

UNIVERSIDADE ESTADUAL DE MARINGÁ
CENTRO DE TECNOLOGIA
DEPARTAMENTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

KLEBER LOPES PETRY

**SMartyTesting: Uma Abordagem de Teste Baseado em Modelos
SMarty para Linha de Produto de Software**

Maringá

2019

KLEBER LOPES PETRY

**SMartyTesting: Uma Abordagem de Teste Baseado em Modelos
SMarty para Linha de Produto de Software**

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação do Departamento de Informática, Centro de Tecnologia da Universidade Estadual de Maringá, como requisito parcial para obtenção do título de Mestre em Ciência da Computação.

Orientador: Prof. Dr. Edson A. Oliveira
Junior

Maringá
2019

SMartyTesting: Uma Abordagem de Teste Baseado em Modelos SMarty para Linha de Produto de Software

RESUMO

A utilização de abordagens de teste no desenvolvimento de software com a finalidade de garantir qualidade e a segurança, vem crescendo exponencialmente entre os modelos de processos nas últimas décadas. Linha de Produto de Software é um modelo de processo onde um grupo de sistemas possuem funcionalidades similares, pois faz reutilização de componentes de software, obtendo maior produtividade, reduzindo tempo e custo. Já as características que diferenciam um produto dos demais em torno deste núcleo, damos o nome de variabilidade, e é ai um dos pontos que devemos levar em consideração quando abordamos teste em LPS, pois se torna um grande desafio realizar a cobertura de todas as funções variáveis de um grupo de produtos, além de que teste exaustivo é inviável. Porém, se faz importante a análise e o gerenciamento da variabilidade como um todo, pois a variabilidade representa diferentes tipos de variações sob diferentes níveis com diferentes tipos de dependências. Sobre Variabilidade, hoje, já existe algumas abordagens de gerenciamento, uma delas é o *Stereotyp-based Management of Variability (SMarty)* ela guia o usuário na identificação e representação de variabilidades, formando um conjunto de estereótipos e meta-atributos para representar variabilidades em modelos UML de uma LPS. A vantagem na utilização do TBM em LPS, seria pelo