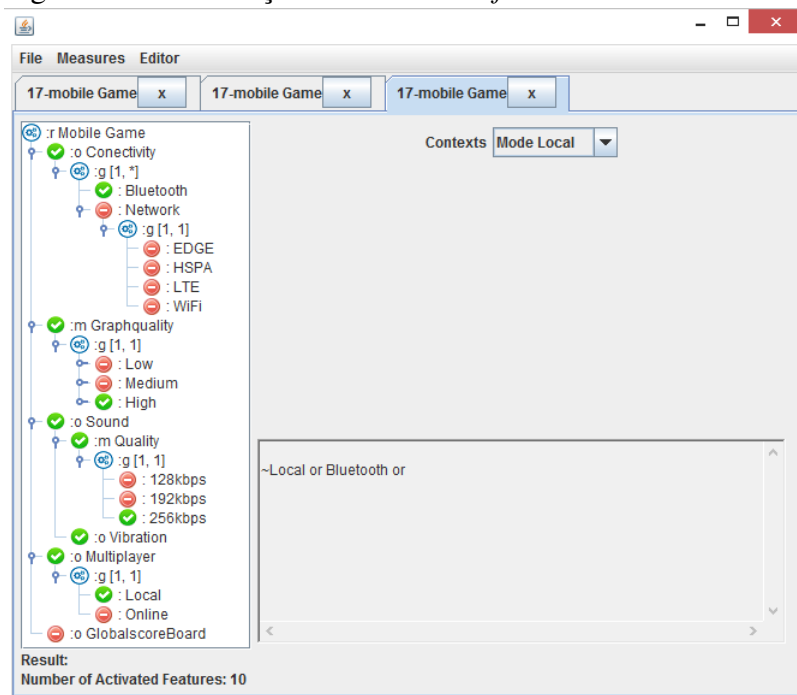
Fonte – Bezerra et al. (2016).

A ferramenta DyMMer utiliza-se do componente S.P.L.A.R como biblioteca, para

verificar de forma automática a integridade dos modelos de *features* consistentes, que possuem pelo menos uma configuração válida, e os modelos de *features* corretos, que não possui *features* mortas, utilizando uma resolução de satisfatibilidade (*SAT solving*) e diagramas de decisão binária (BDD) (MENDONCA et al., 2008).

Para realizar a importação de modelos de *features*, a DyMMer recebe como entrada modelos de *features* existentes que são descritos no formato do S.P.L.O.T em XML. Então, a ferramenta recebe como entrada um modelo em formato XML, processa o arquivo e cria uma representação em forma de lista hierarquizada, conforme ilustrado na Figura 9 (FREIRES JUNIOR, 2014). Além disso, áreas específicas para edição dos modelos como observado na Figura 10.

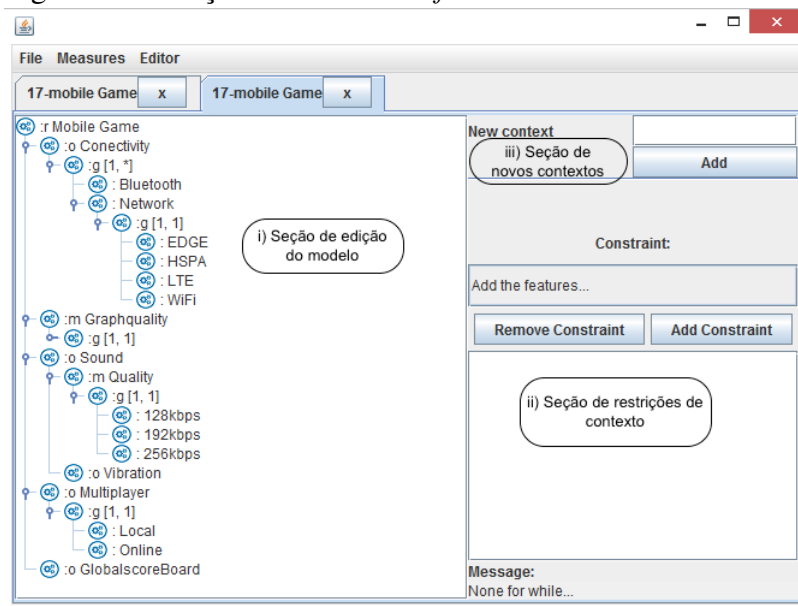
Figura 9 – Visualização do modelo de *features*.



Fonte – Freires Junior (2014).

Na Figura 9, verificam-se áreas específicas para a visualização do modelo de *features*, representado em uma estrutura de lista hierárquica do lado esquerdo da Figura, e outra para exibir informações do modelo. Do lado superior direito da Figura é exibido o contexto presente no modelo em visualização. Informações adicionais, na parte inferior da Figura, que exibe a medida escolhida, conforme selecionada no *menu* de *Measures* da aplicação e o resultados obtido para cada medida. Assim como indicadores de ativação e desativação de *features* representado na lista hierárquica (FREIRES JUNIOR, 2014).

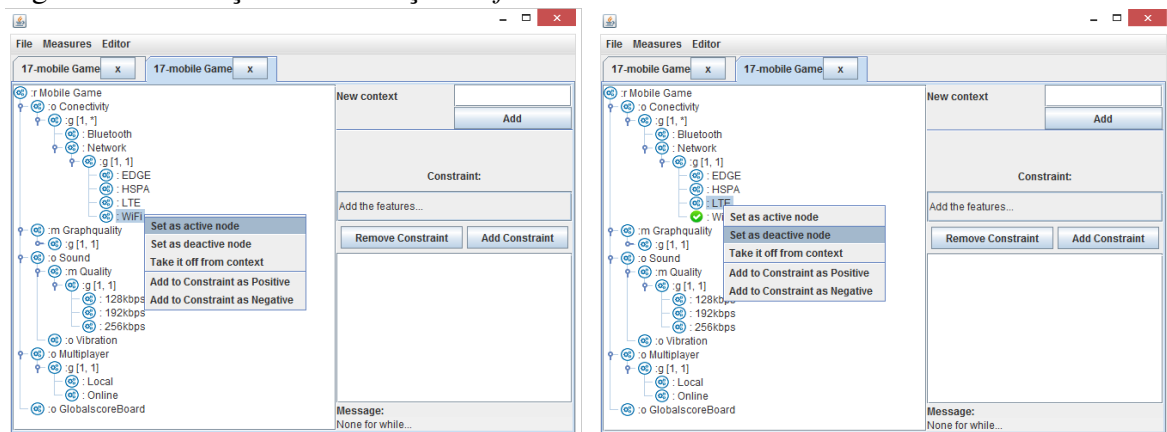
Figura 10 – Edição do modelo de *features*.



Fonte – Freires Junior (2014).

Para realizar a edição do modelo de *features*, como ilustrado na Figura 10, foram definidas três seções: i) seção de edição de modelo, em que uma *feature* é representada de acordo com o contexto em edição. Além disso, a adição de regras de restrições, dada a seleção de uma determinada *feature* alterando o seu estado de ativada ou não ativada, conforme ilustrado na Figura 11; ii) adição de restrições de integridade de contexto, pode-se visualizar as restrições que estão sendo criadas, assim como a adição e remoção de restrições, sejam elas em criação ou que já foram adicionadas ao contexto de edição; e iii) adição de novos contexto, em que é possível adicionar um nome ao contexto atual em edição, associando-o ao modelo de *features* (FREIRES JUNIOR, 2014).

Figura 11 – Ativação e desativação de *features*.



Fonte – Freires Junior (2014).

3 TRABALHOS RELACIONADOS

Para o embasamento deste trabalho foi realizada uma busca bibliográfica, com o objetivo de identificar trabalhos relacionados. Nesta Seção são discutidos os trabalhos relacionados e identificados pontos que auxiliaram na execução deste trabalho.

Zhang, Ye e Lin (2014) apresentam uma modelagem de atributo de qualidade e configuração do produto cientes de qualidade em LPSs. Semelhante ao objetivo deste trabalho, os autores propuseram uma abordagem para modelagem de atributos de qualidade no modelo de *features* conduzindo a uma configuração do produto ciente de qualidade. Os autores executam os seguintes passos: i) identificação e representação de atributos de qualidade no modelo de *features*; ii) medição de interdependência entre *features* e atributos de qualidade; e iii) configuração do produto ciente de qualidade. Entre as fases i) e ii) os autores estendem o modelo de *features* com a modelagem de atributos de qualidade para a Engenharia de Domínio, enquanto que em iii), os autores configuram um produto ciente de qualidade para engenharia da aplicação.

Entretanto, neste trabalho, foi definida uma abordagem para identificação e representação de PNFs e identificação de restrições e cenários de adaptações de contexto, para o modelo de *features* de LPSDs, por meio da extensão do modelo de *features* representado pela notação FODA com cardinalidade para representação de RNFs e cenários de adaptações de contexto. Além disso, neste trabalho foi desenvolvido um extensão da ferramenta DyMMer para suportar a aplicação da abordagem.

Sanchez et al. (2015) apresentam uma abordagem para sistemas autoadaptativos, com base no processo de especificação, medição e otimização de propriedades de atributos de qualidade em modelos de *features*. Além disso, os autores descrevem uma plataforma para apoiar a auto adaptação de sistemas baseados em componentes. Os autores representam propriedades e medidas de atributos de qualidade no modelo de *features* dividindo o modelo de *features* em dois submódulos, um para especificação dos componentes da linha de produtos e outro para especificação de atributos de qualidade, objetos de interesse e contexto.

Neste trabalho, a abordagem desenvolvida teve como base a identificação e representação de PNFs no modelo de *features* e a identificação de restrições entre PNFs e adaptações de contexto. Além disso, ao invés de estender o modelo de *features* em dois submódulos, foi realizado a criação de um modelo de *features* de qualidade para suportar a representação de PNFs. Esta representação foi suportada pela extensão da ferramenta DyMMer, que já oferece suporte à representação de adaptações de contexto e medidas de qualidade para o

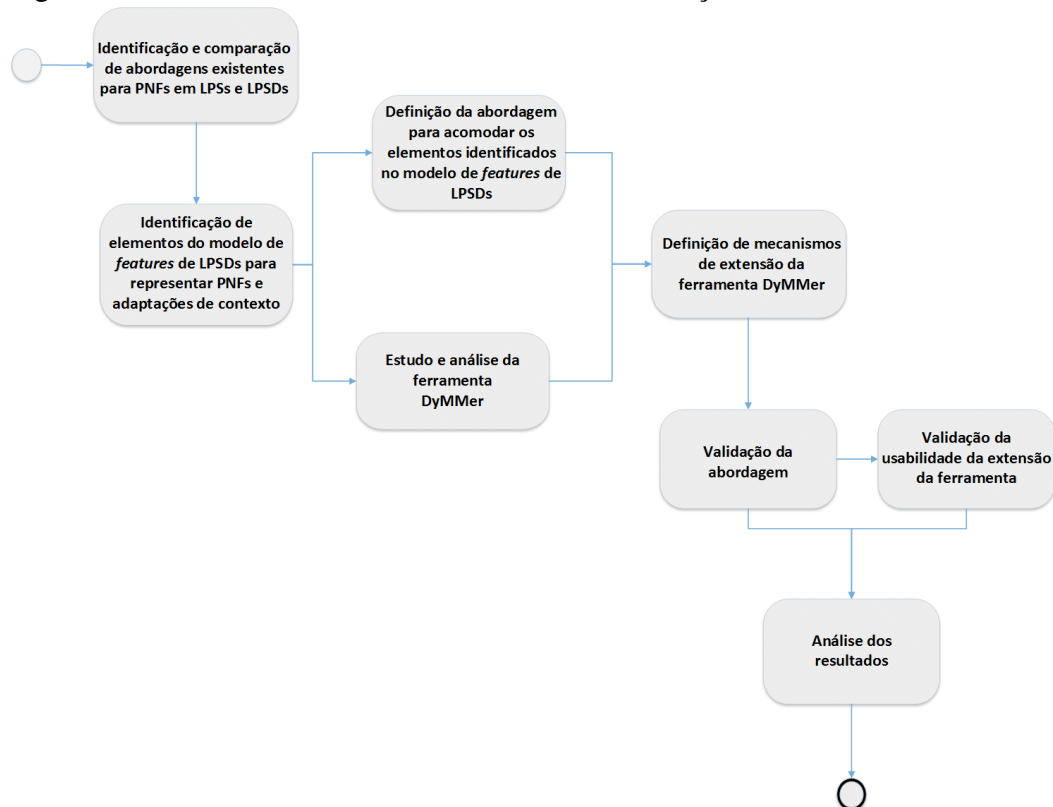
modelo de *features* de LPS e LPSD.

4 PROCEDIMENTOS METODOLÓGICOS

O objetivo desta Seção é apresentar os passos estabelecidos para a execução deste trabalho. A Figura 12 apresenta as etapas e atividades relacionadas, definidas a partir da identificação e comparação de abordagens existentes para PNFs em LPSs e LPSDs até a análise dos resultados deste trabalho.

Primeiramente foi realizada a identificação e comparação de abordagens existentes para modelagem de PNFs em LPSs e LPSDs. Em seguida, foi realizada a identificação de elementos do modelo de *features* de LPSDs para representar PNFs e adaptações de contexto. Após esta etapa, foi executada a atividade de definição da abordagem para acomodar os elementos identificados no modelo de *features* de LPSDs em paralelo com o estudo e análise da ferramenta DyMMer. Posteriormente, foram definidos os mecanismos de extensão da ferramenta DyMMer, para suportar a abordagem definida. Em seguida foi realizada a extensão da ferramenta DyMMer. Por fim, foi executada a etapa de validação da abordagem e da usabilidade da ferramenta após a adição das extensões.

Figura 12 – Procedimentos realizados durante a execução deste trabalho.



Fonte – Elaborada pelo Autor.

4.1 Identificação e comparação de abordagens existentes para PNFs em LPSs e LPSDs

Na primeira etapa para o desenvolvimento deste trabalho foi a realização uma revisão da literatura de forma não sistemática, com o objetivos de identificar abordagens que modelem PNFs em LPSs e LPSDs. Ao final da revisão, foram identificadas três abordagens, Ext-FM, ISPLM, e FM-QC, sendo a abordagem FM-QC a única identificada neste trabalho que aplica-se a LPSDs. Após a identificação das abordagens, foi definido um conjunto de critérios, a fim de avaliá-las em relação as suas deficiências e limitações. Foram escolhidos oito critérios. A seguir, cada critério utilizado para avaliação das abordagens são descritos:

- **PNFs quantitativos e qualitativos** – se a abordagem suporta a representação de PNFs quantitativos e qualitativos, caso a abordagem ofereça suporte a representação de ambas as PNFs ela é bem avaliada neste critério;
- **Suporte a LPSDs** – se a abordagem oferece suporte a LPSDs, desse modo podemos distinguir abordagens voltadas para LPSs e LPSDs;
- **Restrições de contexto** – se a abordagem prevê a representação de restrições de adaptações de contexto;
- ***Binding times*** – se a abordagem prevê a representação de ativação e desativação de *features* em tempo de execução ou em ambiente simulado;
- **Suporte a ferramenta** – se a utilização da abordagem é suportada por uma ferramenta, seja ela um protótipo ou uma ferramenta já consolidada;
- **Configuração em *runtime*** – se a abordagem por meio de sua ferramenta suportada, caso houver, apresenta um mecanismo de configuração do modelo de *features* em tempo de execução de forma completa, ou parcial de acordo com as adaptações de contexto;
- **Modelo de *features* único** – se o modelo de *features* representado pela abordagem é único, ou seja, não divide o modelo em sub-módulos ou utiliza dois ou mais modelos de *features* para representar informações relevantes, por exemplo, um modelo de *features* para representar o contexto e outro para representar as *features*, aumenta a complexidade estrutural do modelo; e
- **Avaliação da qualidade do modelo de *features*** – se a abordagem ou a ferramenta suportada por ela, avalia a qualidade estrutural do modelo de *features*.

Com os critérios de avaliação estabelecidos, foi realizada uma comparação entre as abordagens identificadas e a abordagem proposta.

Quadro 3 – Comparação de abordagens que representam PNFs

Critérios	Ext-FM	ISPLM	FM-QC	Proposta
PNFs quantitativos e qualitativos	×	×	✓	✓
Restrições de contexto	×	✓	✓	✓
Configuração em <i>runtime</i>	×	Parcialmente	✓	Parcialmente
<i>Binding times</i>	×	×	✓	✓
Suporte a LPSDs	×	×	✓	✓
Suporte a ferramenta	✓	✓	✓	✓
Modelo de <i>features</i> único	✓	×	×	✓
Avaliação estrutural do modelo de <i>features</i>	×	×	×	✓

Fonte – Uchôa e Bezerra (2016).

O Quadro 3, apresenta o resultado da comparação entre as abordagens identificadas que representam PNFs em LPSs e LPSDs e a abordagem proposta neste trabalho. Os critérios estabelecidos para análise das abordagens encontram-se representadas na primeira coluna vertical contrapondo-se as abordagens avaliados, situados na primeira linha horizontal. O processo de análise foi realizado por dois avaliadores, em que cada abordagem foi avaliada em conformidades aos critérios estabelecidos. Verificam-se deficiências entre as abordagens que não suportam LPSD na modelagem de PNFs quantitativos e qualitativos, representação de PNFs em um modelo de *features* único e avaliação estrutural do modelo de *features*. A abordagem FM-QC que apresenta suporte a LPSD, embora seja bem avaliada nos critérios estabelecidos, não realiza a avaliação de qualidade estrutural do modelo de *features*, e divide o modelo de *features* em submódulos, aumentando a sua complexidade estrutural.

4.2 Identificação de elementos do modelo de *features* de LPSDs para representar PNFs e adaptações de contexto

Após a análise das deficiências e limitações identificadas anteriormente, identificou-se a necessidade de prover uma extensão do modelo de *features* que utiliza a notação FODA com cardinalidade (CZARNECKI; HELSEN; EISENECKER, 2005b), adicionando novos de elementos que auxiliem na representação de PNFs, restrições entre *features* e PNFs, e adaptações de contexto no modelo de *features* de LPSDs. Os seguintes elementos foram adicionados:

- **Modelo de *features* de qualidade** – este modelo visa a representação de PNFs que emergem em tempo de execução. Este modelo utiliza a norma ISO/IEC 25010 (SQuaRE) (ISO/IEC, 2011), para representar de forma satisfatória uma

PNF. Desse modo, o modelo de *features* de qualidade foi dividido em três níveis, no primeiro nível do modelo de *features* de qualidade, é representada pela *feature* raiz, o segundo nível é composto por características de qualidade, já o terceiro nível representa as subcaracterísticas de qualidade relacionadas a uma característica de qualidade, por fim, o quarto nível representa as PNFs;

- **Links de contribuição** – para representar as restrições de interdependências entre as *features* e as PNFs foi adicionado o conceito de *links* de contribuição, originalmente utilizado para representar restrições no domínio de engenharia de requisitos orientada a objetivos. Por meio destes *links* de contribuição foi possível representar as interdependências de acordo com o impacto que uma *feature* possui quando ela é ativada ou desativada em relação a uma PNF definida. Este impacto representa se uma determinada *feature* satisfaz completamente ou negativamente uma PNF, ou se possui uma influência positiva ou negativa sobre uma PNF; e
- **Representação de cenários de adaptações de contexto** – para representar as adaptações de contexto por meio de cenários foi proposto a identificação de informações de contexto relevantes e definição de regras de contexto, desse modo foi possível identificar e representar as adaptações de contexto e quais *features* devem ser ativadas e desativadas em um determinado cenário de adaptação.

4.3 Definição da abordagem para acomodar os elementos identificados no modelo de *features* de LPSDs

Após a identificação dos elementos do passo anterior, foi realizada a definição da abordagem para a especificação de um modelo de *features* de LPSDs que incorpore os elementos identificados no passo anterior. A abordagem proposta neste trabalho foi definida em duas fases: (1) Identificação e Representação de PNFs no Modelo de *Features*; e (2) Identificação de Restrições e Cenários de Adaptações de Contexto.

A fase inicial (1) tem como objetivos representar as PNFs por meio do modelo de *features* de qualidade. Nesta fase foi definido um catálogo de 52 PNFs que emergem em tempo de execução, as PNFs utilizadas neste catálogo são classificadas de acordo com as características e subcaracterísticas de qualidade da norma ISO/IEC 25010 (SQuaRE) (ISO/IEC, 2011). A fase (2) visa identificar as restrições de ativação e desativação entre as *features* e os cenários de

adaptações de contexto, com base na definição de regras de contexto. Além disso, definir a influência de uma *feature* de contexto ativada e desativada sobre uma PNF em um determinado cenário de adaptação de contexto. Nesta fase foram definidos um conjunto de *templates* para auxiliar na definição das restrições e identificação dos cenários de adaptações de contexto.

4.4 Estudo e análise da ferramenta DyMMer

Nessa etapa foi realizado o estudo da arquitetura da ferramenta DyMMer, concentrando-se nos módulos de edição e de visualização do modelo de *features*. Verificou-se a possibilidade de reutilização de funcionalidade já implementadas no módulo de edição, tais como, atribuição de ativação e de desativação de *features* e adição de regras de composição do modelo. Entretanto, a funcionalidade de adição de contexto no modelo de *features* precisou ser estendida para possibilitar a adição de múltiplos cenários de adaptações de contexto em um mesmo modelo de *features*. Em relação ao módulo de visualização do modelo de *features*, embora a ferramenta já possua uma forma de visualização satisfatória do modelo de *features*, devido a adição de múltiplos cenários de adaptações de contexto ao modelo de *features* foi identificado a necessidade de extensão desse módulo da ferramenta, uma vez que a visualização do modelo de *features* se restringia a um único contexto por vez.

4.5 Definição de mecanismos de extensão da ferramenta DyMMer

Nessa etapa foram definidos os mecanismos de extensão da ferramenta DyMMer de forma a suportar a abordagem definida. A implementação da extensão na ferramenta DyMMer, se concentrou nos módulos de edição e visualização do modelo de *features*, foram adicionadas as seguintes extensões de acordo com os módulos estendidos:

- **Módulo de edição** – criação do modelo de *features* de qualidade, adição do catálogo de PNFs a ferramenta, adição de múltiplos cenários de adaptações de contexto e adição de restrições de interdependências entre *features* e PNFs em um determinado cenário de adaptação de contexto; e
- **Módulo de visualização** – visualização da configuração do modelo de *features* de acordo com os múltiplos cenários de adaptações de contexto adicionados ao modelo de *features*.

4.6 Validação da abordagem

Para realizar a validação da abordagem foi conduzido um estudo de observação, no qual o participante realiza alguma tarefa enquanto é observado pelo experimentador. O estudo de observação tem como finalidade coletar dados sobre como uma determinada tarefa é realizada. Desse modo, pode-se obter informações de como a abordagem é utilizada. O objetivo do estudo de observação foi analisar o processo definido na abordagem, com a finalidade de caracterizar sua aplicação em relação à modelagem de PNFs e de cenários de adaptações de contexto com base no modelo de *features* de LPSDs.

4.7 Validação da usabilidade da ferramenta estendida

Para avaliar a ferramenta estendida, foi realizado um teste de usabilidade. Esse teste teve como objetivo avaliar a usabilidade da ferramenta, bem como a qualidade visual dos artefatos implementados. Neste trabalho foi utilizado um dos quatro métodos de avaliação da usabilidade proposto por Lewis (1995). O método escolhido foi o *The Post-Study System Usability Questionnaire* (PSSUQ). Esse método permite avaliar quatro diferentes fatores a partir das respostas obtidas por um questionário de avaliação composto por 19 perguntas. Os fatores são: satisfação geral, usabilidade da ferramenta, qualidade da informação e qualidade da *interface*. Este método foi escolhido porque possibilitou avaliar a satisfação geral do usuário em relação a ferramenta. Assim como a utilidade da ferramenta na execução do processo da abordagem. Além de ser possível avaliar a qualidade da *interface*.

4.8 Análise dos resultados

Para realizar a avaliação dos resultados da abordagem foram coletados e analisados dados resultantes das tarefas realizadas por cada participante do estudo de observação. Os principais dados coletados foram o tempo e esforço para realização de cada tarefa, a corretude das respostas de cada participante, os benefícios da utilização dos artefatos criados e as vantagens e desvantagens em relação a abordagem.

A partir desses dados foi possível verificar que todos os participantes não tiveram um esforço alto para realização das tarefas. Além disso que a abordagem conseguiu realmente cumprir com o que propõe, pois é de fácil aprendizagem e fácil de se aplicar. Em relação aos

artefatos criados todos os participantes concordaram que eles foram úteis para a realização das tarefas. A desvantagem é que se a abordagem for realizada sem uma ferramenta para automatizar o seu processo, a utilização da abordagem pode ser demorada e cansativa. A principal vantagem é que a abordagem permite identificar as propriedades não-funcionais e sua relação com o comportamento do modelo de *features*, o que é uma coisa não trivial.

Para realizar a avaliação dos resultados do teste de usabilidade da ferramenta, foram coletados e analisados os dados em relação a satisfação geral, usabilidade da ferramenta, qualidade da informação e qualidade da *interface*. De modo geral, após a análise e interpretação de todos os resultados obtidos, conclui-se que a ferramenta é útil para operacionalização da abordagem.

5 REMINDER: UMA ABORDAGEM PARA MODELAGEM DE PROPRIEDADES NÃO-FUNCIONAIS EM LINHAS DE PRODUTOS DE SOFTWARE DINÂMICAS

Como apresentado na Seção 2 o modelo de *features* é um dos principais artefatos de uma LPS e LPSD, sendo utilizado como ponto de partida para a configuração e instanciação de novos produtos, além de servir como *interface* de comunicação padronizando o entendimento do domínio entre os *stakeholders* envolvidos no processo de desenvolvimento de uma LPS e LPSD.

Na Seção 3, foi visto que as abordagens identificadas para o desenvolvimento de LPSs e LPSDs não apresentam uma forma satisfatória de representar PNFs e adaptações de contexto relevantes para um determinado domínio em modelo de *features*. No entanto, essas informações são elementos relevantes para a Engenharia de Domínio para o desenvolvimento de LPS e LPSDs e, portanto, devem ser identificados e representados no modelo de *features* em tempo de projeto.

Com a análise dessas abordagens, foram definidos os principais requisitos para a abordagem proposta neste trabalho, que incluem:

- Identificação de elementos para suportar a representação de PNFs e adaptações de contexto em modelo de *features* de LPSDs de forma explícita;
- Identificação de cenários de adaptações de contexto na configuração dos produtos;
- Identificação de regras de contexto, restrições entre *features* e adaptações de contexto em múltiplos cenários de adaptação;
- Identificação de restrições entre *features* e PNFs em múltiplos cenários de adaptação; e
- Extensão da ferramenta DyMMer para suportar a operacionalização da abordagem proposta.

Como forma de atender aos requisitos listados, foi proposta a abordagem ReMINDER (*An AppRoach to Modeling Non-FunctIoNal Properties in Dynamic SoftwarE PRoduct Lines*) (UCHÔA et al., 2017), que tem como principal objetivo fornecer uma forma sistemática para identificação de PNFs e cenários de adaptações de contexto, com suas respectivas restrições, para apoiar a modelagem de *features* e representação de PNFs em modelos de *features* de LPSDs.

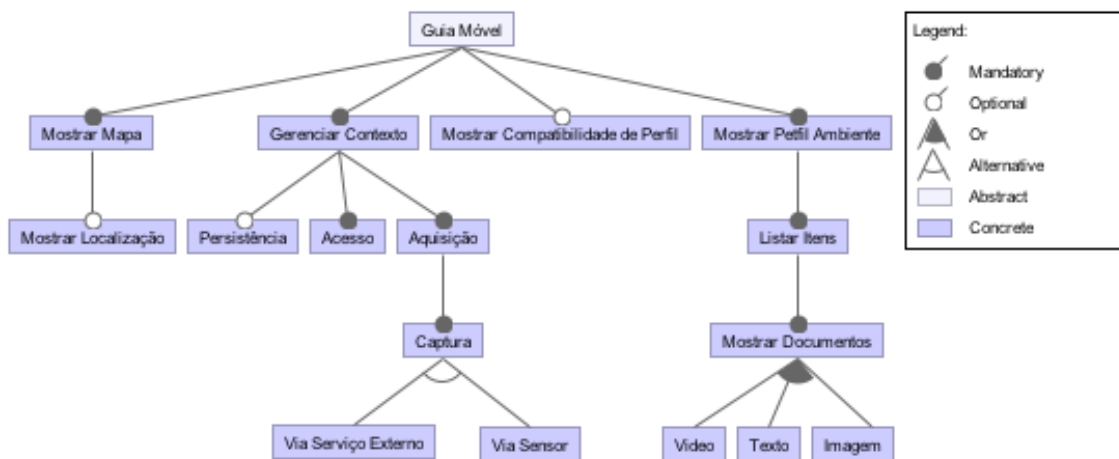
Essa Seção está organizada da seguinte forma: a Seção 5.2 apresenta uma visão geral da abordagem proposta; a Seção 5.1 descreve o modelo de *features* utilizado como exemplo para ilustrar os conceitos da abordagem; as Seção 5.3 e 5.4 mostram as principais fases e passos da

abordagem; e a Seção 7 descreve a extensão realizada na ferramenta DyMMer como parte da abordagem.

5.1 Exemplo para Guiar a Abordagem

O modelo de *features* utilizado para exemplificar a aplicação da abordagem proposta representa uma LPSD do domínio de Guia de Visitas Móvel - MobiLine (MARINHO et al., 2013). Parte do modelo de *features* da MobiLine é ilustrado na Figura 13.

Figura 13 – Domínio exemplo.



Fonte – Adaptado de Marinho et al. (2013).

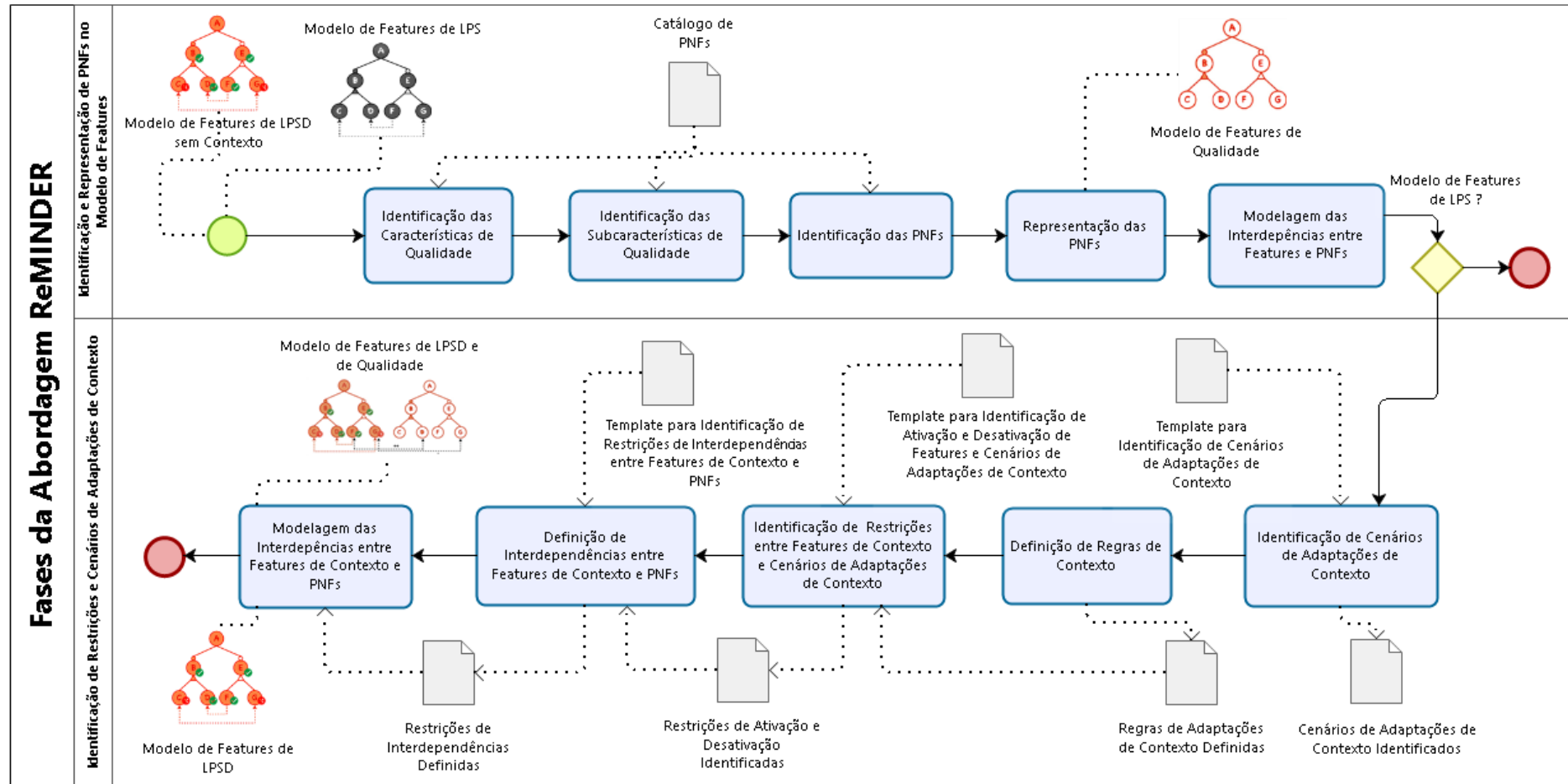
Esse modelo inclui *features*, como por exemplo: Mostrar Mapas do Ambiente e a Compatibilidade do Perfil do Visitante. A *feature* Gerenciar o contexto, representam *features* relacionadas ao Acesso do usuário, Persistência dos dados e formas de Aquisição de Contexto. A *feature* Aquisição é um grupo de cardinalidade que define a forma de captura de contexto, podem ser Via Serviço Externo ou por Via Sensor. Além disso, a *feature* Mostrar Perfil do Ambiente, é uma *feature* obrigatório, que representa os tipos de informações que podem ser mostrados sobre um determinado ambiente, estes tipos de informações são definidas pelo grupo de cardinalidade Mostrar Documentos, de modo que as informações podem ser mostradas por meio de Texto, Imagem ou Vídeo.

5.2 ReMINDER: Visão Geral

Uma visão geral o processo da nossa abordagem está ilustrada na Figura 14, incluindo as principais fases com seus artefatos relacionados. A abordagem é dividida em duas fases

principais: (1) Identificação e Representação de PNFs no Modelo de *Features*; e (2) Identificação de Restrições e Cenários de Adaptações de Contexto. Entre estas fases, (1) e (2) as PNFs são modeladas em um modelo de *features* de qualidade, enquanto que após a execução da fase (2) além das PNFs, são identificadas as adaptações de contexto, definidas regras de contexto, restrições entre *features* e adaptações de contexto e restrições de interdependências entre *features* e PNFs.

Figura 14 – Visão geral da abordagem.



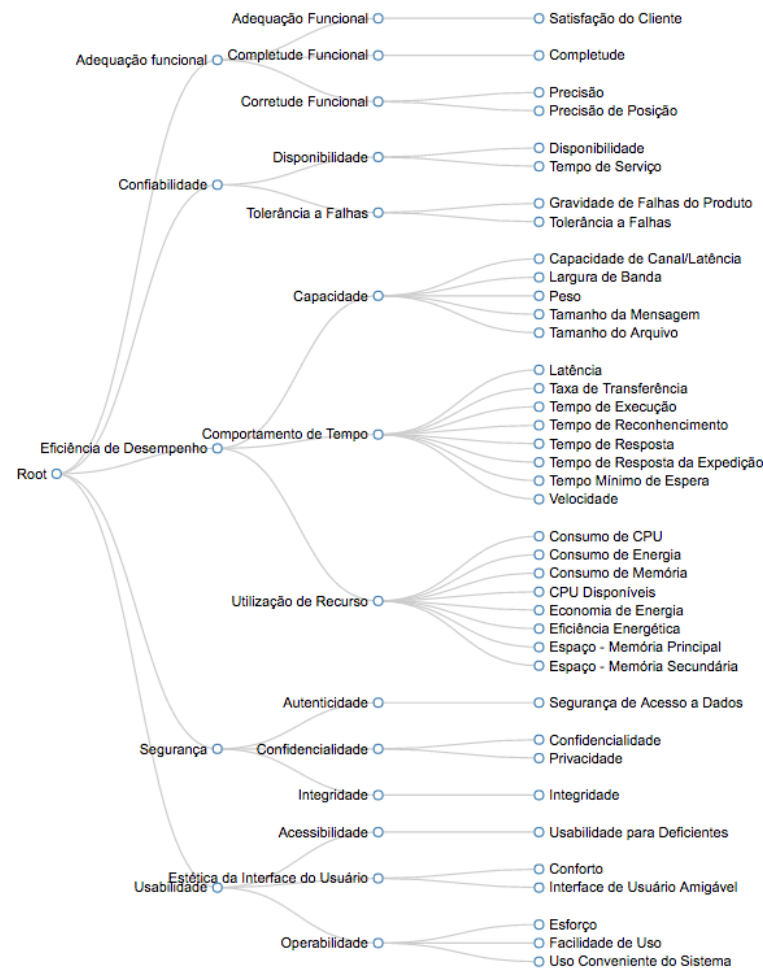
Fonte – Elaborada pelo Autor.

A fase inicial (1) tem como objetivo identificar as PNFs que são críticos para uma LPS ou uma LPSD e representá-los em um modelo de PNFs, tratado aqui como modelo de *features* de qualidade. A entrada dessa fase é um modelo de *features* de LPS ou LPSD sem contexto e modelados com a notação do método FODA com cardinalidade proposto por (CZARNECKI; HELSEN; EISENECKER, 2005a). A saída desta fase é o modelo de *features* de qualidade, juntamente com o modelo de *features* de LPS ou LPSD e suas restrições. A fase (2) visa identificar restrições entre os PNFs e adaptações de contexto com orientação de um Engenheiro de Domínio, por meio da identificação de restrições de ativação e de desativação de *features* e *features* de qualidade de acordo com os cenários de adaptações de contexto. A entrada dessa fase é o modelo de *features* de LPSD sem contexto e o modelo de *features* de qualidade da fase anterior, enquanto que como artefato de saída têm-se o modelo de *features* de LPSD com contexto e o modelo de *features* de qualidade, e os cenários de adaptações de contexto. Nas próximas Seções cada fase da abordagem ReMINDER é descrita em mais detalhes.

5.3 Fase I - Identificação e Representação de PNFs no Modelo de *Features*

A primeira fase da abordagem tem como objetivo identificar as PNFs que são críticas para uma LPS ou uma LPSD e representá-las no modelo de *features* de qualidade. Para auxiliar na identificação destes PNFs, foi criado um catálogo das principais PNFs que emergem em tempo de execução. Estas PNFs foram identificadas por Soares et al. (2014) (ver Quadro 2, página 29). Este catálogo classifica essas PNFs de acordo com cada característica e subcaracterísticas de qualidade apresentada no modelo de qualidade do produto da norma ISO/IEC 25010 SQuaRE (ISO/IEC, 2011), o catálogo é ilustrado na Figura 15.

Figura 15 – Catálogo de PNFs de acordo com as características e subcaracterísticas de qualidade da norma ISO/IEC (2011).



Fonte – Elaborada pelo Autor.

Para a execução desta fase devem ser realizados quatro passos. Primeiramente, a identificação das características de qualidade, em seguida, a identificação das subcaracterísticas de qualidade. Após este Passo, devem ser identificadas as PNFs. Por fim, a representação das PNFs no modelo de *features* de qualidade.

5.3.1 Passo I - Identificação de Características de Qualidade

Este passo tem como objetivo, identificar as características de qualidade com base nas necessidades dos *stakeholders*, essas características de qualidade são identificadas conforme o catálogo apresentado na Figura 15. Tendo como exemplo a LPS MobiLine apresentada na Seção 5.1, uma possível característica de qualidade relevante para este domínio é de Usabilidade, assim como a de Segurança, dentre outras.

5.3.2 *Passo II - Identificação de Subcaracterísticas de Qualidade*

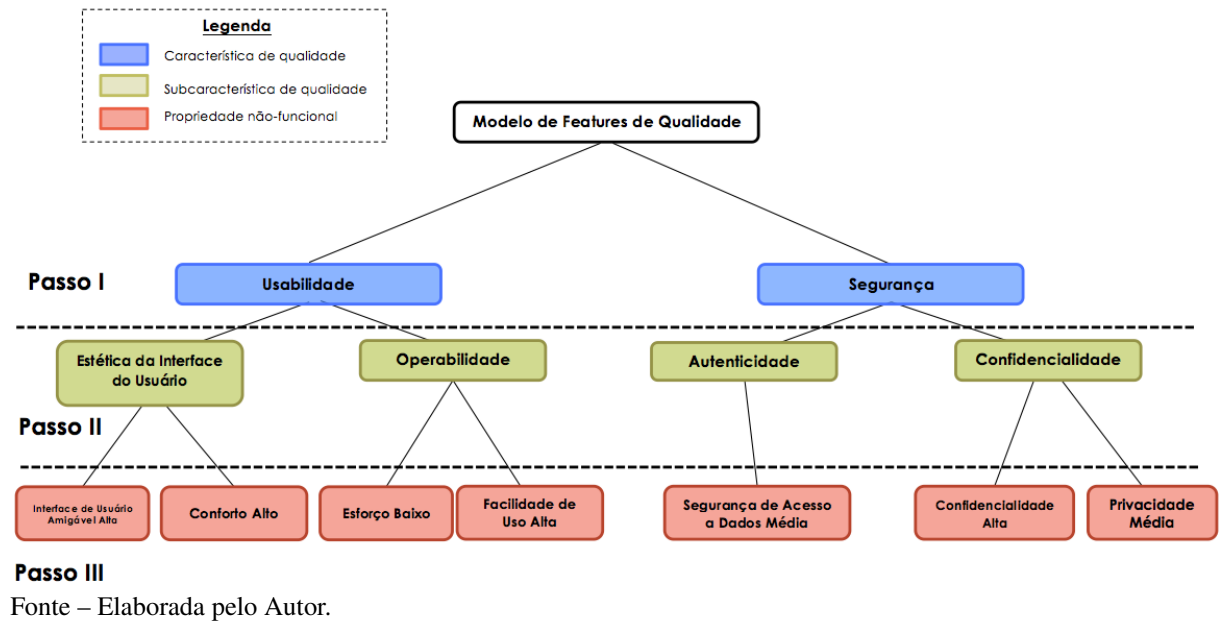
Após a identificação das características de qualidade, deve-se especificar as características de qualidade, em subcaracterísticas de qualidade relacionadas aumentando o grau de especificação, desse modo cada característica de qualidade identificada é decomposta em uma subcaracterísticas de qualidade relacionada, conforme a Figura 15. Por exemplo, Usabilidade pode ser decomposto em Estética da Interface do Usuário, enquanto que a subcaracterística de qualidade Segurança pode ser decomposta em Autenticidade, dentre outras.

5.3.3 *Passo III - Identificação de PNFs*

Uma vez realizada a identificação das subcaracterísticas de qualidade, deve-se adicionar as PNFs que emergem em tempo de execução. Para cada subcaracterísticas de qualidade, deve ser associada uma PNF relacionada, conforme o catálogo de PNFs ilustrado na Figura 15. Estas PNFs representam requisitos não-funcionais que devem ser atendidos pelos produtos derivados de uma LPS ou LPSDs. As PNFs podem ser mensuradas por qualificadores, normal, alto, médio ou baixo. Por exemplo, para a subcaracterísticas de qualidade Estética da Interface do Usuário, a PNF Interface de Usuário Amigável, pode ser mensurada como Interface de Usuário Amigável Alta, ou seja, para este domínio de aplicação é importante garantir que Interface de Usuário Amigável seja alta. Para subcaracterísticas de qualidade Autenticidade, a PNF Segurança de Acesso a Dados pode ser mensurada como Segurança de Acesso a Dados Média.

5.3.4 *Passo IV - Representação das PNFs*

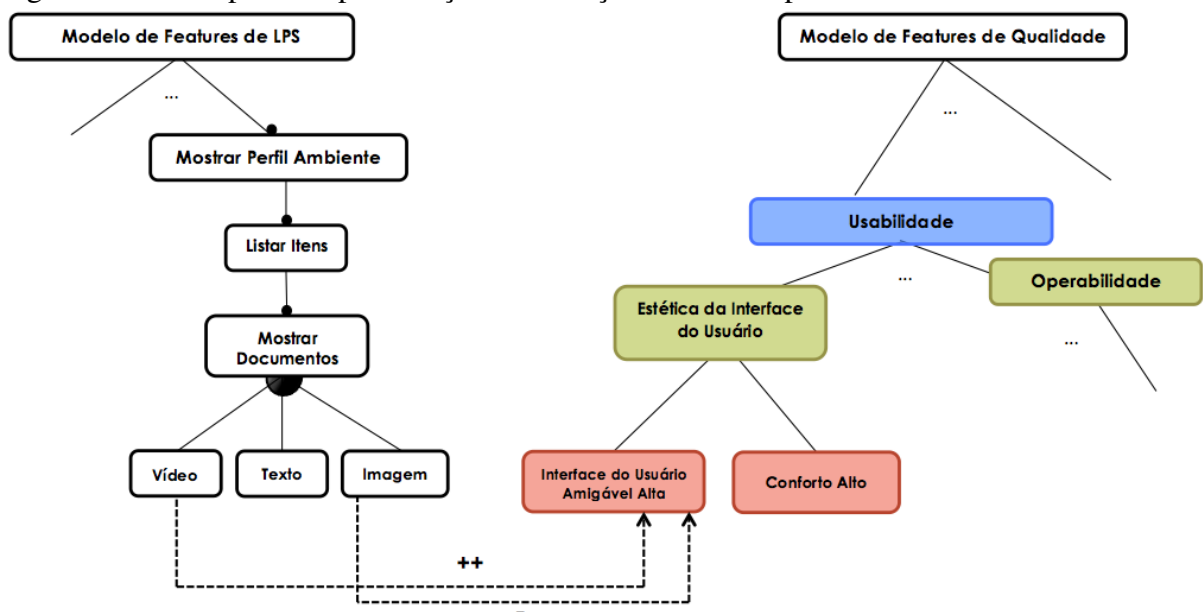
Finalmente, pode-se representar as PNFs, por meio de um modelo de *features* de qualidade, constituído por características, subcaracterísticas de qualidade e PNFs (ver Figura 16). É importante ressaltar que este modelo não é o modelo de *features* que representa a variabilidade do produto. Desse modo, no primeiro nível do modelo de *features* de qualidade, é representada pela *feature* raiz, o segundo nível é composto por cada característica de qualidade identificada no Passo I, tais como, usabilidade e segurança, dentre outros, já o terceiro nível representa as subcaracterísticas de qualidade relacionadas a cada característica de qualidade identificada, conforme o Passo II, por fim, o quarto nível representa as PNFs identificadas no Passo III.

Figura 16 – Exemplo de Modelo de *features* de qualidade

5.3.5 Passo V - Modelagem de Restrições de Interdependências entre Features e PNFs

Uma vez com as restrições identificadas, é possível mapeá-las para o modelo de *features* de LPS. Este mapeamento pode ser representado por meio da associação das *features* de e as PNFs, de forma que nesta associação é possível observar as restrições de interdependências entre as *features* e as PNFs (ver Figura 17).

Figura 17 – Exemplo da representação de restrições de interdependências entre Features e PNFs



Na Figura 17 é possível observar a representação das restrições de interdependências entre as *Features* e PNFs. A *feature* Vídeo quando é incluída possui uma restrição de satisfaz completamente (++) a PNF Interface de Usuário Amigável Alta, entretanto a *feature* Imagem quando não é incluída possui um influência negativa na PNF Interface de Usuário Amigável Alta.

Após a execução desta fase, se obtém como artefato de saída, o modelo de *features* de qualidade representando as PNFs que devem ser atendidas pelos produtos derivados de uma LPS ou uma LPSD. Este artefato juntamente com o modelo de *features* de LPSD caso seja ele a entrada da primeira fase são dados como entrada para a fase seguinte.

5.4 Fase II - Identificação de Restrições e Cenários de Adaptações de Contexto

A segunda fase da abordagem tem como objetivo identificar as restrições de ativação e desativação entre as *features* e os cenários de adaptações de contexto, com base na definição de regras de contexto. Além disso, definir a influência de uma *feature* de contexto ativada e desativada sobre uma PNF em um determinado cenário de adaptação de contexto.

Esta fase, tem como artefato de entrada o modelo de *features* de qualidade juntamente como o modelo de *features* de LPSD.

Para a execução desta fase devem ser realizados três passos. Primeiramente, a identificação dos cenários de adaptações de contexto, as definições das regras de contexto, em seguida, a identificação das restrições entre as *features* e cenários de adaptações de contexto. Após estes passos, devem ser identificadas as restrições de interdependências entre PNFs e *features* de contexto. Por fim, a representação destas restrições e cenários de adaptações de contexto.

5.4.1 Passo I - Identificação de Cenários de Adaptações de Contexto

Nesta abordagem é utilizado a definição e classificação de contexto proposta Hong, Chiu e Shen (2005). A classificação de contexto é apresentada no Quadro 4.

Quadro 4 – Classificação de tipos de contexto em categorias

Contexto Computacional	Contexto do Usuário	Contexto Físico
Dispositivos disponíveis	Preferência	Localização
CPU	Intenção	Tempo
Memória	Calendário do usuário	Destinação
Tamanho da tela	Informações pessoais	Condições de tráfego
Energia	Energia	Limitações físicas
Largura de banda	Facilidades	Clima
	Incapacidade	

Fonte – Adaptado de Hong, Chiu e Shen (2005).

Segundo Hong, Chiu e Shen (2005) entende-se por contexto computacional à configuração de hardware utilizado, por exemplo, os processadores disponíveis para realizar um tarefa, os dispositivos de entrada e saída para o usuário e a largura de banda. O contexto do usuário representa todos os fatores humanos, por exemplo, o perfil do usuário, calendários e perfis. Contexto físico, refere-se à informação não relacionada à computação, que são fornecidas por um ambiente do mundo real, por exemplo, localização, tempo, iluminação, níveis de ruído, condições de tráfego e temperatura.

As variações de contexto são identificadas de acordo com os possíveis cenários de adaptações do domínio da linha de produtos. As definições de cada cenário de adaptações de contexto descrevem situações relevantes para o domínio, tendo como base as seguintes propriedades, representadas no Quadro 5:

- **Identificador do cenário** – utilizado para identificar cada cenário, por exemplo, C1 ou C2.
- **Contexto** – utilizado para informar o tipo de contexto de acordo com a classificação apresentada no Quadro 4.
- **Informações de contexto** – utilizado para identificar as informações que sofrem variações de acordo com cada tipo de contexto. Em nosso modelo exemplo o nível de memória livre é uma informação de contexto relevante, esta informação de contexto é do tipo de contexto computacional.
- **Qualificadores** – utilizado para identificar as qualificações das variações de cada informação de contexto. Uma qualificação pode ser do tipo: booleano ou uma escala qualitativa (normal, alto, médio e baixo), por exemplo, a informação de contexto - nível de memória livre, pode ter variações medidas por nível normal e baixo, dessa modo sua qualificação é descrita em um escala qualitativa, por outro lado, a informação de contexto - conexão com internet, pode ter variações

medidas por *true* ou *false*, dessa modo sua qualificação é descrita de forma booleana.

- **Quantificadores** – utilizado para descrever os valores associados a cada tipo de qualificador, estes quantificadores são definidos por operadores relacionais: maior que ($>$), menor que ($<$), maior ou igual que ($>=$), menor ou igual que ($<=$), igual ($=$) e diferente ($<>$). Seguidos por um valor do tipo: *string*, inteiro, *float* ou booleano. Caso este quantificador for definido por um intervalo numérico, adiciona-se um operador lógico que pode ser do tipo: OU (OR) ou E (AND), seguidos por outro valor do tipo *string*, inteiro, *float* ou booleano. Por exemplo, nível de memória livre normal $>= 128$ ou nível de memória livre baixa < 128 .
- **Status dos quantificadores por cenário** – utilizado para indicar qual das variações de cada informação de contexto é considerada na definição do cenário, por exemplo, no cenário C1 dentre as variações do nível de memória livre, apenas a de nível normal é considerada na definição do cenário, desse modo o status dessa variação é (\checkmark), enquanto que as demais são (\times).

Quadro 5 – Modelo para identificação de cenários de adaptações de contexto

CENÁRIOS DE ADAPTAÇÕES					
Contextos	Informações de Contexto	Qualificadores	Quantificadores	Status Quantificadores C1	Status Quantificadores C2
Computacional	Nível de Memória Livre	Normal	$>= 128$	\checkmark	\times
		Baixo	< 128	\times	\checkmark
Computacional	Conexão com Internet	True	Não se aplica	\checkmark	\times
		False	Não se aplica	\times	\checkmark
Usuário	Similaridade de Perfil	True	Não se aplica	\checkmark	\times
		False	Não se aplica	\times	\checkmark
Físico	Ambiente do Usuário	True	Não se aplica	\checkmark	\times
		False	Não se aplica	\times	\checkmark

Fonte – Elaborada pelo Autor.

5.4.2 Passo II - Definição das Regras de Contexto

Após a identificação dos cenários de adaptações de contexto, é necessário definir as regras de contexto que especificam como uma informação de contexto, previamente especificada, impacta na configuração dos produtos de uma LPSD, indicando, por exemplo, quais *features* devem ser ativadas e desativadas em um grupo de cardinalidade. Essas regras têm como propriedades: um identificador e uma expressão.

O identificador define qual regra de contexto esta sendo definida, por exemplo RC1. A expressão é formada por um antecedente, um operador “*implica*”, e um consequente. Um

antecedente é uma expressão que pode conter uma informação de contexto com sua variação ativada ou desativada. O operador “*implica*” determinar que se o antecedente for considerado verdadeiro no cenário de adaptação, então o conseqüente deve ser ou não incluído na configuração do modelo de *features*. O conseqüente é uma expressão que pode conter uma *feature* e um operador lógico. Um operador lógico pode ser do tipo: E (AND) ou NÃO (NOT). A expressão determina quais *features* precisam ser ativadas ou desativadas para a configuração do modelo de *features* de acordo com cada cenário de adaptação de contexto. A Figura 18, apresenta um exemplo de regra de contexto.

Figura 18 – Exemplo de regras de contexto

RC 1

(Mesmo Ambiente de Usuário AND Similaridade de Perfil) *implica* Mostrar Compatibilidade de Perfil

RC 2

Nível de Memória disponível Baixo *implica* NOT (Persistência)

Fonte – Elaborada pelo Autor.

5.4.3 Passo III - Identificação das Restrições entre Features e Cenários de Adaptações de Contexto

O próximo passo é identificar as restrições de ativação e desativação de *features* de acordo com as regras de contexto que serão executadas com base nas definições dos cenários de adaptações de contexto ativadas e nas *features* presentes na configuração inicial do produto.

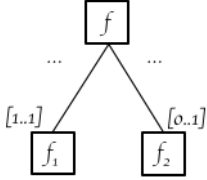
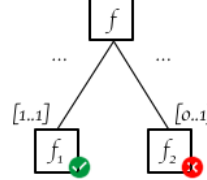
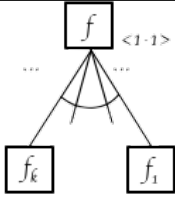
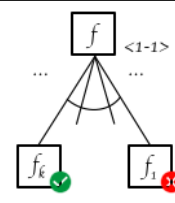
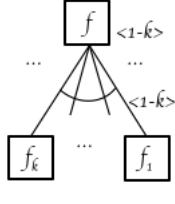
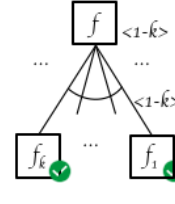
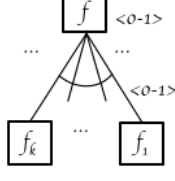
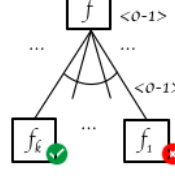
A ativação e desativação de *features* são propriedades que devem ser identificadas e representadas no modelo de *features* de LPSDs. Essas propriedades definem as possíveis configurações de um produto em um determinado contexto.

Conforme ilustrado na Figura 18, para um determinado cenário de adaptação de contexto em que o antecedente Mesmo Ambiente de Usuário e Similaridade de Perfil for verdadeiro, a RC1 deve ser executada, ou seja a *feature* Mostrar Compatibilidade de Perfil deve ser ativada.

É importante observar que os modelos de *features* de LPSDs sem contexto dados como entrada nesta fase, utilizam a notação do método FODA com cardinalidade, estes modelos de *features* já possuem restrições de cardinalidade de *features* e de grupos de cardinalidade, que generalizam se uma determinada *feature* é obrigatória ou opcional e se somente uma ou mais *features* podem fazer parte de um produto. Essas restrições podem ser mapeadas para restrições

de ativação e desativação de *features* (ver Quadro 6).

Quadro 6 – Mapeamento de restrições de ativação e desativação de modelos de *features* de LPSD

Tipo de <i>features</i>	Modelo de <i>features</i> de LPSD sem contexto	Modelo de <i>features</i> de LPSD com contexto
<i>Features</i> mandatórias e opcionais		
Grupos com cardinalidade $\langle 1 - 1 \rangle$		
Grupos com cardinalidade $\langle 1 - K \rangle$		
Grupos com cardinalidade $\langle 0 - 1 \rangle$		

Fonte – Elaborada pelo Autor.

Como apresentado no Quadro 6, todas as *features* que possuem restrições de obrigatoriedade $[1, 1]$ devem ser ativadas no modelo de *features* de LPSD, ao contrário das *features* com restrições de opcionalidade $[0, 1]$ que podem ser ativadas ou desativadas no modelo de *features* de LPSD. *Features* também possuem restrições de grupos de cardinalidade, determinados por um intervalo $\langle n, m \rangle$, onde n e m , determinam o número mínimo e máximo de *subfeatures* que podem fazer parte de um produto, caso a *feature* pai seja ativada. Desse modo o conjunto de *features* são divididos em relacionamento exclusivo (XOR) $\langle 1 - 1 \rangle$, em que somente uma das *subfeatures* pode fazer parte do produto, e não-exclusivo (OR) $\langle 1 - k \rangle$, em que mais de uma *subfeature* pode fazer parte do produto. Dessa forma, no máximo uma das *subfeatures* de um relacionamente (XOR) pode ser ativada no modelo de *features* de LPSD, ao contrário do (OR) que no mínimo uma das *subfeatures* pode ser ativada, mas limitada por no máximo k *subfeatures*.

Com o mapeamento das restrições de cardinalidade de *features* e grupos de

cardinalidade para restrições de ativação e desativação de *features*, e analisando as regras de contexto que devem ser executadas de acordo com os possíveis cenários de adaptações, têm-se a necessidade de identificar as restrições entre as *features* e cenários de adaptações de contextos. Desse modo, deve-se estabelecer as restrições de ativação e desativação das *features* de acordo com cada cenário de adaptação de contexto.

Quadro 7 – Modelo para identificação de restrições de ativação e desativação de *features* de acordo com cada cenário de adaptação de contexto

Nome da Feature	Variabilidade	STATUS DE ATIVAÇÃO POR CENÁRIOS	
		Cenário C1	Cenário C2
Mostrar Mapa	Obrigatória	✓	✓
Mostrar Localização	Opcional	✓	✓
Gerenciar Contexto.Persistência	Opcional	✓	×
Gerenciar Contexto.Acesso	Obrigatória	✓	✓
Gerenciar Contexto.Aquisição	Obrigatória	✓	✓
Capturar.Via Serviço Externo	-	✓	×
Capturar.Via Sensor	-	×	✓
Mostrar Compatibilidade de Perfil	Opcional	✓	×
Mostrar Perfil do Ambiente	Obrigatória	✓	✓
Listar Itens	Obrigatória	✓	✓
Mostrar Documentos.Vídeo	-	✓	×
Mostrar Documentos.Texto	-	✓	✓
Mostrar Documentos.Imagem	-	×	×

Fonte – Elaborada pelo Autor.

O Quadro 7, representa a identificação destas restrições, de modo que, para cada *features* do modelo de *features*, deve-se informar seu nome, a variabilidade (opcional ou obrigatória) e seu status de ativação (✓) ou desativação (×) de acordo com cada cenário de adaptação de contexto. Dessa forma é possível visualizar o impacto das adaptações de contexto na ativação ou desativação das *features*.

5.4.4 Passo IV - Definição das Restrições de Interdependências entre PNFs e Features de Contexto

Uma vez identificada as restrições entre *features* e os cenários de adaptações de contexto, tem-se a necessidade de identificar as relações de interdependências entre as PNFs identificadas e as *features* de acordo com cada configuração de ativação e desativação das *features* do modelo de *features* em um dado cenário de adaptação de contexto. Uma vez que para cada cenário de adaptação de contexto, uma *feature* pode ser ativada ou desativada, e estas *features* podem ter diferentes impactos em uma ou mais PNFs.

Para definir as restrições de interdependências entre as PNFs e as *features*, foi adicionado um conceito de modelagem guiada a objetivos, em particular o conceito de links de contribuição (LAMSWEEERDE, 2001). Desse modo, foram atribuídas restrições de interdependências (também conhecidos como links de contribuição) para cada *features* ativada e desativada em um dado contexto. Estas *features* podem possuir quatro tipos de restrições de interdependências em relação a uma PNF:

- “++” – significa que a *feature* satisfaz completamente uma PNF se ela for ativada;
- “--” – significa que a *feature* não satisfaz completamente uma PNF se ela for ativada;
- “+” – significa que a *feature* tem uma influência positiva em relação a uma PNF se ela for ativada; e
- “-” – significa que a *feature* tem uma influência negativa em relação a uma PNF se ela for ativada.

Por exemplo, a *feature* criptografia satisfaz totalmente (++) a PNF segurança de acesso a dados, enquanto que tem uma influência negativa (-) na PNF desempenho quando está *feature* é ativada na configuração de um produto. É importante ressaltar que uma influência negativa (-) em uma PNF feita por uma *feature* opcional ativada pode ser transformada em uma influência positiva (+) se esta *feature* for desativada na sua configuração atual.

Quadro 8 – Modelo para identificação de restrições de interdependências entre *features* e PNFs em um determinado cenário de adoção de contexto.

CENÁRIO DE ADAPTAÇÃO C1			
Nome da Features	Status	Restrições de Interdependências	Propriedades Não-Funcionais
Mostrar Mapa	✓	+	Interface Amigável Alta
Mostrar Mapa	✓	-	Consumo de Energia Baixo
Mostrar Localização	✓	++	Precisão de Posição Alta
Mostrar Localização	✓	-	Consumo de Energia Baixo
Gerenciar Contexto.Acesso	✓	+	Segurança de Dados Alta
Capturar.Via Serviço Externo	✓	++	Latência Baixa
Capturar.Via Sensor	×	+	Latência Baixa
Mostrar Compatibilidade de Perfil	✓	++	Satisfação do Cliente Alta
Mostrar Documentos.Vídeo	✓	++	Interface Amigável Alta, Satisfação do Cliente Alta
Mostrar Documentos.Vídeo	✓	--	Consumo de Energia Baixo, Consumo de Memória Baixo
Mostrar Documentos.Texto	✓	++	Consumo de Energia Baixo, Consumo de Memória Baixo
Mostrar Documentos.Imagem	×	+	Consumo de Energia Baixo, Consumo de Memória Baixo
Mostrar Documentos.Imagem	×	-	Interface Amigável Alta, Satisfação do Cliente Alta

Fonte – Elaborada pelo Autor.

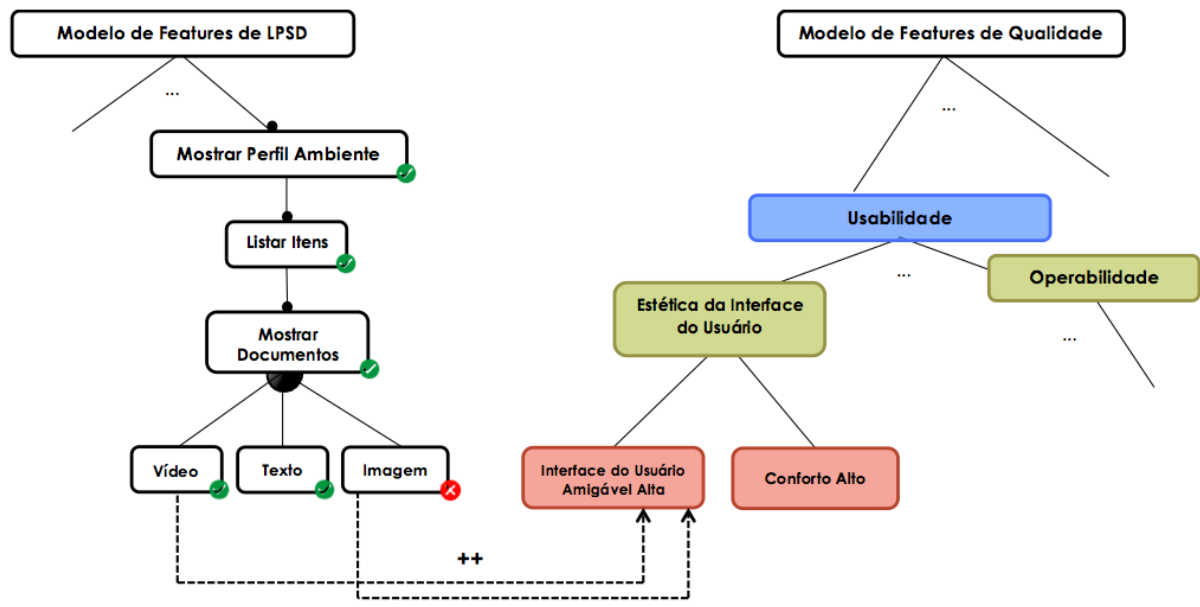
O Quadro 8, representa a identificação das restrições de interdependências entre *features* e PNFs em um determinado cenário de adaptação de contexto, desse modo para cada *features* do modelo de *features*, deve ser informado: i) nome da *feature*; ii) status de ativação

ou desativação; iii) a restrição de interdependência que esta *feature* possui; e iv) quais PNFs possuem restrições de interdependências com a *feature*.

5.4.5 Passo V - Modelagem de Restrições de Interdependências entre Features de Contexto e PNFs

Uma vez com as restrições identificadas, é possível mapeá-las para o modelo de *features* de LPSD. Este mapeamento pode ser representado por meio da associação das *features* de contexto e a PNFs, de forma que nesta associação é possível observar as restrições de interdependências entre as *features* e as PNFs (ver Figura 19).

Figura 19 – Exemplo da representação de restrições de interdependências entre *Features* de Contexto e PNFs



Fonte – Elaborada pelo Autor.

Na Figura 19 é possível observar a representação das restrições de interdependências entre as *Features* de Contexto e PNFs. A *feature* Vídeo quando está ativada possui uma restrição de satisfaz completamente (++) a PNF Interface de Usuário Amigável Alta, entretanto a *feature* Imagem quando está desativada possui um influência negativa na PNF Interface de Usuário Amigável Alta.

6 AVALIAÇÃO DA ABORDAGEM REMINDER

A abordagem ReMINDER, apresentada na Seção 5, tem o objetivo de fornecer uma forma sistemática para identificação de PNFs e cenários de adaptações de contexto, com suas respectivas restrições, para apoiar a modelagem de *features* e representação de PNFs em modelos de *features* de LPSDs. Um estudo preliminar, descrito neste Seção, foi realizado para avaliar a aplicabilidade dessa abordagem.

Esta Seção está organizada da seguinte forma. A Subseção 6.1 apresenta o objetivo do estudo; Subseção 6.2 define o estudo realizado; Subseção 6.3 descreve o estudo realizado, seu procedimento e análise dos resultados; e finalmente a Subseção 6.4 discute as considerações sobre o estudo de observação.

6.1 Objetivo

O estudo realizado para avaliar a aplicabilidade da abordagem ReMINDER é classificado como um estudo de observação, no qual o participante realiza alguma tarefa enquanto é observado por um experimentador. Esse tipo de estudo tem como finalidade coletar dados sobre como uma determinada tarefa é realizada. Desse modo, pode-se obter informações de como a abordagem é utilizada. O objetivo do estudo de observação realizado é descrito no Quadro 9, seguindo o modelo proposto por Wohlin et al. (2012).

6.2 Definição

Para a realização do experimento participaram dois alunos concludentes do curso de Engenharia de Software e de Sistemas de Informação da UFC - Campus Quixadá. Para a seleção dos participantes foi adotado o critério de terem cursado a disciplina de Reuso de Software. Cada participante tinha total liberdade para decidir de sua colaboração no experimento. O experimento foi realizado em uma única etapa, composta por três tarefas.

Quadro 9 – Descrição do objetivo do estudo de observação.

Analisar o processo definido na abordagem ReMINDER
Com a finalidade de caracterizar sua aplicação
Em relação à identificação de PNFs e cenários de adaptações de contexto, com suas respectivas restrições, para apoiar a modelagem de <i>features</i> e representação de PNFs em modelos de <i>features</i> de LPSDs
Do ponto de vista do engenheiro de domínio
No contexto de alunos de graduação, analisando o modelo de <i>features</i> de um produto sensível ao contexto em diferentes cenários de adaptações de contexto, com o uso do processo definido na abordagem ReMINDER

Fonte – Elaborada pelo Autor.

Independentemente das tarefas, inicialmente, foi aplicado um Formulário de Consentimento (Apêndice A), que descreve tópicos relacionados a participação no estudo. Uma vez assinado o formulário, cada participante confirma a participação no experimento. Em seguida, foi aplicado um Questionário para caracterização e avaliação do *background* de cada participante (Apêndice B), que avalia o nível de conhecimento do participante em tópicos relacionados a este estudo, por exemplo, linha de produtos de software dinâmicas, modelo de *features*, modelagem de aspectos dinâmicos, dentre outros.

Após o preenchimento do questionário foi apresentado uma visão geral da abordagem e foi entregue a cada participante a descrição das tarefas que deveriam ser realizadas durante o experimento (Apêndice C), juntamente com o modelo de *features* exemplo (Apêndice D).

Para realizar a primeira tarefa foi proposto um cenário de uso onde os participantes deveriam exercer o papel de um engenheiro de domínio contratado por uma empresa de desenvolvimento de software. Essa empresa possui uma LPSD para o desenvolvimento de sistemas de Guia de Visitas Móvel. Originalmente, essa linha de produtos não especifica PNFs, mas por exigências do mercado, foi necessário incluir tais propriedades. Desse modo os participantes deveriam identificar as possíveis PNFs, de acordo com a descrição do cenário de uso proposto do Guia de Visitas Móvel MobiLine (Apêndice E). Uma vez realizada a análise do cenário de uso cada participante deveria analisar cada característica e subcaracterística da norma ISO/IEC 25010 (SQuaRE) (Anexo A), em seguida identificar as PNFs com base no cenário de uso e com o auxílio do Catálogo para identificação de PNFs (Apêndice F). Uma vez identificadas as PNFs o participante deveria modelar essas PNFs de acordo com a especificação do modelo de *features* de qualidade (Apêndice F).

O resultado esperado após a realização da primeira tarefa é a identificação das seguintes características, subcaracterísticas e PNFs descritas no Quadro 10.

Quadro 10 – Resultados esperados para a primeira tarefa.

Características	Subcaracterísticas	Propriedades não-funcionais
Adequação funcional	Corretude funcional	Precisão de posição
Usabilidade	Operabilidade	Facilidade de uso
	Estética de interface do usuário	Interface de usuário amigável
Segurança	Autenticabilidade	Segurança de acesso aos dados
Eficiências de desempenho	Comportamento de tempo	Tempo de execução
		Tempo de resposta
	Capacidade	Tamanho do arquivo
	Utilização de recurso	Consumo de Energia
		Consumo de Memória

Fonte – Elaborada pelo Autor.

Na segunda tarefa, após a identificação e representação das PNFs cada participante deveria identificar quais *features* deveriam ser ativadas e desativadas com base nas regras e definições dos cenários de adaptação de contexto (Apêndice G). Ao todo foram dois cenários de adaptações de contexto. Para realizar esta tarefa os participantes deveriam preencher o *Template* de identificação de restrições de ativação e desativação de acordo com cada cenário de adaptação de contexto (Apêndice H).

Na última tarefa cada participante deveria identificar as restrições de interdependências entre as PNFs identificadas na tarefa inicial e as *features* ativadas e desativadas de acordo com cada cenário de adaptação de contexto identificadas na segunda tarefa e especificadas com o preenchimento do *Template* de identificação de restrições de ativação e desativação de acordo com cada cenário de adaptação de contexto (Apêndice H). Para realizar esta tarefa cada participante deveria especificar quais *links* de contribuições uma *feature* tem com relação a uma ou mais PNFs quando esta *feature* é ativada e desativada. As especificações das restrições de interdependências deveriam ser realizadas com o auxílio do *Template* apresentado no Apêndice I.

Uma descrição mais detalhada do modelo de *features*, do cenário de uso, das definições de adaptações de contexto e regras de contexto, e das tarefas a serem realizadas podem ser encontradas respectivamente nos Apêndices D, E, G, C.

6.3 Estudo de observação

Esta Subseção apresenta as três tarefas do estudo realizado. A Subseção 6.3.1 descreve o estudo e os principais dados a serem coletados. A Subseção 6.3.2 apresenta o procedimento adotado para a realização das tarefas. Os resultados obtidos são apresentados na

Subseção 6.3.3. Por fim, uma análise geral do estudo é realizada na Subseção 6.3.4.

6.3.1 Descrição

Na primeira tarefa do estudo, os participantes deveriam identificar as PNFs relevantes para o domínio do modelo de *features* exemplo, eles deveriam identificar estas PNFs com o auxílio do catálogo de PNFs, e representá-los visualmente de acordo com o modelo de *features* de qualidade seguindo a definição do processo da abordagem ReMINDER. Os principais objetivos dessa tarefa foram observar quais PNFs os participantes identificaram e a utilizado do Catálogo de PNFs. Os principais dados coletados nessa etapa envolveram: o conjunto de PNFs identificadas; e o tempo para a realização da tarefa.

Na segunda tarefa do estudo, os participantes deveriam identificar quais *features* deveriam ser ativadas e desativadas com base nas regras e definições dos cenários de adaptações de contexto. O principal objetivo dessa tarefa foi observar se os participantes identificaram com facilidade a ativação e desativação de *features* de acordo com os dois cenários de adaptações de contexto diferentes com o auxílio do *template* proposto para essa tarefa. Os principais dados coletados nessa etapa envolveram: os benefícios na utilização do *template*; e o tempo para a realização da tarefa.

Na terceira tarefa do estudo, os participantes deveriam identificar as restrições de interdependências entre as PNFs que foram identificadas na tarefa inicial e as *features* ativadas e desativadas de acordo com cada cenário de adaptação de contexto identificadas na segunda tarefa. O principal objetivo dessa tarefa foi observar se os participantes identificaram com facilidade as restrições de interdependências entre as PNFs e as *features* de acordo com dois cenários de adaptações de contexto diferentes com o auxílio do *template* proposto para essa tarefa. Os principais dados coletados nessa etapa envolveram: as restrições de interdependências, os benefícios na utilização do *template*; e o tempo para a realização da tarefa.

6.3.2 Procedimentos

Para cada tarefa do estudo participaram dois alunos concludentes de graduação. O estudo foi realizado com o experimentador e todos os participantes, em uma única sessão.

Como descrito na Seção 6.2, inicialmente, os participantes deveriam preencher o Formulário de Consentimento (Apêndice A) e o Questionário para caracterização e avaliação do *background* de cada participante (Apêndice B).

Em seguida, foi realizado um treinamento para os participantes de aproximadamente 15 minutos, das fases e passos utilizados na abordagem ReMINDER. Após a apresentação, foi destinado um tempo para que os participantes tirassem possíveis dúvidas sobre o conteúdo apresentado pelo experimentador.

Depois disso, foram fornecidos os seguintes documentos: Descrição das Tarefas (Apêndice C) e Descrição do Modelo Exemplo (Apêndice D). Em seguida, para a condução das tarefas em sua ordem pré-definida foi realizada a leitura de cada tarefa, e entregue aos participantes os artefatos necessários para realização de cada tarefa, respectivamente. De modo que uma tarefa só inicia com o término da tarefa anterior.

Concluída a realização das tarefas, os participantes deveriam, responder às questões do Formulário de Avaliação da Abordagem (Apêndice J), como forma de auxiliar a coleta de dados do estudo.

O tempo total para a realização da sessão foi, em média, de 1 hora e 13 minutos.

6.3.3 Resultados

O Quadro 11 apresenta um resumo da caracterização dos participantes. Nos itens da pergunta 2, os níveis de experiência variam de 1 (baixo) a 5 (excelente).

Quadro 11 – Resumo da caracterização dos participantes.

Pergunta			P1	P2
1		Formação acadêmica	Graduando	Graduando
2	2.1	LPS	4	3
	2.2	LPSD	4	3
	2.3	Modelo de <i>features</i>	4	4
	2.4	Modelagem de software	3	3
	2.5	Modelagem de aspectos dinâmicos	3	2
	2.6	Modelagem de PNFs	1	2

Fonte – Elaborada pelo Autor.

Durante a realização do estudo, foram medidos os tempos em minutos de cada participante para a realização das tarefas propostas. A contabilização desses valores são apresentados no Quadro 12 e levam em consideração apenas o intervalo entre o início até o término de cada tarefa.

Quadro 12 – Tempo de cada participante.

Participante	Tempo da tarefa 1	Tempo da tarefa 2	Tempo da tarefa 3
P1	18	15	23
P2	18	17	17

Fonte – Elaborada pelo autor

A seguir são descritos os resultados obtidos de cada participante em cada uma das três tarefas do estudo de observação.

6.3.3.1 Participante P1

De acordo com o que foi observado com a execução da primeira tarefa, o participante P1 foi capaz de identificar 6 das 9 PNFs esperadas de forma correta para o cenário de uso. Esse fato foi verificado pela resposta do participante P1 apresentada no Quadro 13.

Quadro 13 – Respostas do P1 apresentadas no modelo de *features* de qualidade.

Características	Subcaracterísticas	Propriedades não-funcionais
Usabilidade	Operabilidade	Facilidade de uso Alta
	Estética de interface do usuário	Interface de usuário amigável Normal
Segurança	Autenticabilidade	Segurança de acesso aos dados Alta
Eficiências de desempenho	Comportamento de tempo	Tempo de resposta Baixa
	Capacidade	Tamanho do arquivo Normal
Adequação funcional	Corretude funcional	Precisão de posição Alta

Fonte – Elaborada pelo autor

O participante P1 afirmou que não sentiu dificuldades na identificação das PNFs. Além disso, considerou que o processo da abordagem foi útil, pois todas as tarefas utilizaram de *templates* que auxiliaram na identificação das PNFs. O participante P1 relatou que embora tivesse um catálogo de PNFs, o esforço utilizado para identificar as PNFs foi médio, pois envolvia a consulta das descrições das características e subcaracterísticas de qualidade, em seguida a consulta ao catálogo e identificação das PNFs. Entretanto, o participante P1 relatou que não é cansativo o entendimento do catálogo de PNFs.

Em relação a execução da segunda tarefa, o participante conseguiu analisar as definições de cenários de adaptações de contexto e identificou corretamente todas as regras de contexto que deveriam ser executadas para identificar quais *features* deveriam ser ativadas e desativadas nos dois cenários de adaptações de contexto. Além disso, o participante considerou que a utilização da abordagem auxiliou na identificação destas restrições, pois o *template* utilizado para definição de cenários de adaptações de contexto permitiu que fosse possível identificar o comportamento das *features* em cada cenário. Já em relação ao *template* de identificação de restrições de ativação e de desativação, o participante disse que ele facilitou no andamento da tarefa, pois permitiu ver as regras que deveriam ser executadas em cada contexto, e como todas os nomes das *features* forma previamente listados, foi intuitivo identificar as ativações e desativações para cada cenário.

De acordo com o que foi observado com a execução da última tarefa, embora o participante não tenha identificado todas as PNFs previstas, as restrições de interdependências identificadas pelo participante nos cenários 1 e 2 foram consistentes (ver Quadros 14 e 15).

Quadro 14 – Respostas do P1 das restrições de interdependências do cenário 1

CENÁRIO DE ADAPTAÇÃO C1			
Nome da feature	Status	Restrições de Interdependências	Propriedades Não-Funcionais
Mostrar Mapa	✓	+	Interface de usuário amigável Normal
Mostrar Localização	×	-	Interface de usuário amigável Normal
		-	Facilidade de uso Alta
Gerenciar Contexto.Persistência	✓	+	Tamanho do arquivo Normal
Gerenciar Contexto.Acesso	✓	++	Tempo de resposta Baixo
		++	Precisão de posição Baixa
Gerenciar Contexto.Aquisição	✓	++	Tempo de resposta Baixo
		++	Precisão de posição Alta
Capturar.Via Serviço Externo	×		
Capturar.Via Sensor	✓	++	Tempo de resposta Baixo
		++	Precisão de posição Alta
Mostrar Compatibilidade de Perfil	✓	-	Tempo de resposta Baixo
Mostrar Perfil do Ambiente	✓	++	Interface de usuário amigável Normal
Listar Itens	✓	++	Facilidade de uso Alta
		+	Interface de usuário amigável Normal
Mostrar Documentos.Vídeo	✓	-	Tamanho do arquivo Normal
Mostrar Documentos.Texto	✓	++	Tamanho do arquivo Normal
Mostrar Documentos.Imagem	✓	++	Tamanho do arquivo Normal

Fonte – Elaborada pelo autor

Quadro 15 – Respostas do P1 das restrições de interdependências do cenário 2

CENÁRIO DE ADAPTAÇÃO C2			
Nome da feature	Status	Restrições de Interdependências	Propriedades Não-Funcionais
Mostrar Mapa	✓	+	Interface de usuário amigável Normal
		+	Facilidade de uso Alta
Mostrar Localização	✓	+	Interface de usuário amigável Normal
		+	Facilidade de uso Alta
Gerenciar Contexto.Persistência	×	+	Tamanho do arquivo Normal
Gerenciar Contexto.Acesso	✓	++	Tempo de resposta Baixo
		++	Precisão de posição Alta
Gerenciar Contexto.Aquisição	✓	++	Tempo de resposta Baixo
		++	Precisão de posição Alta
Capturar.Via Serviço Externo	×		
Capturar.Via Sensor	✓	++	Tempo de resposta Baixo
		++	Precisão de posição Alta
Mostrar Compatibilidade de Perfil	×	+	Tempo de resposta Baixo
Mostrar Perfil do Ambiente	✓	++	Interface de usuário amigável Normal
Listar Itens	✓	+	Facilidade de uso Alta
		+	Interface de usuário amigável Normal
Mostrar Documentos.Vídeo	×	++	Tamanho do arquivo Normal
Mostrar Documentos.Texto	✓	++	Tamanho do arquivo Normal
Mostrar Documentos.Imagem	✓	++	Tamanho do arquivo Normal

Fonte – Elaborada pelo autor

Segundo o participante P1 uma melhoria na abordagem seria a padronização do nome da variabilidade das *features*, por meio de siglas para utilização dos *templates* (Apêndices H e I), pois tomaria menos tempo e esforço. Segundo o participante P1 embora a abordagem permita identificar as PNFs e sua relação com o comportamento do modelo de *features*, o grau de esforço

para identificar essas restrições é médio, pois o participante relatou que sentiu dificuldades sobre a funcionalidade de algumas *features* do modelo de *features* exemplo. O participante relatou que durante a utilização da abordagem ele percebeu que deixou de identificar PNFs relevantes, entretanto para as PNFs que ele identificou o esforço para especificar as restrições foi baixo.

6.3.3.2 Participante P2

De acordo com o que foi observado com a execução da primeira tarefa, o participante foi capaz de identificar 3 das 9 PNFs de forma correta para o cenário de uso. Esse fato foi verificado pela resposta do participante P2 apresentada no Quadro 16.

Quadro 16 – Respostas do P2 apresentadas no modelo de *features* de qualidade

Características	Subcaracterísticas	Propriedades não-funcionais
Segurança	Autenticabilidade	Segurança de acesso aos dados Alta
Eficiências de desempenho	Comportamento de tempo	Taxa de transferência Alta
		Tempo de execução Baixa
Adequação funcional	Corretude funcional	Precisão de posição Alta

Fonte – Elaborada pelo autor

O participante afirmou que não sentiu dificuldades na identificação das PNFs. Além disso, considerou que o processo da abordagem foi útil para a identificação das PNFs e destacou que o catálogo de PNFs foi de fácil entendimento e auxiliou na identificação das PNFs de acordo com as características e subcaracterísticas de qualidade da norma SQuARE, resultando em um baixo esforço para identificação das PNFs.

Em relação a execução da segunda tarefa, o participante conseguiu analisar as definições de cenários de adaptações de contexto e identificou corretamente todas as regras de contexto que deveriam ser executadas para identificar quais *features* deveriam ser ativadas e desativadas nos dois cenários de adaptações de contexto. Além disso, considerou que a utilização da abordagem auxiliou na identificação destas restrições, pois o *template* utilizado para definição de cenários de adaptações de contexto foi de fácil entendimento, auxiliando na visualização de quais informações de contextos são consideradas em cada cenário. Já em relação ao *template* de identificação de restrições de ativação e de desativação, o participante disse que ele facilitou no andamento da tarefa e que ele é importante pois as informações preenchidas nesse *template* são reutilizadas nas próximas tarefas.

De acordo com o que foi observado com a execução da última tarefa, embora o participante não tenha identificado todas as PNFs previstas, as restrições de interdependências

identificadas pelo participante no cenário 1 e 2 foram consistentes (ver Quadros 17 e 18).

Quadro 17 – Respostas do P2 das restrições de interdependências do cenário 1.

CENÁRIO DE ADAPTAÇÃO C1			
Nome da feature	Status	Restrições de Interdependências	Propriedades Não-Funcionais
Mostrar Mapa	✓	-	Tempo de execução Baixa
Mostrar Localização	×		
Gerenciar Contexto.Persistência	✓		
Gerenciar Contexto.Acesso	✓		
Gerenciar Contexto.Aquisição	✓		
Capturar.Via Serviço Externo	×	++	Segurança de acesso aos dados Alta
Capturar.Via Sensor	✓	--	Segurança de acesso aos dados Alta
Mostrar Compatibilidade de Perfil	✓		
Mostrar Perfil do Ambiente	✓		
Listar Itens	✓		
Mostrar Documentos.Video	✓	-	Tempo de execução Baixa
		--	Taxa de transferência Alta
Mostrar Documentos.Texto	✓	++	Tempo de execução Baixa
Mostrar Documentos.Imagem	✓	+	Tempo de execução Baixa

Quadro 18 – Respostas do P2 das restrições de interdependências do cenário 2.

CENÁRIO DE ADAPTAÇÃO C2			
Nome da feature	Status	Restrições de Interdependências	Propriedades Não-Funcionais
Mostrar Mapa	✓	-	Tempo de execução Baixa
Mostrar Localização	✓	++	Precisão de posição Alta
Gerenciar Contexto.Persistência	×		
Gerenciar Contexto.Acesso	✓		
Gerenciar Contexto.Aquisição	✓		
Capturar.Via Serviço Externo	×	++	Segurança de acesso aos dados Alta
Capturar.Via Sensor	✓	--	Segurança de acesso aos dados Alta
Mostrar Compatibilidade de Perfil	×		
Mostrar Perfil do Ambiente	✓		
Listar Itens	✓		
Mostrar Documentos.Video	×	++	Tempo de execução Baixa
		++	Taxa de transferência Alta
Mostrar Documentos.Texto	✓	++	Tempo de execução Baixa
Mostrar Documentos.Imagem	✓	+	Tempo de execução Baixa

O participante não apresentou nenhuma sugestão de melhoria para o processo. Segundo o participante P2 embora a abordagem auxilie na identificação das restrições de interdependências, o grau de esforço para identificar essas restrições é médio, pois é um pouco complicado pra quem não tem muita prática, relacionar as interdependências na forma de *links* de contribuição, pois em alguns casos dificultam a identificação das restrições.

6.3.4 Análise geral

Conforme o Quadro 11, é possível perceber que os participantes deste estudo de observação possuíam experiência nos tópicos envolvidos no estudo, uma vez que foi adotado o critério dos participantes terem cursado a disciplina de Reuso de Software.

Em relação aos tempos para realização das tarefas, apresentados no Quadro 12, a

variação máxima entre dois participantes foi de 7 minutos. O participante P1 demorou mais tempo para a realização da terceira tarefa, o que pode ser explicado pelo fato de ter sido o único participante que afirmou ter sentido dificuldades nas funcionalidades de algumas *features* e por ter menos experiência na modelagem de PNFs.

No geral, todos os participantes responderam que a abordagem auxiliou na realização das tarefas e que não tiveram um esforço alto para realização das tarefas. Além disso foi possível que abordagem conseguiu realmente cumprir com o que propõe, pois é de fácil aprendizagem e fácil de se aplicar. Em relação aos artefatos criados todos os participantes concordaram que eles foram úteis para a realização das tarefas. A desvantagem é que se a abordagem for realizada sem uma ferramenta para automatizar o seu processo, a utilização da abordagem pode ser demorada e cansativa. A principal vantagem é que a abordagem permite identificar as PNFs e sua relação com o comportamento do modelo de *features*, o que é uma coisa não trivial.

6.4 Considerações Finais

Nesta Seção, foi apresentado o estudo de observação para avaliação da abordagem ReMINDER, que tem como objetivo fornecer uma forma sistemática para identificação de PNFs e cenários de adaptações de contexto, com suas respectivas restrições, para apoiar a modelagem de *features* e representação de PNFs em modelos de *features* de LPSDs.

A partir da análise dos resultados obtidos nesse estudo, foi possível perceber que a abordagem foi bem avaliada e que a execução de seus passos não exigiu um esforço alto. No entanto, esses resultados são limitados, uma vez que o número de participantes foi reduzido.

Comparando os resultados obtidos dos dois participantes, em relação à identificação das PNFs (Quadro 10), podemos perceber que o número de PNFs encontradas pelo P1 foi maior que o do P2. Se fizermos uma média da quantidade de PNFs encontradas, temos que o P1 encontrou aproximadamente 66% das PNFs esperadas, enquanto que o P2 encontrou aproximadamente 33%. Esses resultados mostram que é possível identificar e especificar PNFs por meio da utilização da abordagem.

No geral todos os participantes responderam que a abordagem auxiliou na identificação das PNFs e a sua relação com o comportamento do modelo de *features* por meio de restrições de interdependências. No entanto, estudos mais específicos poderiam ser realizados para uma comprovação mais forte deste fato.

7 EXTENSÃO DA FERRAMENTA DYMMER E TESTE DE USABILIDADE

Na Seção anterior a abordagem REMINDER foi apresentada e exemplificada. A fim de suportar a operacionalização da abordagem foi desenvolvida uma extensão da ferramenta DyMMer (UCHÔA et al., 2017), a qual é apresentada nesta Seção.

Para realizar a extensão da ferramenta foram definidos um conjunto de requisitos funcionais. Após a definição desse requisitos, foi iniciado o desenvolvimento da extensão da ferramenta. Em seguida, foi realizado um teste de usabilidade, afim de avaliar a extensão da ferramenta proposta neste trabalho.

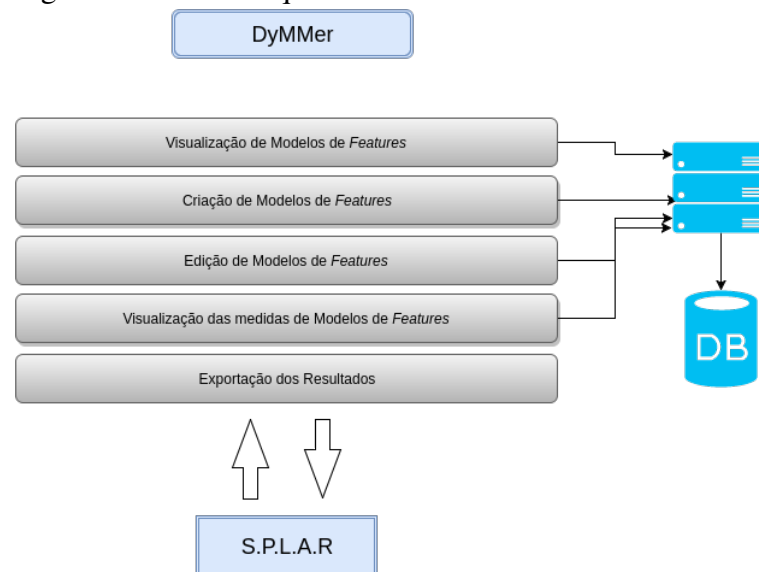
Esta Seção é organizada da seguinte forma. Na Subseção 7.1 é apresentada uma visão geral da ferramenta, estrutura em camadas, em pacotes e divisão de classes, após a extensão realizada. Na Subseção 7.2 é apresentado os requisitos funcionais. Na Subseção 7.3 são apresentados as extensões realizados nos módulos de edição e visualização da DyMMer. Finalmente, na Subseção 7.4 é apresentado o teste de usabilidade.

7.1 Visão geral da ferramenta

7.1.1 *Estrutura em camadas*

O processo de desenvolvimento da ferramenta teve como base a arquitetura já definida da DyMMer em sua versão inicial. No entanto, para suportar as extensões adicionadas a ferramenta foram adicionados um conjunto de funcionalidades nos modelos de edição e visualização do modelo de *features*. Uma visão arquitetural em camadas da ferramenta é ilustrada na Figura 20.

Figura 20 – Visão arquitetural em camadas da ferramenta



Fonte: Elaborada pelo autor

A ferramenta está dividida em cinco módulos, são eles:

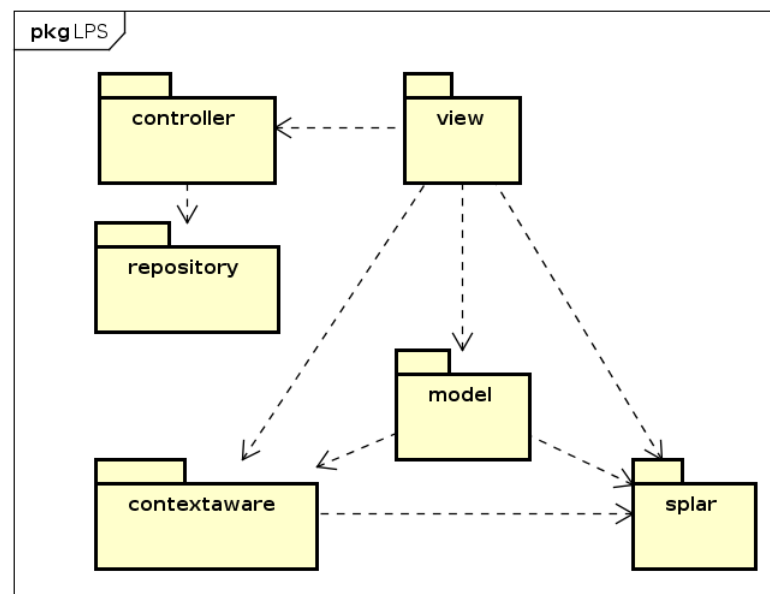
- **Visualização de modelos de *Features*** - permite a visualização de modelos de *features*, de acordo com os cenários de adaptações de contexto, podem ser visualizados os modelos que se encontram no repositório da DyMMer ou modelos armazenados no diretório de arquivos local do usuário;
- **Criação de Modelos de *Features*** - permite criar novos modelos de *features* de LPS, que podem ser adicionados ao repositório ou no diretório de arquivos local do usuário;
- **Edição de Modelos de *Features*** - permite a edição de modelos de *features* de LPS e LPSD, a adição do modelo de *features* de qualidade para representação das PNFs, especificação das restrições e dos múltiplos cenários de adaptações de contexto. Os modelos em edição podem ser armazenados no repositório da DyMMer ou no diretório de arquivos local do usuário;
- **Visualização das Medidas de Modelos de *Features*** - permite a visualização de um conjunto de 40 medidas que avaliam a qualidade do modelo de *features* de LPS e LPSD;
- **Exportação dos resultados** - permite a exportação dos resultados das medidas aplicadas aos modelos de LPS e de LPSD no formato *Excel*; e
- **Camada S.P.L.A.R** - representa o core da aplicação, onde a partir dessa biblioteca é possível realizar a leitura de arquivos que representam o modelo de

features e a estruturação do mesmo em artefatos menores que representam cada objeto, seja *feature* ou restrição de integridade do modelo.

7.1.2 Divisão de pacotes

Uma visão da divisão de pacotes da ferramenta é ilustrada na Figura 21.

Figura 21 – Diagrama de pacotes.



Fonte: Elaborada pelo autor

- **Pacote *View*:** Responsável por receber as entradas realizadas pelo usuário e exibir diferentes dados de acordo com a ação recebida;
- **Pacote *Model*:** Define a estrutura do modelo de *features* para diferentes tipos de ferramentas, como por exemplo, FAMILIAR e SPLOT;
- **Pacote *Splar*:** Responsável por conter os arquivos da biblioteca S.P.L.A.R.
- **Pacote *Controller*:** Responsável pelo controle da exportação da DyMMer, e manipulação de XMLs; e
- **Pacote *repository*:** Responsável pela comunicação realizada com a API da DyMMer.

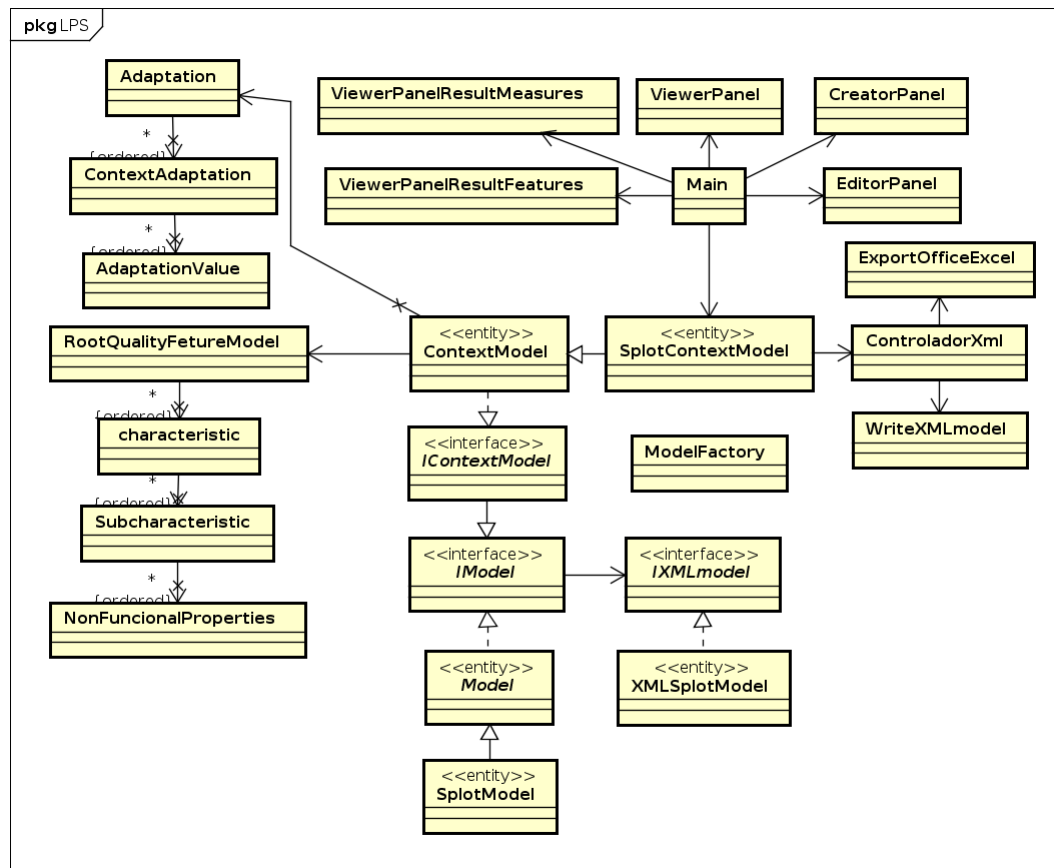
7.1.3 Diagrama de Classes

Figura 22 ilustra o diagrama de classes da ferramenta. Este diagrama representa os componentes da ferramenta, guia o desenvolvimento dos requisitos funcionais da aplicação

conforme a estrutura apresentada na Subseção 7.2.

Comunicação das classes para a aplicação desenvolvida.

Figura 22 – Diagrama de classes.



Fonte – Elaborada pelo autor

7.2 Requisitos funcionais

Primeiramente foram definidos os requisitos funcionais da extensão da ferramenta. Estes requisitos foram baseados nas necessidades de representar PNFs e múltiplos cenários de adaptações de contexto para o modelo de *features* de LPSDs. Segue as funcionalidades da aplicação:

- Criação do modelo de *features* de qualidade, para representar as PNFs;
- Adição de restrições de interdependências entre *features* e PNFs em forma de *links* de contribuições;
- Adição de múltiplos cenários de adaptações de contexto em um modelo de *features* de LPSD;
- Adição de restrições de integridade de contexto em um dado cenário de adaptação

de contexto; e

- Visualização do modelo de *features* conforme um ou mais cenários de adaptações de contexto, também exibindo a visualização das restrições.

7.3 Representação das extensões propostas na ferramenta DyMMer

Esta Seção descreve como cada uma das extensões propostas é representada na ferramenta DyMMer, incluindo a adição do modelo de *features* de qualidade (Subseção 7.3.1), as definições de cenários de adaptações de contexto (Subseção 7.3.2), adição de restrições de interdependências entre *features* e PNFs (Subseção 7.3.3), adição de múltiplos cenários de adaptações de contexto (Subseção 7.3.4) e finalmente a visualização da configuração do modelo conforme um ou mais cenários de adaptações de contexto (Subseção 7.3.5).

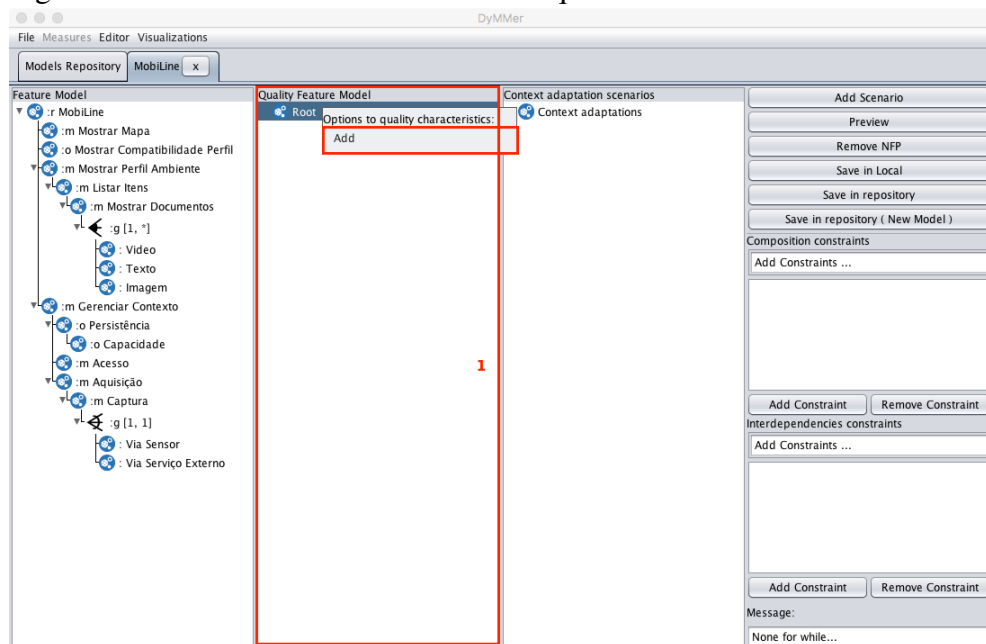
7.3.1 Modelo de *features* de qualidade

Na DyMMer, a adição do modelo de *features* de qualidade é representado por meio de uma lista hierarquizada composta por três níveis de especificação. O detalhamento do modelo de *features* de qualidade pode ser realizado por meio da adição das características e subcaracterísticas de qualidade e as PNFs que devem ser atendidas para o domínio da linha de produtos em análise. Estas características, subcaracterísticas e PNFs são mapeados de acordo com o catálogo de PNFs utilizado nesta abordagem.

7.3.1.1 Adicionando característica de qualidade

A Figura 23, apresenta o editor do modelo de *features* de qualidade. Esta área é o local onde o modelo de *features* de qualidade é especificado, representado pelo número 1. Para adicionar uma característica de qualidade é necessário selecionar a opção “Add” representado pelo menu de seleção em destaque na Figura 23.

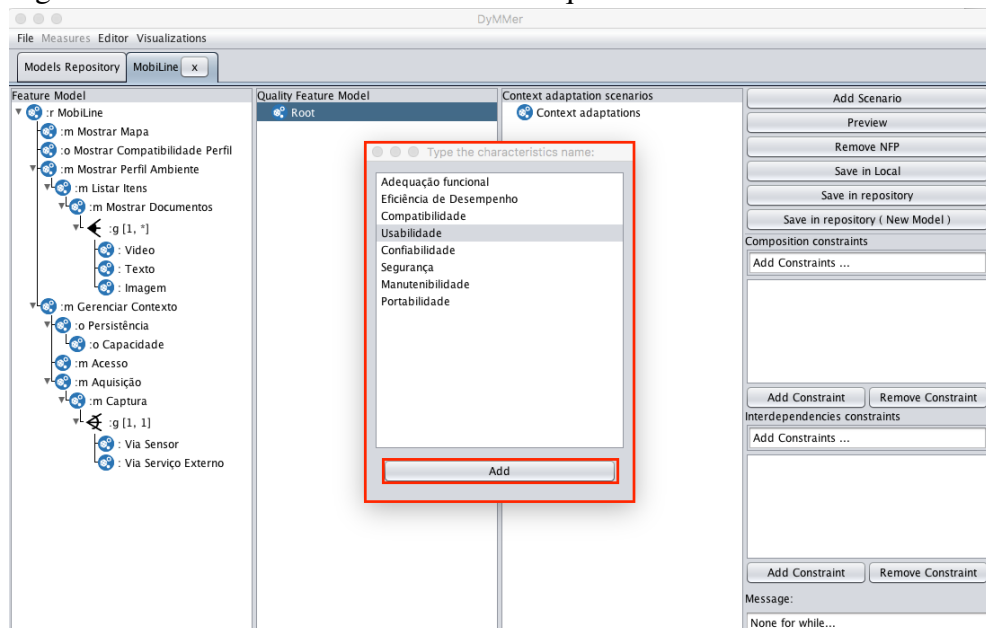
Figura 23 – Adicionando característica de qualidade



Fonte – Elaborada pelo Autor.

Após selecionar a opção “Add”, uma painel de seleção é apresentado (ver Figura 24). Este painel de seleção lista cada característica de qualidade do catálogo de PNFs. Para adicionar uma característica de qualidade é necessário selecionar uma das características apresentada e selecionar a opção “Add”.

Figura 24 – Selecionando característica de qualidade

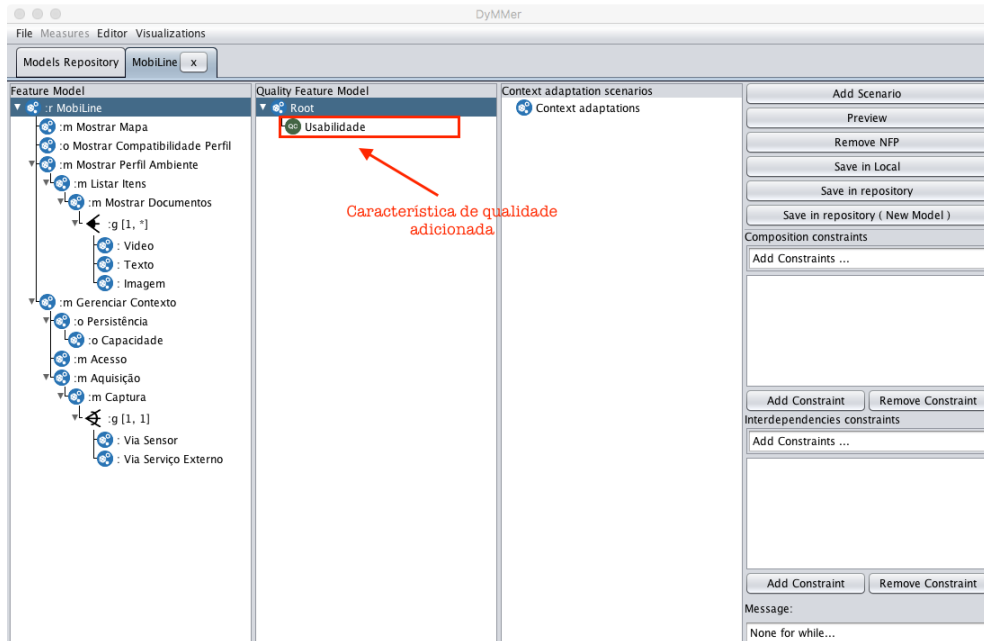


Fonte – Elaborada pelo Autor.

Uma vez adicionada a característica de qualidade, é possível visualizá-la (ver Figura

25). Desse modo, cada característica de qualidade adicionada é apresentada no modelo de *features* de qualidade.

Figura 25 – Característica de qualidade adicionada

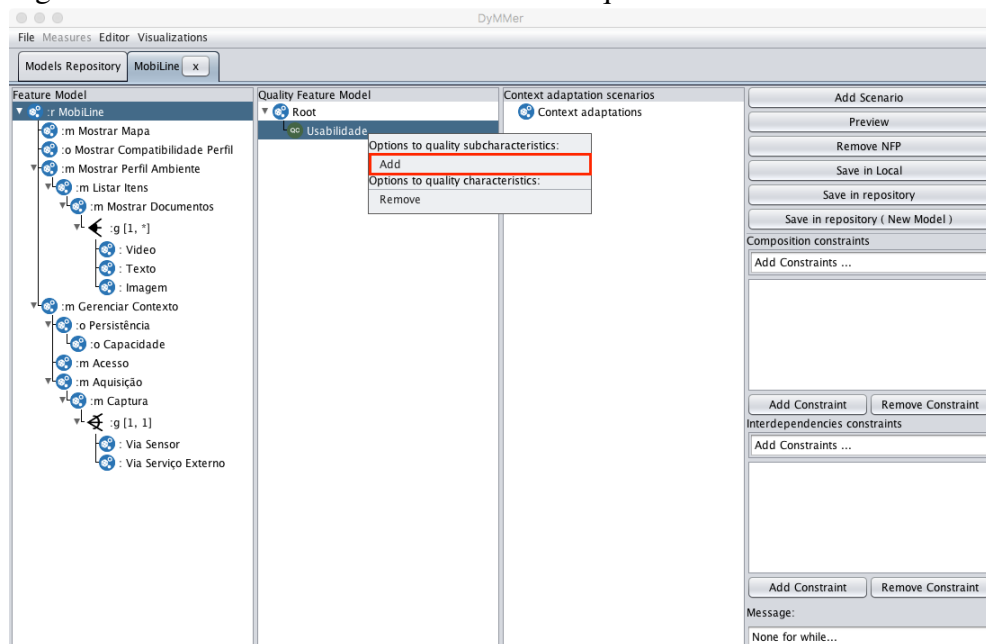


Fonte – Elaborada pelo Autor.

7.3.1.2 Adicionando subcaracterística de qualidade

Uma vez adicionada uma característica de qualidade ao modelo de *features* de qualidade, pode-se adicionar uma ou mais subcaracterísticas de qualidade relacionadas de acordo com o catálogo de PNFs. Para adicionar uma subcaracterística é necessário selecionar a opção “Add” representado pelo menu de seleção (ver Figura 26).

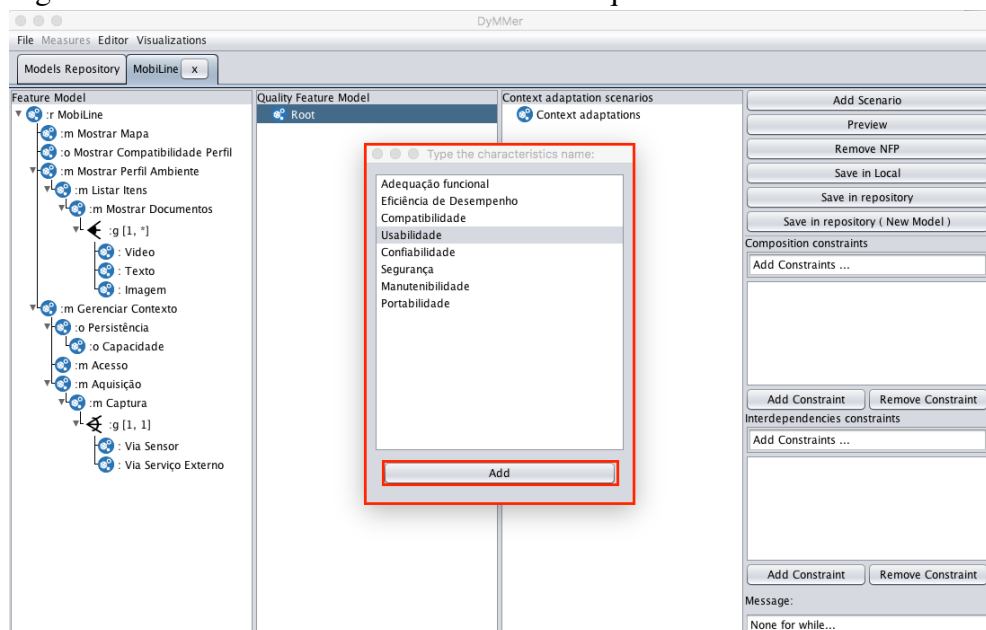
Figura 26 – Adicionando subcaracterística de qualidade



Fonte – Elaborada pelo Autor.

Após selecionar a opção “Add”, uma painel de seleção é apresentado (ver Figura 27). Este painel de seleção lista cada subcaracterística de qualidade relacionada a característica de qualidade que esta sendo especificada. Para adicionar uma subcaracterística de qualidade é necessário selecionar uma das subcaracterísticas apresentadas e selecionar a opção “Add”.

Figura 27 – Selecionando subcaracterística de qualidade

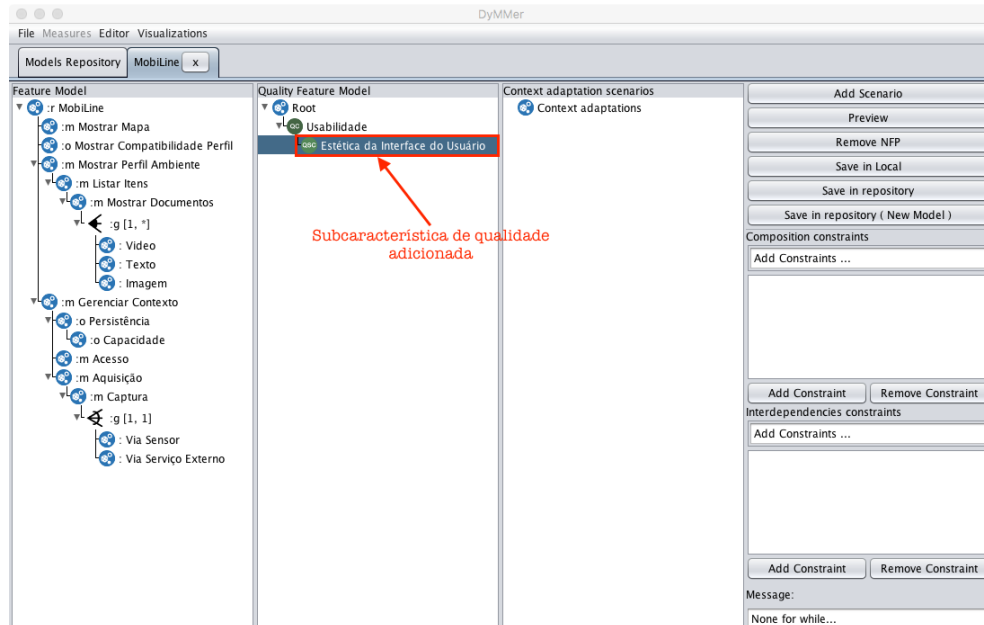


Fonte – Elaborada pelo Autor.

Uma vez adicionada a subcaracterística de qualidade, é possível visualizá-la (ver

Figura 28). Desse modo, cada subcaracterística de qualidade adicionada é apresentada no modelo de *features* de qualidade.

Figura 28 – Subcaracterística de qualidade adicionada

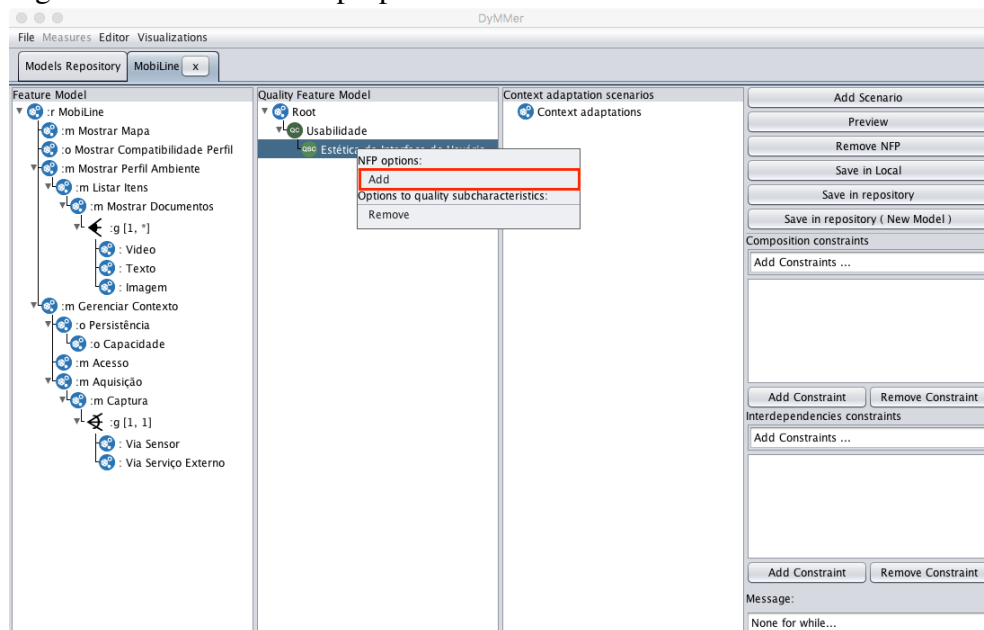


Fonte – Elaborada pelo Autor.

7.3.1.3 Adicionando propriedades não-funcionais

Uma vez adicionada uma subcaracterística de qualidade ao modelo de *features* de qualidade, pode-se adicionar uma ou mais propriedades não-funcionais relacionadas a subcaracterística adicionada. Estas propriedades não-funcionais são especificadas de acordo com o catálogo de PNFs. Para adicionar uma propriedade não-funcional é necessário selecionar a opção “Add” representado pelo menu de seleção (ver Figura 29).

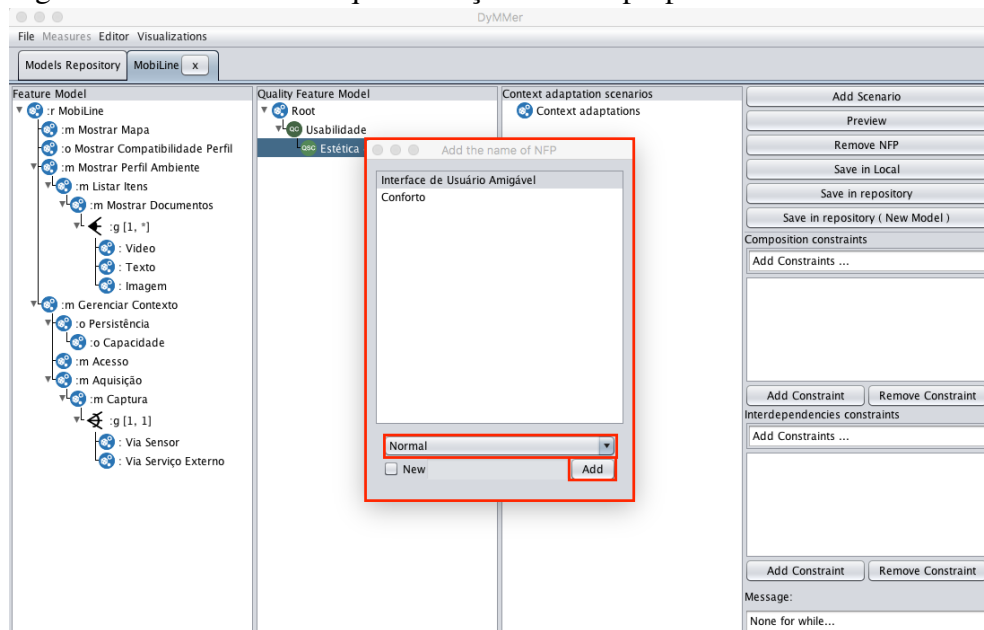
Figura 29 – Adicionando propriedades não-funcionais



Fonte – Elaborada pelo Autor.

Após selecionar a opção “Add”, uma painel de seleção é apresentado (ver Figura 30). Este painel de seleção lista as possíveis propriedades não-funcionais que característica de qualidade do catálogo de PNFs. Para adicionar uma propriedade não-funcional é necessário selecionar uma das propriedades não-funcionais apresentadas, em seguida especificar a sua escala de quantificação. A escala de quantificação é especificada ao selecionar um dos valores da lista de seleção (ver Figura 30). Estes valores variam entre, normal, alto, médio ou baixo. Por fim a opção “Add” deve ser selecionada.

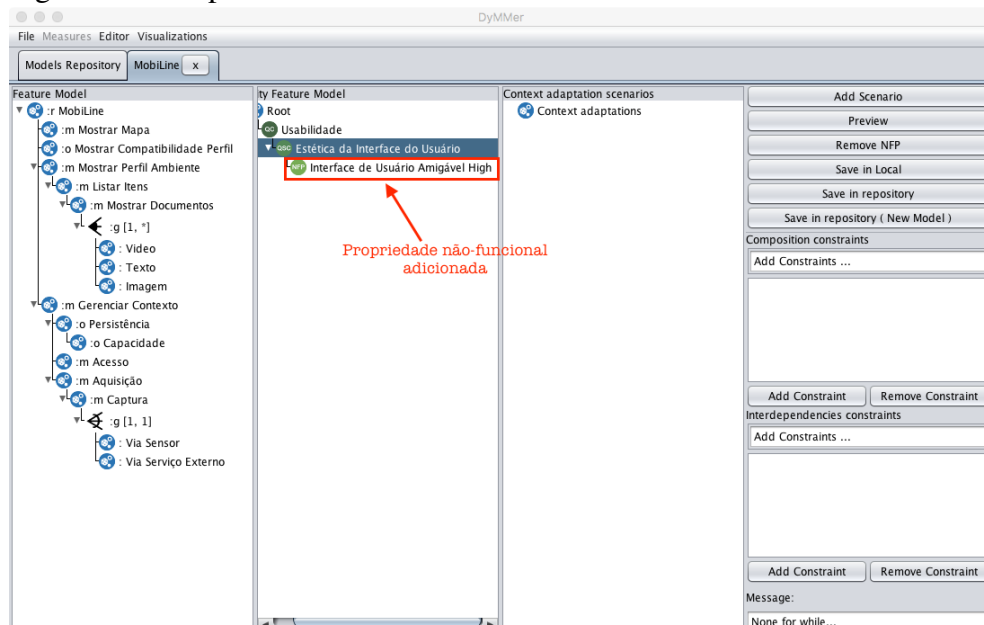
Figura 30 – Selecionando quantificação de uma propriedade não-funcional



Fonte – Elaborada pelo Autor.

Uma vez adicionado a propriedade não-funcional, é possível visualizá-la (ver Figura 31). Desse modo, cada propriedade não-funcional adicionada é apresentada no modelo de *features* de qualidade.

Figura 31 – Propriedade não-funcional adicionada



Fonte – Elaborada pelo Autor.

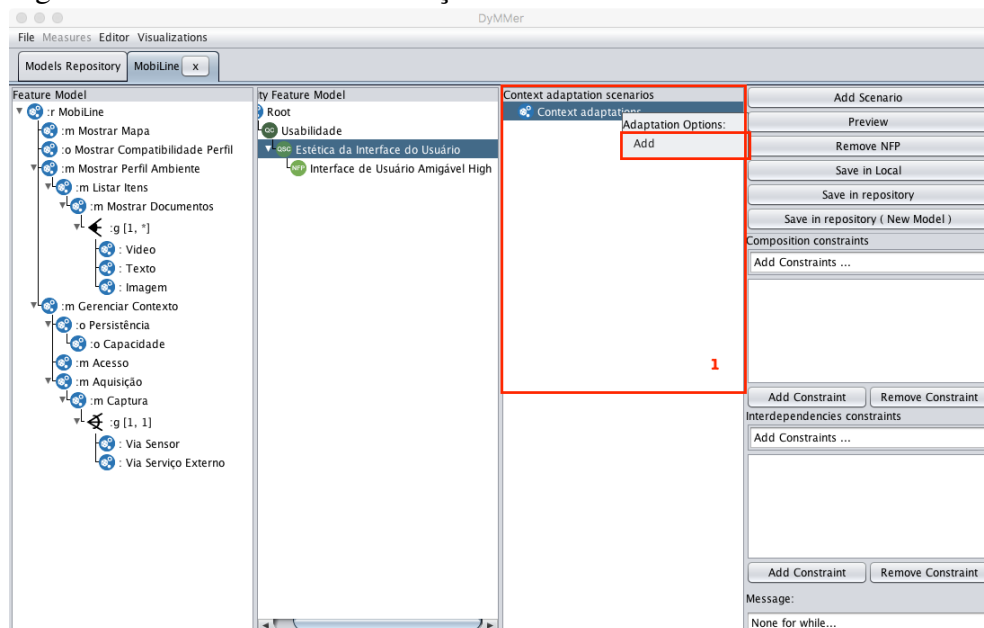
7.3.2 Definições de cenários de adaptações de contexto

Na DyMMer, a adição de cenários de adaptações de contexto é representado por meio de uma lista hierarquizada composta por dois níveis de especificação. O detalhamento dos cenários de adaptações de contexto podem ser realizados por meio da adição de informações de contexto, especificações das variações de uma informação de contexto e especificação de integridade de contexto. Estas variações representam as possíveis variações de uma informação de contexto em um determinado cenário de adaptação.

7.3.2.1 Adicionando informação de contexto

A Figura 32, apresenta o editor dos múltiplos cenários de adaptações de contexto. Esta área é o local onde os cenários de adaptações de contexto são especificados, representado pelo número 1. Para adicionar uma informação de contexto é necessário selecionar a opção “Add” representado pelo menu de seleção em destaque na Figura 32.

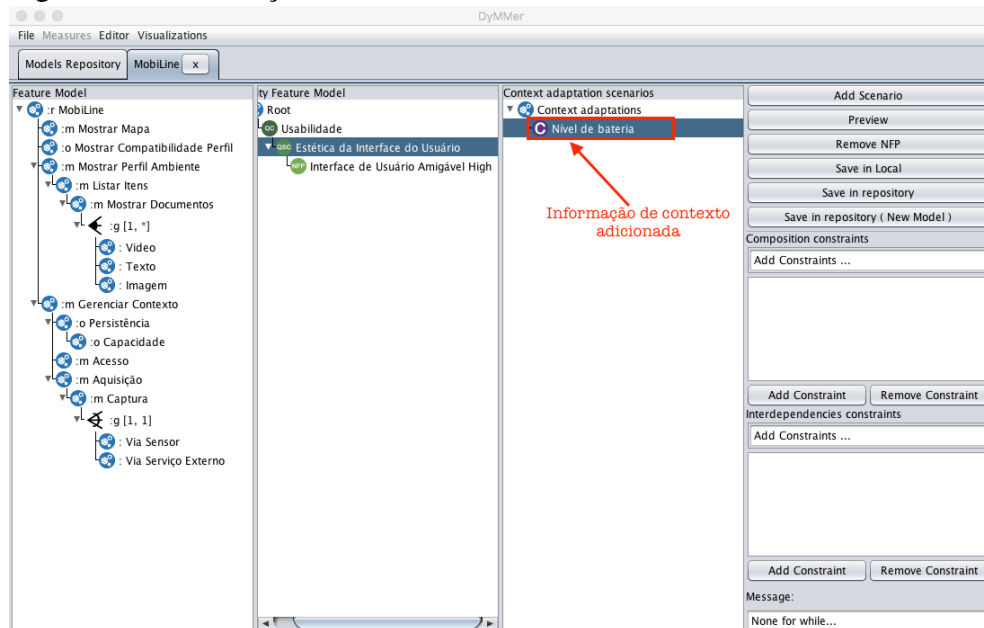
Figura 32 – Adicionando informação de contexto



Fonte – Elaborada pelo Autor.

Uma vez adicionada uma informação de contexto, é possível visualizá-la (ver Figura 33).

Figura 33 – Informação de contexto adicionada



Fonte – Elaborada pelo Autor.

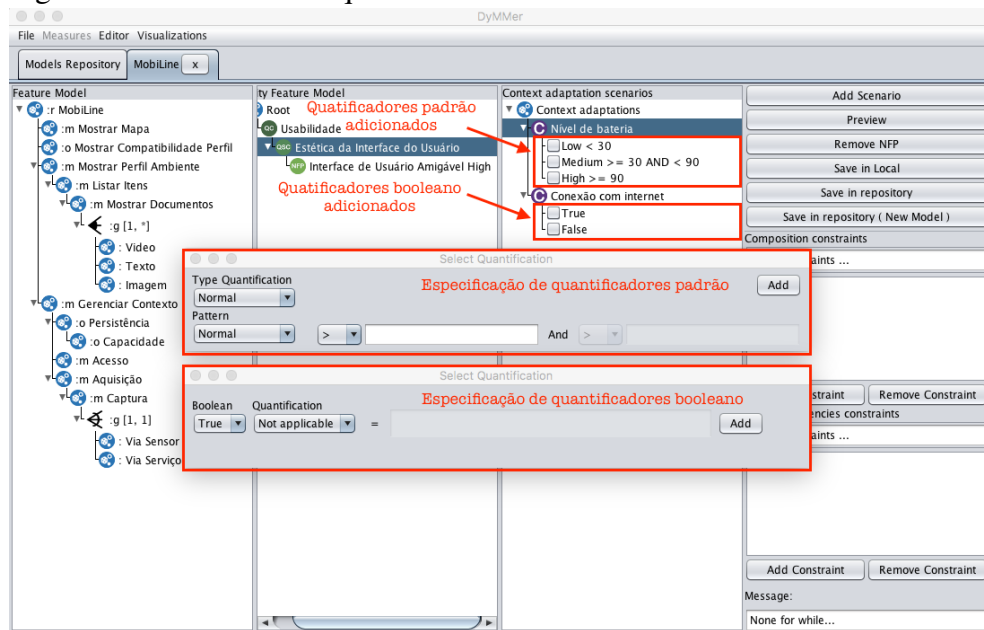
7.3.2.2 Especificando variações de adaptação

Uma vez adicionada uma informação de contexto, pode-se especificar as suas variações. Estas variações podem ser especificadas por meio de quantificadores. Um quantificador possui duas variações:

- Quantificador padrão: formado por um tipo de qualificação (Normal, Baixo, Médio, Alto ou Não se aplica); e um quantificador. Este quantificador é definido por operadores relacionais: maior que ($>$), menor que ($<$), maior ou igual que ($>=$), menor ou igual que ($<=$), igual ($=$) e diferente ($<>$). Seguidos por um valor do tipo: *string*, inteiro, *float*. Caso este quantificador for definido por um intervalo numérico, adiciona-se um operador lógico que pode ser do tipo: OU (OR) ou E (AND), seguidos por outro operador relacional e um valor do tipo *string*, inteiro, *float*; e
- Quantificador booleano: formado por um tipo booleano *true* ou *false* e opcionalmente um valor numérico.

Uma vez especificado as variações de uma informação de contexto, é necessário seleccionar a opção “Add” (ver Figura 34).

Figura 34 – Adicionando quantificadores

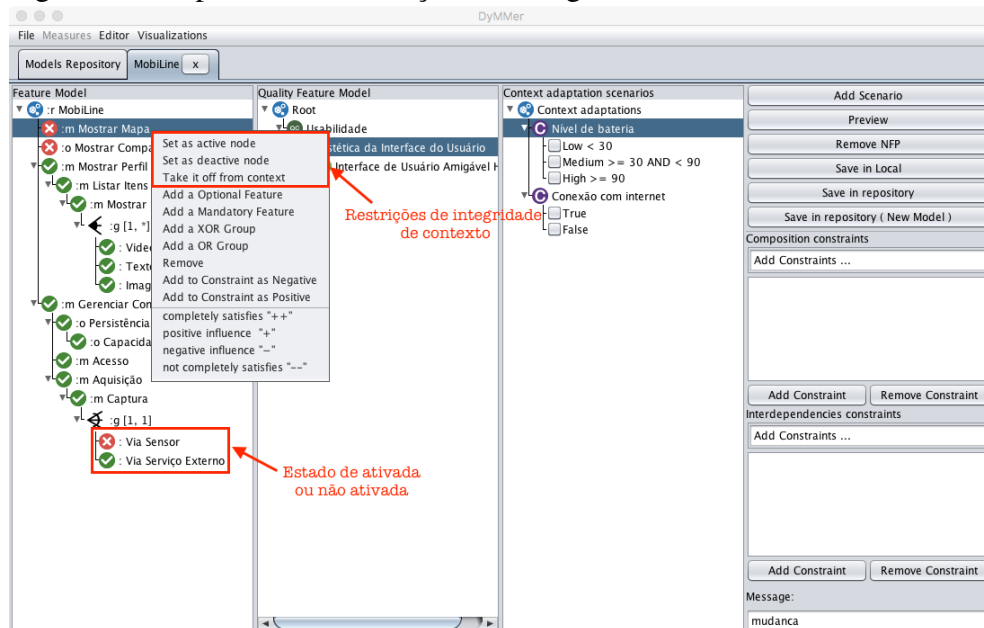


Fonte – Elaborada pelo Autor.

7.3.2.3 Especificando restrições de integridade de contexto

Para realizar a especificação da integridade de contexto em cada cenário de adaptação, devem ser selecionadas uma ou mais informações de contexto. De modo que o conjunto de informações de contexto selecionadas definem o cenário de adaptação de contexto em edição. Após a definição do cenário de adaptação, as restrições de integridade de contexto, são especificadas, dada a seleção de uma determinada *feature* alterando o seu estado de ativada ou não ativada, conforme ilustrado na Figura 35. As restrições de integridade de contexto que estão sendo criadas podem ser visualizadas durante a especificação das restrições.

Figura 35 – Especificando restrições de integridade de contexto



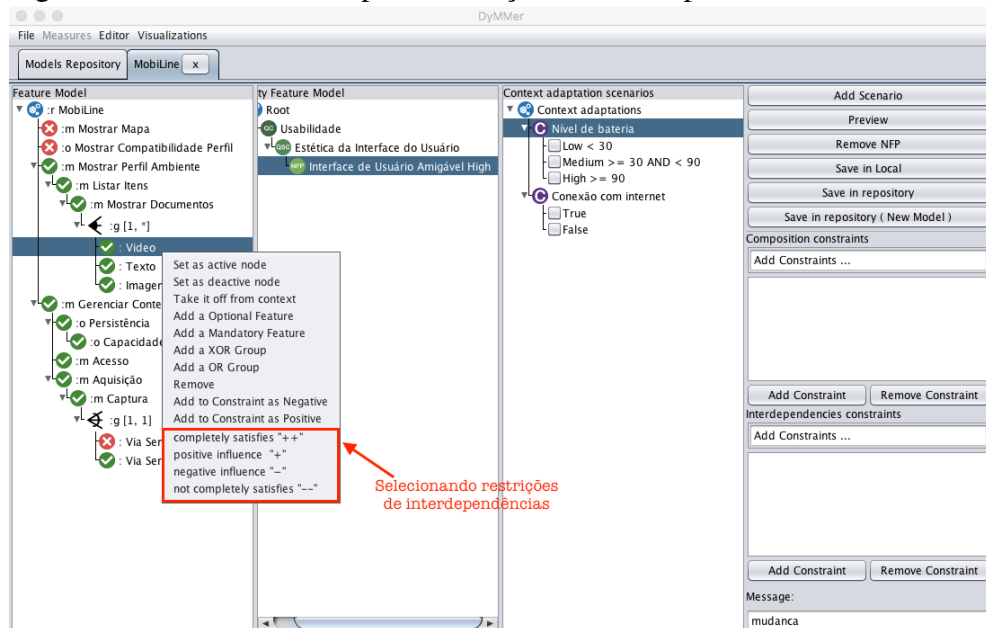
Fonte – Elaborada pelo Autor.

7.3.3 Restrições de interdependências entre features e PNFs

Na DyMMer, a especificação de restrições de interdependências entre *features* e PNFs em um determinado cenário de adaptação é representado por meio de *links* de contribuições (ver Figura 36). Por meio desses *links* de contribuições é possível especificar as interdependências entre *features* e PNFs da seguinte forma: “++”, significa a *feature* satisfaz completamente uma PNF se ela for ativada; ii) “-”, significa que a *feature* não satisfaz completamente uma PNF se ela for ativada; iii) “+”, significa que a *feature* tem uma influência positiva em relação a uma PNF se ela for ativada; e iv) “-”, significa que a *feature* tem uma influência negativa em relação a uma PNF se ela for ativada.

Para realizar a especificação de uma restrição é necessário selecionar uma determinada *feature* ativada ou desativada, em seguida, selecionar uma das possíveis restrições de interdependências apresentadas por meio de menu de seleção (ver Figura 36).

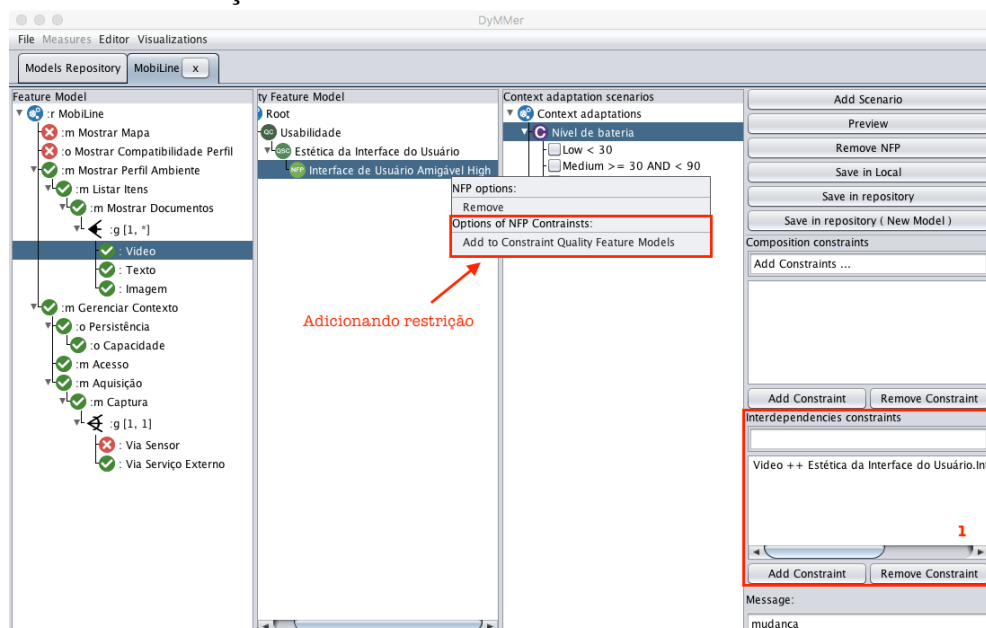
Figura 36 – Selecionando tipo de restrição de interdependência



Fonte – Elaborada pelo Autor.

Após a seleção do tipo de restrição, deve-se selecionar um das propriedades não-funcionais do modelo de *features* de qualidade, selecionando a opção de adicionar restrição ao modelo de *features* de qualidade. A Figura 37, apresenta o editor de restrições de interdependências. Esta área é o local onde as restrições de interdependências são efetivamente adicionadas ou removidas, representado pelo número 1.

Figura 37 – Selecionando PNF impactada pela restrição e adicionando restrição

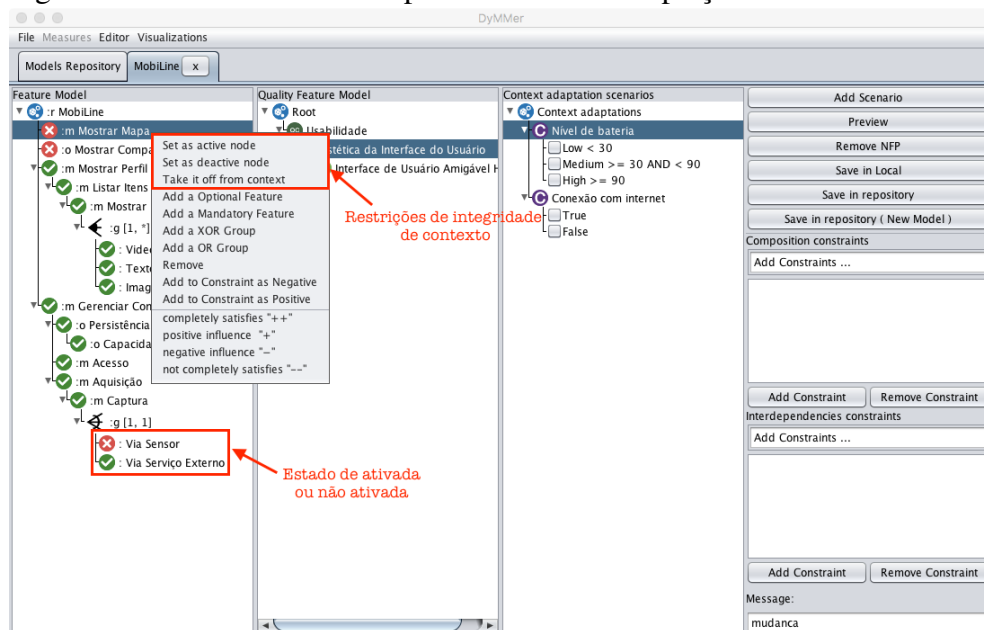


Fonte – Elaborada pelo Autor.

7.3.4 Adição de múltiplos cenários de adaptações de contexto

Um cenário de adaptação de contexto é composto opcionalmente por um modelo de *features* de qualidade, as definições de adaptações de contexto e as restrições de interdependências entre *features* e PNFs. Com a extensão realizada é possível adicionar um ou mais cenários de adaptações de contexto em um mesmo modelo de *features*, quando isso ocorre é possível representar múltiplos cenários de adaptações de contexto em um modelo de *features*. Para adicionar um ou mais cenário de adaptação de contexto é necessário ter realizado previamente as definições exemplificadas anteriormente, em seguida selecionar a opção “Add Scenario” (ver Figura 38). A cada definição de cenário de adaptação de contexto diferente, deve ser selecionado a opção “Add Scenario” desse modo é possível representar múltiplos cenários de adaptações de contexto.

Figura 38 – Adicionando múltiplos cenários de adaptações de contexto



Fonte – Elaborada pelo Autor.

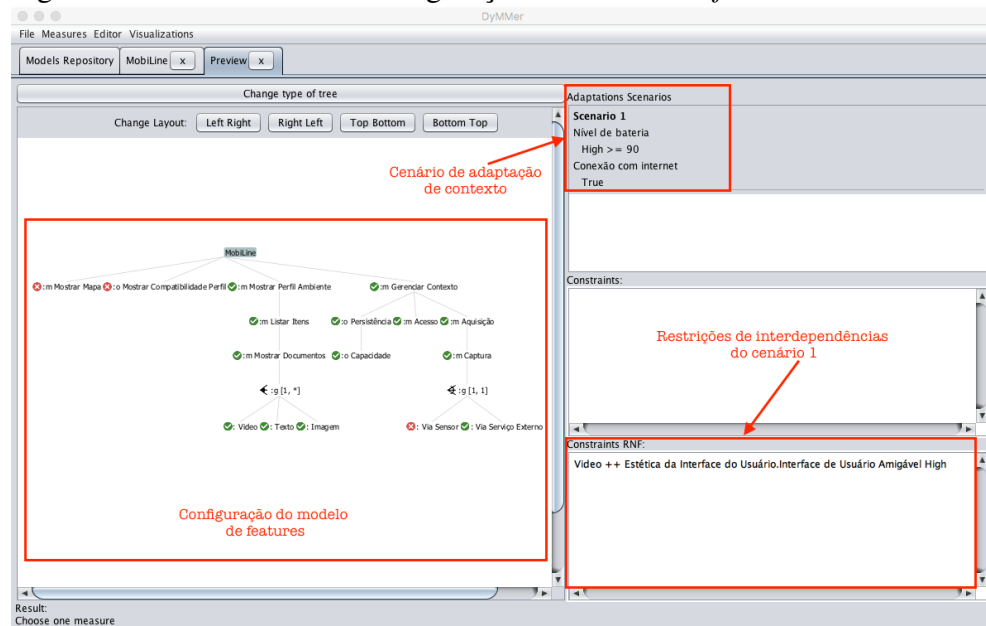
7.3.5 Visualização da configuração do modelo de features

A visualização da configuração do modelo de *features* pode ser feita de duas formas: i) após a adição de um cenário de contexto em edição; e ii) por meio do módulo de visualização do modelo de *features*.

A visualização de um cenário de adaptação de contexto após sua edição e por meio do módulo de visualização do modelo de *features* é apresentado na Figura 39), desse modo a

cada adição de um novo cenário de contexto o engenheiro de domínio pode visualizar uma ou mais configurações adicionadas. A visualização por meio do módulo de visualização pode ser realizada sem a necessidade de edição do modelo. A partir da visualização da configuração do modelo de *features* pode-se observar facilmente as configurações de um modelo de acordo com os múltiplos cenários de adaptações de contexto.

Figura 39 – Visualizando a configuração do modelo de *features*



Fonte – Elaborada pelo Autor.

7.4 Teste de usabilidade

Para avaliar a ferramenta, foi realizado um teste de usabilidade. Esse teste teve como objetivo avaliar a usabilidade das extensões adicionadas a DyMMer, assim como a sua qualidade visual.

Para avaliar a usabilidade foi utilizado o método *The Post-Study System Usability Questionnaire (PSSUQ)* proposto por Lewis (1995). Esse método permite calcular quatro diferentes fatores a partir das respostas obtidas por meio da aplicação de um questionário de avaliação. Os quatro fatores são: i) satisfação geral; ii) utilidade da ferramenta; iii) qualidade da informação; e iv) qualidade da interface. Este método foi escolhido, pois é possível avaliar de forma simples e eficaz a satisfação geral do usuário e a qualidade da interface em relação a ferramenta. Assim como é possível avaliar a utilidade da ferramenta por meio da execução da abordagem.

Segundo PEREIRA (2014) este método é ideal para testes de usabilidade para

experimentos e possui quatro fatores que podem ser avaliados, permitindo uma ampla avaliação da usabilidade da ferramenta.

Para a aplicação deste método foi definido um conjunto de tarefas, em que o usuário após a execução das tarefas respondem a um questionário do método PSSUQ (Apêndice). Este questionário é composto por 19 afirmações, em que cada afirmação possuem sete itens de respostas em um escala de sete pontes, variando entre “Discordo totalmente” com sete pontos e “Concordo totalmente” com 1 ponto.

PEREIRA (2014) afirma que quando maior a escala, mais difícil dos participantes interpretarem o questionário, desse modo foi utilizado a escala utilizada por PEREIRA (2014), esta escala de possui cinco pontos que variam entre concordo totalmente, concordo, neutro, discordo e discordo totalmente.

7.4.1 Usuários

Os usuários selecionados para o teste de usabilidade foram alunos da área de tecnologia da informação e comunicação. Ao todo participaram dois usuários, um do curso de Engenharia de Software e outro do curso de Sistemas de Informação. Todos os participantes possuem conhecimentos na modelagem de software, de as dinâmicos, e de PNFs. Sendo que dois dos participantes trabalham com desenvolvimento de sistemas.

7.4.2 Tarefas

Para realizar o teste, os usuários tiveram que realizar uma sequência de tarefas que cobriram todas os passos da abordagem ReMINDER. O objetivo foi à aplicação das etapas e passos da abordagem ReMINDER fazendo uso dos artefatos criados no estudo de observação. Sendo possível a avaliação da operacionalização da abordagem.

As tarefas atribuídas ao usuário envolvem todos os passos das fases da abordagem ReMINDER. As tarefas envolvem a modelagem das PNFs no modelo de *features* de qualidade, as definições de cenários de adaptações de contexto, especificação das restrições de ativação e de desativação de *features* de acordo com os cenários de adaptações, especificações de restrições de interdependências entre *features* de contexto e PNFs e visualização dos cenários de adaptações de contexto.

7.4.3 Execução do teste

Para a execução do teste de usabilidade, foi reservado um período de 1 hora. Esse período foi dividido em dois momentos de 20 minutos e 40 minutos. Primeiramente foi recapitulado brevemente as fases, etapas e artefatos gerados pelo estudo de observação. Esses artefatos são utilizados no teste de usabilidade como dados de entrada para utilização da ferramenta, afim de facilitar a execução das tarefas.

No segundo momento foi realizado o teste de usabilidade. Inicialmente cada participante foi alocado em uma máquina com a ferramenta DyMMer já instalada e aberta. Para a realização do teste, foi entregue uma lista de tarefas (Apêndice K) a serem realizadas. Por fim, após o término das execução do teste, cada participante deveria responder um questionário para avaliação da usabilidade da ferramenta.

7.4.4 Resultados

A análise dos dados dos questionários foi realizada com base na método de avaliação proposta por Lewis (1995). Esse método propõe uma forma de analisar os resultados obtidos em questionários que utilizam a escala de Likert, ou seja, níveis de satisfação.

De acordo esse método deve ser calculada a média para representar o número neutro, utilizado para representar se um usuário é neutro sobre determinado aspecto da ferramenta. Para realização deste teste de usabilidade foi utilizado o número neutro calculado por PEREIRA (2014), cujo valor é 57. Como descrito na Seção 7.4, neste teste foi adotada a escala proposta por PEREIRA (2014) para facilitar a interpretação do questionário, desse modo o número neutro será 57.

Na Tabela 1 são apresentadas as respostas dos usuários em relação ao questionário de avaliação da usabilidade da ferramenta. Nesta Tabela os usuários são apresentados pelas linhas A e B e as perguntas do questionário pelas colunas de 1 a 19. A coluna Total corresponde a somas das respostas de cada participante.

Tabela 1 – Respostas do questionário.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	Total
A	5	5	5	5	5	5	5	5	3	3	3	5	5	5	5	4	5	5	5	88
B	5	4	5	3	5	5	5	5	1	4	2	4	5	5	5	5	5	5	5	83
Total	10	9	10	7	10	10	10	10	4	7	5	9	10	10	10	9	10	10	10	171

Fonte – Elaborada pelo Autor.

7.4.4.1 Satisfação Geral

A análise da satisfação geral é realizada com a comparação do total do somatório das respostas de cada usuário com o valor da média geral. Para realizar a comparação, primeiramente deve-se calcular a média geral, esse valor é o resultado da soma dos dados da coluna Total dividido pelo número de usuários. Dessa forma, a média é $(171/2)$, cujo resultado é 85.5. Portanto, pode-se observar que a média de todos os usuários foi superior a 57, número neutro, desse forma a ferramenta foi bem aceita pelos usuários.

Com base no número neutro foi possível verificar que nenhum dos participantes demonstrou neutralidade em suas respostas, pois a soma das suas respostas são superior a 57.

Além de avaliar a satisfação geral de cada usuário, também foi avaliado a satisfação em relação a cada pergunta do questionário, com objetivo de identificar as perguntas cujas respostas foram de maneira geral positivas, negativas ou neutras. Para realizar essa avaliação, foi calculado a média do valor neutro em relação ao número de usuários, cujo valor é $3 \times 2 = 6$.

Para cada pergunta foi comparado o seu somatório com a média descoberta. Uma vez realizada essa comparação, verificou-se que a média das perguntas 4 (média 7), 10 (média 7) e 11 (média 5) se aproximam do valor da média 6, assim é possível concluir que os usuários possuem opiniões neutras sobre as perguntas 4 (Conseguir completar rapidamente as tarefas e cenários utilizando esta ferramenta), 10 (Sempre que cometi um erro durante a utilização da ferramenta, consegui recuperar de forma fácil e rápida) e 11 (A informação fornecida pela ferramenta (como ajuda online, mensagens ou outra documentação) foi clara). A neutralidade nesses perguntas ocorreu porque no desenvolvimento da extensão da ferramenta, não houve a preocupação na implementação de uma *Help*.

Em relação a pergunta 9, notou-se que a média desta pergunta (média 4) é inferior a média 6. Portanto, os usuários ficaram insatisfeitos com a pergunta 9 (A ferramenta deu mensagens de erros que me indicaram claramente como resolver os problemas). Essa insatisfação ocorre, pois no desenvolvimento da extensão da ferramenta, não houve a preocupação na implementação de muitos validadores para as possíveis atividades de edições permitidas na ferramenta.

7.4.4.2 Utilidade da Ferramenta

Essa análise tem como objetivo verificar a utilização da ferramenta para operacionalização da abordagem. Para verificar a utilidade da ferramenta foram analisadas as perguntas 1 até a 8 do questionário. As respostas destas perguntas são apresentadas na Tabela 2.

Tabela 2 – Dados referente a utilidade da ferramenta.

	1	2	3	4	5	6	7	8	Total
A	5	5	5	5	5	5	5	5	40
B	5	4	5	3	5	5	5	5	37
Total	10	9	5	7	10	10	10	10	77

Fonte – Elaborada pelo autor

Nesta análise a média é calculada pelo número 3, que corresponde ao valor neutro, multiplicado por 8, que representa o total de perguntas analisadas, logo temos que $3 \times 8 = 24$.

A média das respostas dos usuários é igual ao total do somatório das respostas de cada usuário pelo número de total de usuários, logo temos que a média é $77/2 = 38,5$. Assim é possível perceber que a média das respostas dos usuários é maior que a média das perguntas. Portanto, pode-se concluir que a ferramenta foi considerada útil para a operacionalização da abordagem ReMINDER.

7.4.4.3 Qualidade da Informação

Essa análise tem como objetivo saber qual a qualidade dos artefatos gerados com a utilização da ferramenta. Para verificar a qualidade da informação foram analisadas as perguntas 13, 14 e 15. As respostas destas perguntas são apresentadas na Tabela 3.

Tabela 3 – Dados referente a qualidade da informação.

	9	10	11	12	13	14	15	Total
A	3	3	3	5	5	5	5	29
B	1	4	2	4	5	5	5	26
Total	4	7	5	9	10	10	10	55

Fonte – Elaborada pelo autor

Da mesma forma realizada na análise da Utilidade da Ferramenta, primeiramente foi calculada a média. Nesta análise a média é calculada pelo número 3, que corresponde ao valor neutro, multiplicado por 3, que representa o total de perguntas analisadas, logo temos que $3 \times 3 =$

9.

A média das respostas dos usuários é igual ao total do somatório das respostas de cada usuário pelo número de total de usuários, logo temos que a média é $55/2 = 27,5$. Assim é possível perceber que a média das respostas dos usuários é maior que a média das perguntas. Portanto, pode-se concluir que os participantes consideraram que as informações obtidas são de boa qualidade.

7.4.4.4 Qualidade da Interface

Essa análise tem como objetivo saber qual a qualidade da interface da ferramenta. Para verificar a qualidade da interface foram analisadas as perguntas 16, 17 e 18. As respostas destas perguntas são apresentadas na Tabela 4.

Tabela 4 – Dados referente a qualidade da interface.

	16	17	18	Total
A	4	5	5	14
B	5	5	5	15
Total	9	10	10	29

Fonte – Elaborada pelo autor

Nesta análise a média é calculada pelo número 3, que corresponde ao valor neutro, multiplicado por 3, que representa o total de perguntas analisadas, logo temos que $3*3 = 9$.

A média das respostas dos usuários é igual ao total do somatório das respostas de cada usuário pelo número de total de usuários, logo temos que a média é $29/2 = 14,5$. Assim é possível perceber que a média das respostas dos usuários é maior que a média das perguntas. Portanto, pode-se concluir que os participantes consideraram que a interface da ferramenta possui uma boa qualidade.

7.5 Considerações Finais

Nesta Seção foram apresentadas as extensões adicionadas a ferramenta DyMMer para operacionalizar da aplicação da abordagem ReMINDER. As extensões adicionadas foram a criação do modelo de *features* de qualidade, para representar as PNFs; adição de restrições de interdependências entre as *features* e PNFs em forma de links de contribuição; adição de múltiplos cenários de adaptações de contexto em um modelo de *features* de LPSD; adição

de restrições de integridade de contexto em um dado cenário de adaptação de contexto; e a visualização do modelo de *features* conforme um ou mais cenários de adaptações de contexto, também exibindo a visualização das restrições.

Por fim, foi apresentado o teste de usabilidade das extensões adicionadas a ferramenta DyMMer. Para realizar esse teste, cada participante realizou uma série de atividades e cada fase da abordagem ReMINDER com o auxílio dos templates preenchidos no estudo de observação. Após o teste, cada participante respondeu um questionário, a fim de obter uma avaliação de cada usuário em relação à usabilidade da ferramenta. O questionário utilizado foi proposto no método *The Post-Study System Usability Questionnaire* (PSSUQ). Esse método permitiu que seja avaliado quatro fatores: i) a satisfação geral da ferramenta; ii) a utilidade da informação; iii) a qualidade da informação; e iv) a qualidade da interface.

Em relação a satisfação geral da ferramenta, podemos concluir que no geral a ferramenta foi bem aceita pelos usuários. No entanto, foi identificado que a ferramenta deixou a desejar quanto às mensagens de erro e a disponibilização de um *Help*, logo a ferramenta não auxiliou em todos os casos o usuário a corrigir os erros que venham a ocorrer, e devido a falta de um *Help* esse problema pode ter sido potencializado. Uma solução para este problema seria a implementação de mais validadores e de um sistema de *Help*.

Em relação a utilidade da ferramenta foi verificado que a ferramenta permite operacionalizar todas as fases e passos da abordagem ReMINDER. A partir da avaliação da qualidade, pode-se concluir que os artefatos gerados pela ferramenta são de qualidade. Por fim, na avaliação da qualidade da interface, pode-se concluir que a ferramenta possui uma interface de qualidade.

De modo geral, após a análise e interpretação de todos os resultados obtidos por meio da aplicação do teste de usabilidade, conclui-se que a ferramenta é útil para operacionalização da abordagem.

8 CONSIDERAÇÕES FINAIS

Esta Seção é descreve as considerações do finais do trabalho. Primeiramente serão apresentadas as possíveis ameaças à validade dos resultados obtidos deste trabalho. Em seguida, é apresentado um levantamento sobre os possíveis trabalhos futuros. Logo após são apresentadas as contribuições deste trabalho. Por fim, são descritas as conclusões sobre a execução deste trabalho.

8.1 Ameaças à Validade

Esta seção discute as ameaças que podem ter afetado os resultados deste trabalho. Foi verificado que as ameaças identificadas estão relacionadas à validade de construção. Validade de construção preocupa-se com a generalização dos resultados que diz respeito a aplicabilidade do trabalho em diferentes contextos de uma mesma área (WOHLIN et al., 2012).

A primeira possível ameaça é sobre a efetividade do estudo de observação e do teste de usabilidade, uma vez que a quantidade de participantes foi reduzida. E os participantes selecionados faziam parte do mesmo grupo de estudos do experimentador, devido às necessidade de pessoas com conhecimento prévio nos tópicos envolvidos no estudo. Dessa forma, os resultados podem ter sido influenciados pela relação entre os participantes e o experimentador. Para resolver esta ameaça o estudo de observação tem que ser aplicado com outros especialistas.

A segunda possível ameaça foi com relação ao modelo de *features* utilizado ser pequeno. Isso foi necessário devido a limitação de tempo para aplicação do estudo de observação. Desse modo, é possível que a aplicação da abordagem em modelo de *features* maiores aumente a sua complexidade.

8.2 Trabalhos Futuros

A partir dos resultados deste trabalho foi possível identificar melhorias e possíveis trabalhos futuros.

Uma melhoria sobre as extensões adicionadas a ferramenta DyMMer é a adição de um sistema de *help*. Além disso a implementação de mais validadores para as áreas de edição da ferramenta. Com isso seria possível evitar erros cometidos pelos usuários da ferramenta e um aumento da sua qualidade de informação e usabilidade.

Em relação a abordagem uma possível melhoria observada durante o estudo de

observação seria a utilização de siglas em alguns *templates* afim de reduzir o esforço do seu preenchimento.

Com relação aos trabalhos futuros foram identificados os seguintes tópicos:

- Definição de uma abordagem para garantir o atendimento das PNFs em modelos de *features* de LPSD após o processo de reconfiguração em tempo de execução, garantindo um aumento na qualidade do produto final; e
- Adição de novas PNFs ao catálogo de PNFs, que emergem em tempo de execução associadas as características e subcaracterísticas de qualidade segundo a norma SQUaRE.

8.3 Contribuições

As principais contribuições identificadas com a realização deste trabalho foram:

- Extensão do modelo de *features* de LPSD para a representação de PNFs e de múltiplos cenários de adaptações de contexto;
- Proposição de um modelo de *features* de qualidade para representação das PNFs;
- Criação de um catálogo de PNFs que emergem em tempo de execução;
- Definição de restrições de interdependências entre *features* e PNFs por meio de *links* de contribuição; e
- Extensão da ferramenta DyMMer para especificação de múltiplos cenários de adaptações de contexto e representação de PNFs.

8.4 Conclusões

Este trabalho realizou a criação de uma abordagem sistemática para modelagem de PNFs em LPSD. A abordagem foi dividida em duas fases. A primeira fase tem como objetivo a identificação e representação de PNFs no modelo de *features* sendo composta por quatro passos. A segunda fase tem como objetivo a identificação de restrições e cenários de adaptações de contexto, as restrições identificadas nesta fase incluíram regras de contexto, restrições de ativação e de desativação de *features*, restrições de interdependência entre *features* e PNFs sendo composta por cinco fases. A abordagem foi denominada ReMINDER.

Foram apresentados um conjunto de conceitos que fundamentaram a execução deste trabalho. Também foram apresentados e comparados alguns trabalhos relacionados.

Primeiramente, foi apresentada a abordagem ReMINDER, com as fases e passos necessários para a sua aplicação. Após a apresentação da abordagem foi apresentado um estudo de observação realizado para validar a abordagem. Com base nesse estudo foram coletados dados que apresentaram que foi possível modelar as PNFs relevantes para um modelo de *features* de LPSD, assim como a especificação das restrições de interdependências entre as *features* e PNFs em determinado cenário de adaptação de contexto por meio da utilização da abordagem.

Um dos pontos positivos da abordagem criada foi a possibilidade de especificar um conjunto de PNFs que afetam a qualidade de um produto derivado de uma LPS e de uma LPSD em tempo de projeto. Além disso, a criação de um catálogo de PNFs que emergem em tempo de execução de modo que essas PNFs são classificadas de acordo com as características e subcaracterísticas da norma SQUaRE. Outro ponto relevante foi a adição de restrições de interdependências entre *features* e as PNFs na forma de *links* de contribuição, possibilitando a especificação de quando uma determinada *feature* em seu estado atual afeta uma determinada PNF.

Após a apresentação do estudo de observação, foram apresentadas as extensões adicionadas a ferramenta DyMMer como suporte a operacionalização da abordagem ReMINDER. Essa ferramenta auxilia na execução das fases e passos da abordagem. Uma vez apresentada as extensões realizadas na ferramenta foi apresentado um teste de usabilidade a fim de verificar a usabilidade das extensões adicionadas. Por meio desse teste foi possível verificar que a ferramenta foi bem aceita pelos participantes da avaliação. Porém, também foram identificados algumas limitações da ferramenta.

Com a realização deste trabalho, espera-se que a abordagem ReMINDER auxilie engenheiros de domínio e pesquisadores na redução do tempo e esforço para especificação e modelagem de PNFs em LPSD, garantindo uma maior qualidade dos produtos derivados de uma LPSD.

REFERÊNCIAS

- ALVES, V.; NIU, N.; ALVES, C.; VALENÇA, G. Requirements engineering for software product lines: A systematic literature review. **Information and Software Technology**, Elsevier, v. 52, n. 8, p. 806–820, 2010.
- BENAVIDES, D.; SEGURA, S.; RUIZ-CORTÉS, A. Automated analysis of feature models 20 years later: A literature review. **Information Systems**, Elsevier, v. 35, n. 6, p. 615–636, 2010.
- BENAVIDES, D.; TRINIDAD, P.; RUIZ-CORTÉS, A. Automated reasoning on feature models. In: SPRINGER. **Advanced Information Systems Engineering**. Porto, Portugal, 2005. p. 491–503.
- BENCOMO, N.; BLAIR, G. S.; FLORES-CORTÉS, C. A.; SAWYER, P. Reflective component-based technologies to support dynamic variability. In: CITESEER. **VaMoS**. Universität Duisburg-Essen, Germany, 2008. p. 141–150.
- BENCOMO, N.; HALLSTEINSEN, S.; ALMEIDA, E. S. D. A view of the dynamic software product line landscape. **Computer**, v. 45, n. 10, p. 36–41, 2012.
- BEUCHE, D.; DALGARNO, M. Software product line engineering with feature models. **Overload Journal**, v. 78, p. 5–8, 2007.
- BEZERRA, C. I. M.; BARBOSA, J.; FREIRES, J. H.; ANDRADE, R. M. C.; MONTEIRO, J. M. S. Dymmer: A measurement-based tool to support quality evaluation of dspl feature models. In: SPRINGER. **International Conference on Software Product Lines**. [S.l.], 2016.
- BOEHM, B. W.; BROWN, J. R.; KASPAR, H. Characteristics of software quality. North-Holland, 1978.
- BOSCH, J. Software variability management. In: IEEE COMPUTER SOCIETY. **Proceedings of the 26th international Conference on Software Engineering**. Edinburgh, United Kingdom, 2004. p. 720–721.
- CAPILLA, R.; BOSCH, J. The promise and challenge of runtime variability. **Computer**, IEEE, v. 44, n. 12, p. 93–95, 2011.
- CAPILLA, R.; BOSCH, J.; TRINIDAD, P.; RUIZ-CORTÉS, A.; HINCHEY, M. An overview of dynamic software product line architectures and techniques: Observations from research and industry. **Journal of Systems and Software**, Elsevier, v. 91, p. 3–23, 2014.
- CHUNG, L.; LEITE, J. C. S. do P. On non-functional requirements in software engineering. In: **Conceptual modeling: Foundations and applications**. Innsbruck, Austria: Springer, 2009. p. 363–379.
- CLEMENTS, P.; NORTHROP, L. Software product lines: practices and patterns. Addison-Wesley, 2002.
- CLEMENTS, P. C. **Software architecture in practice**. Tese (Doutorado) — Software Engineering Institute, 2002.
- CZARNECKI, K.; HELSEN, S.; EISENECKER, U. Formalizing cardinality-based feature models and their specialization. **Software process: Improvement and practice**, Wiley Online Library, v. 10, n. 1, p. 7–29, 2005.

CZARNECKI, K.; HELSEN, S.; EISENECKER, U. Staged configuration through specialization and multilevel configuration of feature models. **Software Process: Improvement and Practice**, Citeseer, v. 10, n. 2, p. 143–169, 2005.

ETXEBERRIA, L.; SAGARDUI, G. Product-line architecture: New issues for evaluation. In: **Software Product Lines**. Rennes, France: Springer, 2005. p. 174–185.

ETXEBERRIA, L.; SAGARDUI, G.; BELATEGI, L. Quality aware software product line engineering. **Journal of the Brazilian Computer Society**, SciELO Brasil, v. 14, n. 1, p. 57–69, 2008.

FERNANDES, P.; WERNER, C.; MURTA, L. G. P. Feature modeling for context-aware software product lines. In: **SEKE**. San Francisco, CA, USA: [s.n.], 2008. p. 758–763.

FERNANDES, P. C. C. **COPPE/UFRJ**. Tese (Doutorado) — Universidade Federal do Rio de Janeiro, 2009.

FREIRES JUNIOR, J. H. **DyMMER: Uma ferramenta para avaliação de qualidade do modelo de features baseado em medidas de linhas de produtos de software dinâmicas**. 66 f. Monografia (Graduação) — Universidade Federal do Ceará, Quixadá, 2014.

HALLSTEINSEN, S.; HINCHEY, M.; PARK, S.; SCHMID, K. Dynamic software product lines. **Computer**, IEEE, v. 41, n. 4, p. 93–95, 2008.

HAMMANI, F. Z. Survey of non-functional requirements modeling and verification of software product lines. In: IEEE. **Research Challenges in Information Science (RCIS), 2014 IEEE Eighth International Conference on**. Marrakech, Morocco, 2014. p. 1–6.

HINCHEY, M.; PARK, S.; SCHMID, K. Building dynamic software product lines. **Computer**, IEEE, n. 10, p. 22–26, 2012.

HONG, D.; CHIU, D. K.; SHEN, V. Y. Requirements elicitation for the design of context-aware applications in a ubiquitous environment. In: ACM. **Proceedings of the 7th international conference on Electronic commerce**. Xi'an, China, 2005. p. 590–596.

ISO/IEC. **ISO/IEC FDIS 9126-1 - Software Engineering – Product Quality", Part 1: Quality Model, first ed.** [S.l.], 2001.

ISO/IEC. **ISO/IEC 25010 - Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuARE) - System and software quality models**. [S.l.], 2011.

KANG, K. C.; COHEN, S. G.; HESS, J. A.; NOVAK, W. E.; PETERSON, A. S. **Feature-oriented domain analysis (FODA) feasibility study**. [S.l.], 1990.

LAMSWEERDE, A. V. Goal-oriented requirements engineering: A guided tour. In: IEEE. **Requirements Engineering, 2001. Proceedings. Fifth IEEE International Symposium on**. [S.l.], 2001. p. 249–262.

LEE, J.; MUTHIG, D. Feature-oriented variability management in product line engineering. **Communications of the ACM**, ACM, v. 49, n. 12, p. 55–59, 2006.

LEWIS, J. R. Ibm computer usability satisfaction questionnaires: psychometric evaluation and instructions for use. **International Journal of Human-Computer Interaction**, Taylor & Francis, v. 7, n. 1, p. 57–78, 1995.

MARI, M.; EILA, N. The impact of maintainability on component-based software systems. In: IEEE. **Euromicro Conference, 2003. Proceedings. 29th**. Belek-Antalya, Turkey, 2003. p. 25–32.

MARINHO, F. G.; ANDRADE, R. M.; WERNER, C.; VIANA, W.; MAIA, M. E.; ROCHA, L. S.; TEIXEIRA, E.; FILHO, J. B. F.; DANTAS, V. L.; LIMA, F. et al. Mobiline: A nested software product line for the domain of mobile and context-aware applications. **Science of Computer Programming**, Elsevier, v. 78, n. 12, p. 2381–2398, 2013.

MASSEN, T. von der; LICHTER, H. Modeling variability by uml use case diagrams. In: CITESEER. **Proceedings of the International Workshop on Requirements Engineering for product lines**. [S.l.], 2002. p. 19–25.

MENDONCA, M.; WASOWSKI, A.; CZARNECKI, K.; COWAN, D. Efficient compilation techniques for large scale feature models. In: ACM. **Proceedings of the 7th international conference on Generative programming and component engineering**. [S.l.], 2008. p. 13–22.

NOORIAN, M.; BAGHERI, E.; DU, W. Non-functional properties in software product lines: A taxonomy for classification. In: **SEKE**. Hotel Sofitel, Redwood City, San Francisco Bay, USA: [s.n.], 2012. v. 12, p. 663–667.

NORTHROP, L. M. Sei's software product line tenets. **IEEE software**, IEEE Computer Society, v. 19, n. 4, p. 32, 2002.

PACHECO, C.; GARCIA, I.; CALVO-MANZANO, J. A.; ARCILLA, M. A proposed model for reuse of software requirements in requirements catalog. **Journal of Software: Evolution and Process**, Wiley Online Library, v. 27, n. 1, p. 1–21, 2015.

PEREIRA, T. C. Bvcon-tool: uma ferramenta para apoiar uma abordagem de configuração de processos de negócio dinâmicos. Universidade Federal de Pernambuco, 2014.

POHL, K.; BÖCKLE, G.; LINDEN, F. J. van der. **Software product line engineering: foundations, principles and techniques**. [S.l.]: Springer Science & Business Media, 2005.

SANCHEZ, L. E.; DIAZ-PACE, J. A.; ZUNINO, A.; MOISAN, S.; RIGAULT, J.-P. An approach based on feature models and quality criteria for adapting component-based systems. **Journal of Software Engineering Research and Development**, Springer, v. 3, n. 1, p. 1–30, 2015.

SIEGMUND, N.; KUHLEMANN, M.; ROSENMÜLLER, M.; KAESTNER, C.; SAAKE, G. Integrated product line model for semi-automated product derivation using non-functional properties. In: **VaMoS**. Universität Duisburg-Essen, Germany: [s.n.], 2008. p. 25–32.

SOARES, L. R.; POTENA, P.; MACHADO, I. do C.; CRNKOVIC, I.; ALMEIDA, E. S. de. Analysis of non-functional properties in software product lines: a systematic review. In: IEEE. **2014 40th EUROMICRO Conference on Software Engineering and Advanced Applications**. [S.l.], 2014. p. 328–335.

SOMMERVILLE, I.; KOTONYA, G. **Requirements engineering: processes and techniques**. [S.l.]: John Wiley & Sons, Inc., 1998.

UCHÔA, A. G.; BEZERRA, C. I.; MACHADO, I. C.; MONTEIRO, J. M.; ANDRADE, R. M. Reminder: an approach to modeling non-functional properties in dynamic software product lines. In: SPRINGER. **International Conference on Software Reuse**. [S.l.], 2017. p. 65–73.

UCHÔA, A. G.; BEZERRA, C. I. M. Modelagem de requisitos não-funcionais em modelos de features de linha de produto de software dinâmicas. In: SBC. **Proceedings of the 3rd Latin-American School on Software Engineering**. Natal, 2016.

UCHÔA, A. G.; LIMA, L. P.; BEZERRA, C. I.; MONTEIRO, J. M.; ANDRADE, R. M. Dymmer-nfp: Modeling non-functional properties and multiple context adaptation scenarios in software product lines. In: SPRINGER. **International Conference on Software Reuse**. [S.l.], 2017. p. 175–183.

WOHLIN, C.; RUNESON, P.; HÖST, M.; OHLSSON, M. C.; REGNELL, B.; WESSLÉN, A. **Experimentation in software engineering**. [S.l.]: Springer Science & Business Media, 2012.

ZHANG, G.; YE, H.; LIN, Y. Quality attribute modeling and quality aware product configuration in software product lines. **Software Quality Journal**, Springer, v. 22, n. 3, p. 365–401, 2014.

APÊNDICE A – FORMULÁRIO DE CONSENTIMENTO

1. OBJETIVO DO ESTUDO

Este estudo visa caracterizar a aplicação da abordagem ReMINDER no apoio a modelagem de Propriedades Não-Funcionais no desenvolvimento de Linha de Produtos de Software Dinâmicas (LPSD).

2. IDADE

Eu declaro ter mais de 18 anos de idade e concordar em participar de um estudo conduzido por Anderson Gonçalves Uchôa na Universidade Federal do Ceará - Campus Quixadá.

3. PROCEDIMENTO

Este estudo acontecerá em três etapas. Na primeira etapa, é aplicado um questionário para avaliação do *background* dos participantes, em seguida é apresentada uma visão geral da abordagem e o conjunto de artefatos que serão utilizados neste estudo, parte de um modelo de *features* de uma Linha de Produto de Software (LPS) e a especificação de diferentes cenários de adaptações de contexto e regras de adaptações de contexto que devem ser modeladas. Na segunda etapa, os participantes deverão realizar a identificação das propriedades não-funcionais e suas interdependências com as *features* ativadas e desativadas de acordo os cenários de adaptações de contexto, de forma manual, com o auxílio do preenchimento dos artefatos. Na terceira etapa, é apresentado o processo de utilização da ferramenta que suporta a abordagem ReMINDER. Em seguida, os participantes deverão modelar o modelo de *features* de LPSD conforme as tarefas realizadas na primeira e segunda etapa por meio da utilização da ferramenta. Por último, um formulário de avaliação sobre o estudo realizado deve ser preenchido pelo participante.

4. CONFIDENCIALIDADE

Toda informação coletada neste estudo é confidencial, e meu nome não será divulgado. Da mesma forma, me comprometo a não comunicar os meus resultados enquanto não terminar o estudo, bem como manter sigilo das técnicas e documentos apresentados e que fazem parte do experimento.

5. BENEFÍCIOS E LIBERDADE DE DESISTÊNCIA

Eu entendo que os benefícios que receberei deste estudo são limitados ao aprendizado do material que é distribuído e apresentado. Também entendo que sou livre para realizar perguntas a qualquer momento, solicitar que qualquer informação relacionada a minha pessoa não seja incluída no estudo ou comunicar minha desistência de participação. Por fim, declaro que participo de livre e espontânea vontade com o único intuito de contribuir para o avanço e desenvolvimento de técnicas e processos para a Engenharia de Software.

6. PESQUISADOR RESPONSÁVEL

Anderson Gonçalves Uchôa

Universidade Federal do Ceará - UFC Campus Quixadá

7. PROFESSORA RESPONSÁVEL

Profa. Dra.Carla Ilane Moreira Bezerra

Universidade Federal do Ceará - UFC Campus Quixadá

_____, ____ de _____ de 201____

Nome (em letra de forma)

Assinatura

APÊNDICE B – MODELAGEM DE LINHAS DE PRODUTOS DE SOFTWARE DINÂMICAS - FORMULÁRIO DE *BACKGROUND*

Este formulário de *background* do experimento controlado Modelagem de DSPL para caracterização do participante. Este formulário contém algumas perguntas sobre sua experiência acadêmica e profissional.

B.1 PARTE 1 - PERFIL DO ESPECIALISTA

1. Nome Completo:

2. E-mail:

3. Instituição:

4. Qual o seu nível de formação acadêmica ?

- (a) Pós-Doutorado
- (b) Doutorado
- (c) Doutorando
- (d) Mestrado
- (e) Mestrando
- (f) Graduado
- (g) Graduando

5. Quantos anos desde a sua formação acadêmica ?

- (a) Menos de 1 ano
- (b) Mais de 1 ano e até 5 anos
- (c) Mais de 5 anos e menos de 10 anos
- (d) Mais de 10 anos

B.2 PARTE 2 - CONHECIMENTO TÉCNICO

Esta Seção será utilizada para compreender quão familiar você é com o domínio que será utilizado para as atividades durante o experimento. Por favor, indique o grau de experiência nesta seção, selecionando a opção que melhor se encaixa em seu perfil.

1. Uma Linha de Produto de Software é ?

- (a) Um conjunto de sistemas de software intensivos que compartilham um conjunto comum e gerenciado de *features*, desenvolvido a partir de um conjunto de core *assets* de uma maneira prescrita.
- (b) Um conjunto de produtos construídos a partir de uma plataforma de componentes.
- (c) Não estou certo sobre isso

2. Qual o seu tempo de experiência em Linhas de Produtos de Software (LPS) ?

- (a) Menos de 1 ano
- (b) Mais de 1 ano e até 5 anos
- (c) Mais de 5 anos e menos de 10 anos
- (d) Mais de 10 anos

3. Você já aplicou a abordagem de LPS para desenvolver software ?

- (a) Sim, mas somente no domínio de pesquisa
- (b) Sim, mas somente no domínio da indústria
- (c) Sim, em ambos, pesquisa e indústria
- (d) Não

4. Qual o seu tempo de experiência em Linhas de Produtos de Software Dinâmicas (LPSD) ?

- (a) Menos de 1 ano
- (b) Mais de 1 ano e até 5 anos
- (c) Mais de 5 anos e menos de 10 anos
- (d) Mais de 10 anos

5. Você já usou uma abordagem de LPSD para desenvolver software ?

- (a) Sim, mas somente no domínio de pesquisa
- (b) Sim, mas somente no domínio da indústria
- (c) Mais de 5 anos e menos de 10 anos
- (d) Sim, em ambos domínios pesquisa e indústria
- (e) Não

6. Qual o seu tempo de experiência em modelagem de software ?

- (a) Menos de 1 ano
- (b) Mais de 1 ano e até 5 anos
- (c) Mais de 5 anos e menos de 10 anos
- (d) Mais de 10 anos

7. Você já usou ou analisou uma técnica de modelagem para desenvolver software ?

- (a) Sim, mas somente no domínio de pesquisa
- (b) Sim, mas somente no domínio da indústria
- (c) Sim, em ambos domínios pesquisa e indústria
- (d) Não

8. Qual o seu tempo de experiência em modelagem de aspectos dinâmicos ?

- (a) Menos de 1 ano
- (b) Mais de 1 ano e até 5 anos
- (c) Mais de 5 anos e menos de 10 anos
- (d) Mais de 10 anos

9. Você já usou ou analisou uma técnica de modelagem de aspectos dinâmicos para desenvolver software?

- (a) Sim, mas somente no domínio de pesquisa
- (b) Sim, mas somente no domínio da indústria
- (c) Sim, em ambos domínios pesquisa e indústria
- (d) Não

10. **Qual o seu tempo de experiência em modelagem de Propriedades Não-Funcionais (PNFs) ?**

- (a) Menos de 1 ano
- (b) Mais de 1 ano e até 5 anos
- (c) Mais de 5 anos e menos de 10 anos
- (d) Mais de 10 anos

11. **Você já usou ou analisou uma técnica de modelagem de PNFs para desenvolver software ?**

- (a) Sim, mas somente no domínio de pesquisa
- (b) Sim, mas somente no domínio da indústria
- (c) Sim, em ambos domínios pesquisa e indústria
- (d) Não

12. **Por favor, indique o grau de sua experiência nos tópicos abaixo, seguindo a escala de 5 pontos:**

	Excelente	Alto	Bom	Médio	Baixo
LPS					
LPSD					
Modelo de features					
Modelagem de software					
Modelagem de aspectos dinâmicos					
Modelagem de PNFs					

APÊNDICE C – DESCRIÇÃO DAS TAREFAS

1. Instruções

Este é um estudo de observação, por isso, sempre que possível, verbalize seus pensamentos, para que o experimentador possa melhor avaliar os resultados obtidos. Pergunte e comente tudo que achar necessário.

2. Contextualização

A modelagem de *features* tem sido amplamente utilizada para modelagem de variabilidade de sistemas. Um dos grandes desafios para se construir Linha de Produtos de Software Dinâmicas (LPSDs) é desenvolver um mecanismo para incorporar cenários de adaptações de contextos e Propriedades Não-Funcionais (PNFs). A representação de PNFs em LPSDs é uma atividade complexa, uma vez que dado um determinado contexto de execução devem ser consideradas regras de configuração, restrições de *features* e preferências dos *stakeholders*, especialmente no que se refere aos RFs e RNFs ou atributos de qualidade do sistema.

O modelo de *features* é um dos principais artefatos de uma Linha de Produto de Software (LPS) e LPSD, sendo utilizado como ponto de partida para a configuração e instanciação de novos produtos, além de servir como interface de comunicação padronizando o entendimento do domínio entre os *stakeholders* envolvidos no processo de desenvolvimento de uma LPS e LPSD.

Você recebeu parte do modelo de *features* da LPS MobiLine, do domínio de Guia de Visitas Móvel. Este produto possui a capacidade de se adaptar às variações de contexto durante sua execução. As regras de composição estão previstas no modelo de *features* recebido.

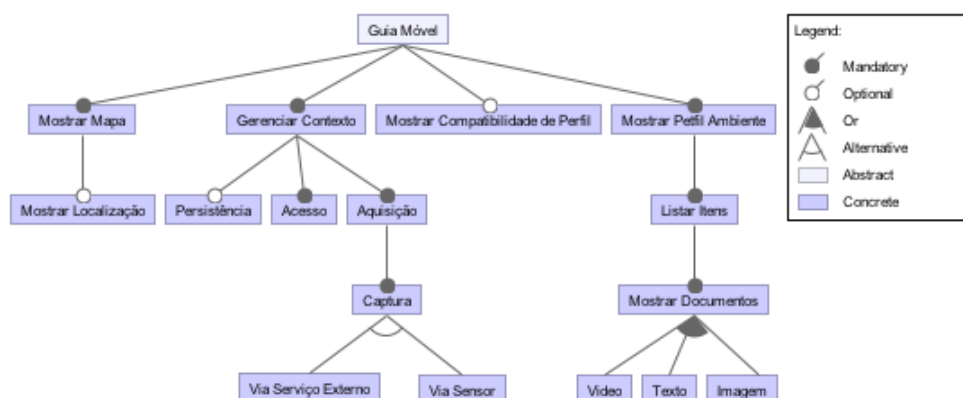
3. Tarefas

- a) Sua tarefa inicial é analisar o modelo de *features* deste produto e identificar as Propriedades Não-Funcionais (PNFs) relevantes para o modelo de *features* apresentado. Para realizar esta atividade você deve:
 - Analisar cada característica e subcaracterística da norma ISO/IEC 25010 (SQuaRE), apresentadas no Anexo A;
 - Em seguida você deve identificar as PNFs que devem ser atendidas para o

domínio do modelo de *features* apresentado, com base na descrição do cenário de uso e com o auxílio do Catálogo para identificação de PNFs, apresentados respectivamente nos Apêndices E e F.

- Por fim, você deve representar cada característica, subcaracterística e PNFs identificadas, no modelo de *features* de qualidade exemplificado no Apêndice F.
- b) A sua próxima tarefa é identificar as restrições de ativação e de desativação das *features* de acordo com os cenários de adaptações de contexto. Para realizar esta tarefa você deve:
- Analisar as regras de contexto que serão executadas com base nas definições dos cenários de adaptações de contexto. As regras e definições dos cenário de adaptação de contexto são apresentados no Apêndice G;
 - Em seguida você deve identificar quais *features* são ativadas e desativadas com base nas regras de contexto executadas, com o auxílio do preenchimento do *Template* apresentado no Apêndice H.
- c) A última tarefa é identificar as restrições de interdependências entre as PNFs identificadas na tarefa inicial e as *features* ativadas e desativadas de acordo com cada cenário de adaptação de contexto. Para realizar esta tarefa você deve:
- Analisar o *Template* preenchido na tarefa anterior, de modo que para cada *features* ativadas e desativadas no cenário de adaptação de contexto analisado, você deve especificar quais links de contribuições uma *feature* têm com relação a uma ou mais PNFs quando esta *feature* é ativada e desativada. As especificações das restrições de interdependências devem ser realizadas com o auxílio do *Template* apresentado no Apêndice I.

APÊNDICE D – DESCRIÇÃO DO MODELO EXEMPLO



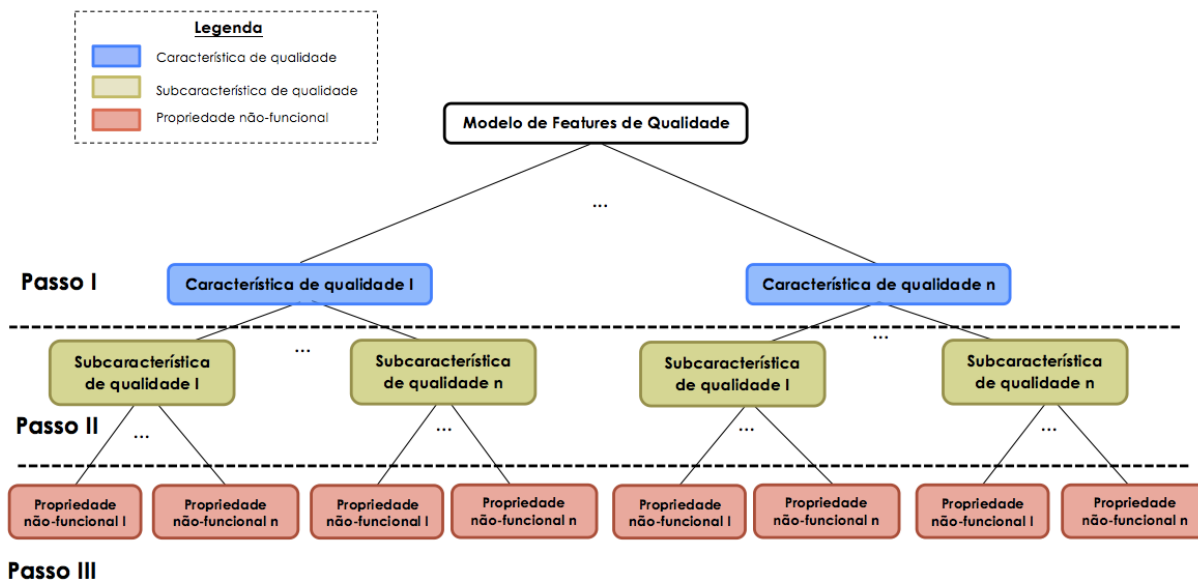
A Figura acima ilustra parte do produto do Guia de Visita Móvel (MobiLine), este produto pode mostrar os mapas do ambiente e a compatibilidade do perfil do visitante. Além disso, gerencia o contexto e exibe o perfil do ambiente em que o visitante se encontra. A persistência, assim como o vídeo e a imagem, possuem atributos que definem, respectivamente, a quantidade de memória disponível para armazenar as informações do contexto e a resolução das imagens e vídeos.

APÊNDICE E – DESCRIÇÃO DO CENÁRIO DE USO

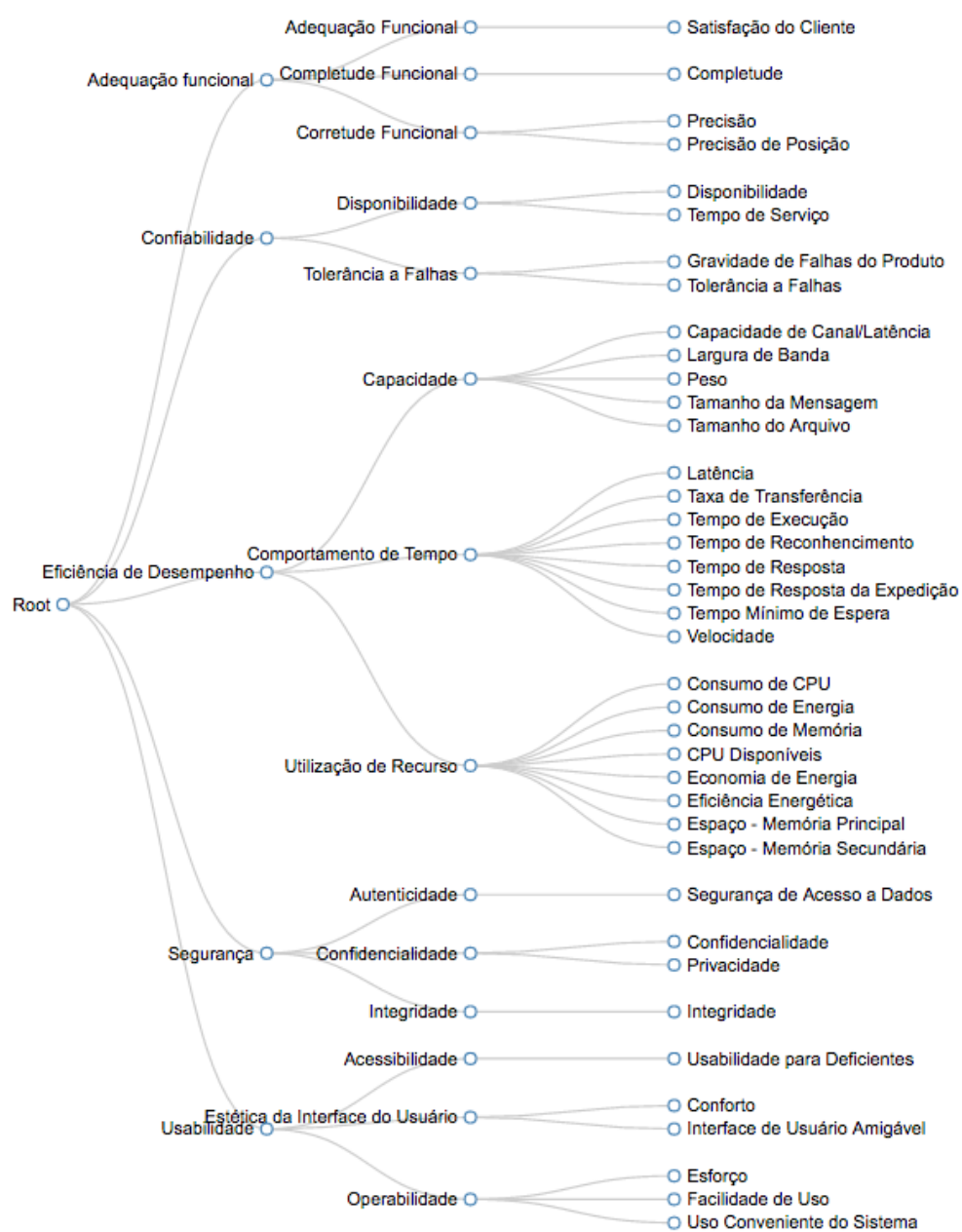
Imagine que você exercer o papel de um engenheiro de domínio contratado por uma empresa de desenvolvimento de software. Essa empresa possui uma linha de produtos para o desenvolvimento de sistemas de Guia de Visitas Móvel. Originalmente, essa linha de produtos não especifica propriedades não-funcionais, mas por exigências do mercado, foi necessário incluir tais propriedades. Para isso, você deve identificar as possíveis propriedades não-funcionais de acordo com o seguinte cenário de uso do Guia de Visitas Móvel MobiLine.

A Universidade Federal do Ceará em Quixadá, está realizando um evento para que pesquisadores possam conhecer o laboratório de Engenharia de Software na área de Linha de Produtos de Software. No entanto, a universidade não possui guias para conduzir a apresentação aos pesquisadores, desse modo para poder conhecer o laboratório os pesquisadores visitantes devem acessar e utilizar de forma fácil e intuitiva o Guia de Visitas Móvel. Para acessar aplicação, a interface do usuário mostra a realização da autenticação. Uma vez com o usuário logado, a aquisição de contexto é iniciada. A aplicação apresenta ao usuário a sua localização atual. Quando o usuário troca a sua posição no ambiente, a aplicação reconhece de forma precisa e rápida a nova localização do usuário e mostra um outro mapa indicando a sua nova localização. Em sua nova localização uma lista de itens sobre esse ambiente são mostradas em forma de vídeo, texto ou imagem. Caso o usuário esteja em um ambiente com um ou mais usuários que possuem perfis compatíveis em relação a suas preferências, uma lista desses usuários são apresentadas.

APÊNDICE F – REPRESENTAÇÃO DO MODELO DE *FEATURES* DE QUALIDADE



F.1 CATÁLOGO DE PROPRIEDADES NÃO-FUNCIONAIS



APÊNDICE G – DEFINIÇÕES DE CENÁRIOS DE ADAPTAÇÕES DE CONTEXTO

CENÁRIOS DE ADAPTAÇÕES					
Contextos	Informações de Contexto	Qualificadores	Quantificadores	Status Quantificadores C1	Status Quantificadores C2
Computacional	Latência	Alto	≥ 200 m/s	✓	×
		Médio	≥ 60 /ms e <200 m/s	×	×
		Baixo	≤ 59 m/s	×	✓
Computacional	Nível de Memória Livre	Normal	≥ 128	✓	×
		Baixo	<128	×	✓
Computacional	Bibliotecas Disponíveis	True	Não se aplica	✓	×
		False	Não se aplica	×	✓
Computacional	Nível de Cobertura de Localização	Alto	\leq Raio do ambiente	×	✓
		Baixo	$>$ Raio do ambiente	✓	×
Usuário	Similaridade de Perfil	True	Não se aplica	✓	×
		False	Não se aplica	×	✓
Físico	Ambiente do Usuário	True	Não se aplica	✓	×
		False	Não se aplica	×	✓
Computacional	Conexão com internet	True	Não se aplica	✓	×
		False	Não se aplica	×	✓

G.1 REGRAS DE CONTEXTO

RC 1

(Mesmo Ambiente de Usuário AND Similaridade de Perfil) implica Mostrar Compatibilidade de Perfil

RC 2

(NOT (Mesmo Ambiente de Usuário) AND NOT (Similaridade de Perfil)) implica NOT (Mostrar Compatibilidade de Perfil)

RC 3

Biblioteca Disponível implica (Texto AND Imagem AND Video)

RC 4

NOT (Biblioteca Disponível) implica (Texto AND Imagem)

RC 5

Nível de Cobertura de Localização Alto implica Mostrar Localização

RC 6

Nível de Cobertura de Localização Baixo implica NOT (Mostrar Localização)

RC 7

(Conexão com internet AND Latência Baixa) implica Via Serviço Externo Internet

RC 8

NOT (Conexão com internet) implica Via Sensor

RC 9

(Conectado Internet AND Latência Alta) implica Via Sensor

RC 10

Nível de Memória disponível Normal implica Persistência

RC 11

Nível de Memória disponível Baixo implica NOT (Persistência)

Cenários	Regras de contexto			
C1	RC 1		RC 7	
	RC 2		RC 8	
	RC 3		RC 9	
	RC 4		RC 10	
	RC 5		RC 11	
	RC 6			
C2	RC 1		RC 7	
	RC 2		RC 8	
	RC 3		RC 9	
	RC 4		RC 10	
	RC 5		RC 11	
	RC 6			

APÊNDICE H – *TEMPLATE* DE IDENTIFICAÇÃO DE RESTRIÇÕES DE ATIVAÇÃO E DE DESATIVAÇÃO DE ACORDO COM CADA CENÁRIO DE ADAPTAÇÃO DE CONTEXTO

[illegible]

APÊNDICE J – FORMULÁRIO DE AVALIAÇÃO DA ABORDAGEM ReMINDER

Este formulário tem como objetivo coletar o *feedback* de cada participante do experimento controlado para realizar uma análise qualitativa da aplicação da abordagem.

1. Você concorda que a utilização da abordagem ReMINDER auxiliou na identificação de PNFs e suas restrições ? Justifique a resposta.

- (a) Concordo totalmente
- (b) Concordo parcialmente
- (c) Indeciso
- (d) Discordo em parte
- (e) Discordo totalmente

2. Você concorda que a utilização da abordagem ReMINDER auxiliou na especificação de cenários de adaptações de contexto e suas restrições ? Justifique a resposta.

- (a) Concordo totalmente
- (b) Concordo parcialmente
- (c) Indeciso
- (d) Discordo em parte
- (e) Discordo totalmente

3. Na sua percepção, qual foi o esforço para identificação das PNFs de acordo com as características e subcaracterísticas de qualidade da norma SQuaRE, por meio do Catálogo de RNFs ? Justifique a resposta.

- (a) Alto

(b) Médio

(c) Baixo

4. Na sua percepção, qual foi o esforço para especificação das restrições de interdependências entre as PNFs identificadas e as *Features* ? Justifique a resposta.

(a) Alto

(b) Médio

(c) Baixo

5. Em relação a proposição de um *template* para Identificação de Restrições de ativação e desativação de acordo com cada cenário de adaptação de contexto. Justifique a resposta.

(a) Não trouxe qualquer benefício para a identificação das restrições de ativação e desativação

(b) Dificultou o andamento da identificação das restrições de ativação e desativação

(c) Facilitou o andamento da identificação das restrições de ativação e desativação

6. Em relação a proposição de um *template* para identificação de Restrições entre *Features* e PNFs em um determinado cenário de adaptação de contexto. Justifique a resposta.

- (a) Não trouxe qualquer benefício para a identificação das restrições entre *Features* e PNFs
- (b) Dificultou o andamento da identificação das restrições entre *Features* e PNFs
- (c) Facilitou o andamento da identificação das restrições entre *Features* e PNFs

7. Em relação a proposição de um *template* para a identificação de cenários de adaptações de contexto. Justifique a resposta

- (a) Não trouxe qualquer benefício para a identificação de cenários de adaptações de contexto
- (b) Dificultou o andamento da identificação de cenários de adaptações de contexto
- (c) Facilitou o andamento da identificação de cenários de adaptações de contexto

8. Quais são as vantagens e desvantagens da abordagem ReMINDER ?

9. Quais sugestões você teria para melhorar a abordagem ReMINDER ?

APÊNDICE K – DESCRIÇÃO DAS TAREFAS - TESTE DE USABILIDADE

Essa lista de tarefas tem como objetivo avaliar a usabilidade das extensões adicionadas a ferramenta DyMMer. Para realizar o conjunto de tarefas abaixo você deve utilizar como dados de entrada os *templates* preenchidos durante o estudo de observação.

1. **Editar o modelo de *features* existente:** para editar um modelo de *features* existente é necessário selecionar o modelo de *features* MobiLine apresentado na lista de modelos na tela inicial da ferramenta. Após selecionar o modelo você deve clicar no botão “Edit”. Em seguida, o modelo de *features* é apresentado por meio de uma aba como o nome do modelo, no lado superior esquerdo da ferramenta. Deve-se clicar nesta aba.
2. **Criar modelo de *features* de qualidade:** para criar esse modelo é necessário inserir cada características e subcaracterísticas de qualidade, e associar as propriedades não-funcionais as subcaracterísticas de qualidade adicionadas. Os dados de entrada pra realizar esta atividade estão no *template* preenchido durante o estudo de observação.
 - **Adicionar característica de qualidade:** para adicionar uma característica de qualidade você deve, visualizar o modelo de *features* de qualidade, que se encontra no centro da ferramenta, entre o modelo de *features* e os cenários de adaptações de contexto. Em seguida você deve, selecionar com o botão direito do mouse em cima da *feature* com o nome “Root” no modelo de *features* de qualidade e selecionar a opção “Add”. Logo após, você deve selecionar uma das características de qualidade listadas e selecionar a opção “Add”.
 - **Adicionar subcaracterística de qualidade:** para adicionar uma subcaracterística de qualidade você deve, selecionar com o botão direito do mouse em cima da característica de qualidade previamente adicionada e selecionar a opção “Add”. Logo após, você deve selecionar uma das subcaracterísticas de qualidade listadas e selecionar a opção “Add”.
 - **Adicionar propriedades não-funcionais:** para adicionar uma propriedade não-funcional você deve, selecionar com o botão direito do mouse em cima da subcaracterística de qualidade previamente adicionada e selecionar a opção “Add”. Logo após, você deve selecionar uma das propriedades não-funcionais listadas. Em seguida, você deve especificar o nível que esta propriedade não-funcional deve ser

atingida, por meio de uma caixa de seleção, inicialmente com valor “Normal”. Por fim, você deve selecionar a opção “Add”.

3. **Definir cenário de adaptação de contexto:** Para definir um cenário de adaptação de contexto é necessário inserir cada informação de contexto com seus respectivos quantificadores e selecionar os quantificadores que definem o cenário de adaptação. Os dados de entrada pra realizar esta atividade estão no *template* preenchido durante o estudo de observação.

- **Adicionar informação de contexto:** para definir uma informação de contexto, você deve visualizar a área de edição dos cenários de adaptações de contexto, que se encontra do lado direito do modelo de *features* de qualidade. Em seguida você deve, selecionar com o botão direito do mouse em cima da *feature* com o nome “Context adaptation” no modelo de *features* de qualidade e selecionar a opção “Add”. Logo após, você deve digitar o nome da informação de contexto e selecionar a opção “OK”.
- **Adicionar Quantificador:** para adicionar um quantificador você deve, selecionar com o botão direito do mouse em cima da informação de contexto previamente adicionada, e selecionar “Add standard quantification” ou “Add boolean quantification”. Para adicionar um qualificador do tipo booleano você deve selecionar a opção “Add boolean quantification” caso contrário a opção “Add standard quantification”. Em seguida, é apresentado um janela onde você deve especificar o quantificador.
 - **Especificando quantificador padrão:** para especificar um quantificador padrão, primeiramente você deve selecionar uma o tipo de qualificação. Os possíveis tipos de qualificação são apresentador por meio de uma lista de seleção, do lado superior esquerdo, cujo valor inicial é “Normal”. Após determinado o tipo de qualificação, você deve especificar o padrão dos valores associados a esse tipo, se é um valor único ou um intervalo de valores. Para especificar o padrão dos valores você deve selecionar um das opções apresentadas por meio de uma lista de seleção, do lado inferior esquerdo, cujo valor inicial é “Normal”. Em seguida você deve, especificar os operadores relacionais caso ele se aplicar a sua especificação. Os operadores relacionais são apresentados por meio de uma lista de seleção, do lado esquerdo da lista de, cujo valor inicial é “Normal”. Por fim, você deve seleciona a opção “Add”.

- **Especificando quantificador booleano:** para especificar um quantificador booleano, primeiramente você deve selecionar uma o tipo de booleano, se é *true* ou *false*. Esses valores podem ser especificados por meio de uma lista de seleção, do lado superior esquerdo, cujo valor inicial é “*True*”. Após determinado o tipo de booleano, você deve opcionalmente especificar o tipo de quantificação. Para especificar a quantificação você deve selecionar a opção “Applicable” apresentada por meio de uma lista de seleção, do lado direito, da especificação do tipo de booleano. Por fim, você deve seleciona a opção “Add”.
 - **Selecionar quantificadores do cenário:** para selecionar os quantificadores você deve selecionar um dos quantificadores associados a cada informação de contexto adicionada. Os quantificadores podem ser selecionar por meio a seleção de seu *checkbox*.
4. **Especificar restrições de contexto:** para especificar as restrições de ativação e de desativação das *features* de acordo com o cenário de adaptação de contexto definido anteriormente, você deve primeiramente visualizar a área de edição do modelo de *features*, apresentada do lado esquerdo da ferramenta. Para especificar as restrições de ativação e de desativação, você deve selecionar com o botão direito do mouse em cima do nome da cada *feature* do modelo de *features* apresentado. Em seguida, é apresentado um menu de seleção com um conjunto de opções. Caso o status da *feature* no cenário de contexto previamente definido seja ativada, você deve selecionar a opção “Set as active node”, caso contrário você deve selecionar a opção “Set as deactivate node”.
 5. **Adicionar restrições de interdependências entre *features* e PNFs:** para adicionar uma restrição de interdependência você deve selecionar com o botão direito do mouse em cima do nome da cada *feature* do modelo de *features* apresentado. Em seguida, é apresentado um menu de seleção com um conjunto de opções. As últimas quatro opções do menu de seleção, são referentes aos tipos de restrições que um *feature* pode ter em relação a uma ou mais PNFs. Você deve selecionar um dos tipos de restrições de interdependências. Em seguida, você deve selecionar com o botão direito do mouse em cima da PNF que tem a relação a *feature* e sua restrição de interdependências. As PNFs são apresentadas no modelo de *features* de qualidade. Uma vez selecionada uma PNFs você deve selecionar a opção “Add to Constraint Quality Feature Models”. As restrições em edição são apresentadas do lado inferior direito da ferramenta, por meio de um caixa de

texto, com o nome “Interdependencies Constraints”, para efetivar a adição da restrição você deve selecionar a opção “Add Constraint”.

6. **Adicionar cenário de adaptação de contexto:** para adicionar o cenário de adaptação de contexto você deve selecionar a opção “Add Scenario” apresentada do lado superior direito da ferramenta. Em seguida é apresentado uma mensagem informando que sucesso na adição do cenário.
7. **Visualizar configuração do modelo de *features* por cenário de adaptação de contexto:** para visualizar a configuração do modelo de *features*, você deve selecionar a opção “Preview” apresentada do lado superior direito da ferramenta, abaixo da opção “Add Scenario”. Em seguida uma aba com o nome “Preview” é aberta do lado superior esquerdo da ferramenta, você deve selecionar esta aba. Após selecionado a aba é apresentado a área de visualização do modelo de *features*, do lado esquerdo pode ser observado o modelo de *features* em forma de árvore, sendo possível alterar a sua orientação de visualização por meio da seleção de uma das opções associadas ao “Change Layout” . Do lado direito é apresentado as definições de cenários de adaptações de contexto adicionadas ao modelo de *features*. Logo abaixo, são apresentadas as restrições de composição do modelo e as restrições de interdependências das PNFs.

Para visualizar a configuração do modelo de *features* por cenário de adaptação de contexto você deve selecionar um dos cenários listados. Em seguida, o modelo de *features* tem sua configuração alterada e é novamente apresentado na área de visualização do modelo de *features*.

ANEXO A – CARACTERÍSTICAS E SUBCARACTERÍSTICAS ISO/IEC 25010 (SQUARE)

O modelo SQuaRE possui oito características e trinta e uma subcaracterísticas relacionadas à qualidade interna e externa. As característica e subcaracterística de qualidade apresentadas, são descritas a seguir:

- **Adequação funcional:** grau em que um produto ou sistema fornece funções que correspondam às necessidades explícitas e implícitas quando utilizado sob as condições especificadas.
 - Completude funcional: grau no qual um conjunto de funções abrange todas as tarefas especificadas e os objetivos do usuário.
 - Corretude funcional: grau no qual um produto ou sistema fornece resultados corretos, com um determinado grau de precisão.
 - Funcionalidade apropriada: grau em que as funções facilitam a realização das tarefas e dos objetivos para os quais o sistema foi especificado.
- **Confiabilidade:** grau em que um sistema, produto ou componente mantém, ao longo do tempo, um comportamento consistente com o esperado, sob as condições especificadas.
 - Maturidade: grau no qual um sistema, produto ou componente satisfaz às necessidades de confiabilidade em sua operação normal.
 - Tolerância a falhas: grau em que um sistema, produto ou componente opera como pretendido, apesar da presença de falhas de hardware ou software.
 - Recuperabilidade: grau em que, em caso de interrupção ou falha, um produto ou sistema pode recuperar os dados diretamente afetados e re-estabelecer o estado desejado do sistema.
 - Disponibilidade: grau em que um sistema, produto ou componente está operacional e acessível quando requisitado para uso.
- **Usabilidade:** grau em que um produto ou sistema pode ser utilizado por usuários específicos para atingir metas especificadas com eficácia, eficiência e satisfação, em um determinado contexto.
 - Conhecimento adequado: grau em que os usuários podem reconhecer se um produto ou sistema é apropriado para suas necessidades.
 - Apreensibilidade: grau em que um produto ou sistema pode ser usado por usuários

para alcançar objetivos específicos de aprendizagem para utilização do produto ou sistema com eficácia, eficiência, inexistência de risco e satisfação, em um contexto de uso especificado.

- Operabilidade: grau de facilidade com a qual um produto ou sistema é operado ou controlado.
- Acessibilidade: grau em que um produto ou sistema pode ser usado por pessoas com a mais ampla gama de características e capacidades, a fim de alcançar um objetivo especificado em um contexto de uso determinado.
- Proteção de erro do usuário: grau em que um sistema protege os usuários de cometer erros.
- Estética da interface de usuário: grau em que uma interface de usuário permite interação agradável e satisfatória.
- **Eficiência de desempenho:** desempenho do produto em relação à quantidade dos recursos utilizados sob condições estabelecidas.
 - Comportamento no tempo: grau em que os tempos de resposta e de processamento e taxas de transferência de um produto ou sistema, no desempenho das suas funções, atende aos requisitos.
 - Utilização de recursos: diz respeito à quantidade de recursos necessários para que um produto ou sistema atenda aos requisitos.
 - Capacidade: grau em que os limites máximos do produto ou sistema satisfazem os requisitos.
- **Manutenibilidade:** grau de eficácia e eficiência com que um produto ou sistema pode ser modificado pela equipe de manutenção.
 - Analisabilidade: grau de eficácia e eficiência com a qual é possível avaliar o impacto sobre um produto ou sistema de uma mudança em uma ou mais de suas partes.
 - Modificabilidade: grau em que um produto ou sistema pode ser modificado de forma eficiente e eficaz sem a introdução de defeitos ou sem degradação de sua qualidade.
 - Modularidade: grau em que um sistema ou programa de computador é composto por componentes discretos, de forma que uma mudança em um componente tem um impacto mínimo sobre os demais.
 - Reusabilidade: grau em que um produto pode ser utilizado em mais do que um sistema, ou na construção de outros produtos.

- Testabilidade: grau de eficácia e eficiência com que critérios de teste podem ser estabelecidos e executados.
- **Portabilidade:** grau de eficácia e eficiência com a qual um sistema, produto ou componente pode ser transferido de um hardware, software ou ambientes de uso.
 - Adaptabilidade: grau em que um produto ou sistema pode eficazmente e eficientemente ser adaptado para um hardware, software ou ambientes de uso diferentes ou em evolução.
 - Instalabilidade: grau de eficácia e eficiência com que um produto ou sistema pode ser instalado e/ou desinstalado com sucesso em num ambiente especificado.
 - Substituibilidade: grau em que um produto pode substituir outro produto de software especificado para o mesmo fim no mesmo ambiente.
- **Segurança:** grau em que as funções e os dados, de um produto ou sistema, são protegidos do acesso não autorizado e o grau em que são disponibilizados para acesso autorizado.
 - Confidencialidade: grau em que um produto ou sistema garante que os dados são acessíveis somente por pessoas autorizadas.
 - Integridade: grau em que um sistema, produto ou componente evita o acesso não autorizado ou a modificação de programas de computador ou dados.
 - Não-repúdio: grau em que um produto ou sistema permite constatar que ações ou acessos foram efetivamente realizados, de forma que não possam ser negados posteriormente.
 - Responsabilização: grau em que as ações de uma entidade podem ser atribuídas exclusivamente a esta entidade.
 - Autenticidade: grau em que a identidade de uma entidade (pessoa ou recurso) pode ser comprovada a quem requisitar.
- **Compatibilidade:** grau em que um produto, sistema ou componente pode trocar informações com outros produtos, sistemas ou componentes, e/ou executar suas funções, enquanto compartilham o mesmo ambiente de hardware ou software.
 - Coexistência: grau em que um produto pode desempenhar as suas funções de forma eficiente ao compartilhar um ambiente e recursos comuns com outros produtos, sem impacto negativo em qualquer outro produto.
 - Interoperabilidade: grau em que dois ou mais sistemas, produtos ou componentes podem trocar informações e utilizar as informações que foram trocadas.

ANEXO B – THE POST-STUDY SYSTEM USABILITY QUESTIONNAIRE (PSSUQ)

Este questionário dá-lhe a oportunidade de nos dizer as suas reações na utilização da DyMMer. Suas respostas nos ajudarão a entender que aspectos da DyMMer você está particularmente preocupado e os aspectos que satisfazem você.

Para o maior grau possível, pense em todas as tarefas que você fez com a ferramenta enquanto responde a essas perguntas. Leia cada declaração e indique o quanto você concorda ou discorda com a declaração.

1. Em geral, estou satisfeito com a facilidade de utilização desta ferramenta.

- (a) Concordo fortemente
- (b) Concordo
- (c) Neutro
- (d) Discordo
- (e) Discordo fortemente

2. Esta ferramenta foi simples de utilizar.

- (a) Concordo fortemente
- (b) Concordo
- (c) Neutro
- (d) Discordo
- (e) Discordo fortemente

3. Consegui completar as tarefas e os cenários utilizando esta ferramenta.

- (a) Concordo fortemente
- (b) Concordo
- (c) Neutro
- (d) Discordo
- (e) Discordo fortemente

4. Consegui completar rapidamente as tarefas e cenários utilizando esta ferramenta.

- (a) Concordo fortemente
- (b) Concordo

- (c) Neutro
- (d) Discordo
- (e) Discordo fortemente

5. Consegui completar as tarefas e os cenários com eficiência utilizando esta ferramenta.

- (a) Concordo fortemente
- (b) Concordo
- (c) Neutro
- (d) Discordo
- (e) Discordo fortemente

6. Senti-me confortável a utilizar esta ferramenta.

- (a) Concordo fortemente
- (b) Concordo
- (c) Neutro
- (d) Discordo
- (e) Discordo fortemente

7. Foi fácil aprender a utilizar esta ferramenta.

- (a) Concordo fortemente
- (b) Concordo
- (c) Neutro
- (d) Discordo
- (e) Discordo fortemente

8. Acredito que me tornaria rapidamente produtivo se utilizasse esta ferramenta

- (a) Concordo fortemente
- (b) Concordo
- (c) Neutro
- (d) Discordo
- (e) Discordo fortemente

9. **A ferramenta deu mensagens de erros que me indicaram claramente como resolver os problemas.**
- (a) Concordo fortemente
 - (b) Concordo
 - (c) Neutro
 - (d) Discordo
 - (e) Discordo fortemente
10. **Sempre que cometi um erro durante a utilização da ferramenta, consegui recuperar de forma fácil e rápida.**
- (a) Concordo fortemente
 - (b) Concordo
 - (c) Neutro
 - (d) Discordo
 - (e) Discordo fortemente
11. **A informação fornecida pela ferramenta (como ajuda online, mensagens no ecrã ou outra documentação) foi clara.**
- (a) Concordo fortemente
 - (b) Concordo
 - (c) Neutro
 - (d) Discordo
 - (e) Discordo fortemente
12. **Foi fácil encontrar a informação que precisava.**
- (a) Concordo fortemente
 - (b) Concordo
 - (c) Neutro
 - (d) Discordo
 - (e) Discordo fortemente

13. A informação fornecida pela ferramenta foi fácil de entender.

- (a) Concordo fortemente
- (b) Concordo
- (c) Neutro
- (d) Discordo
- (e) Discordo fortemente

14. A informação foi eficaz para me ajudar a completar as tarefas e os cenários.

- (a) Concordo fortemente
- (b) Concordo
- (c) Neutro
- (d) Discordo
- (e) Discordo fortemente

15. A organização da informação que a ferramenta transmitiu foi clara.

- (a) Concordo fortemente
- (b) Concordo
- (c) Neutro
- (d) Discordo
- (e) Discordo fortemente

16. A interface da ferramenta foi agradável.

- (a) Concordo fortemente
- (b) Concordo
- (c) Neutro
- (d) Discordo
- (e) Discordo fortemente

17. Gostei de utilizar a interface desta ferramenta.

- (a) Concordo fortemente
- (b) Concordo
- (c) Neutro

- (d) Discordo
- (e) Discordo fortemente

18. Esta ferramenta tem todas as funcionalidades e capacidades que eu esperava.

- (a) Concordo fortemente
- (b) Concordo
- (c) Neutro
- (d) Discordo
- (e) Discordo fortemente

19. Em geral, estou satisfeito com esta ferramenta.

- (a) Concordo fortemente
- (b) Concordo
- (c) Neutro
- (d) Discordo
- (e) Discordo fortemente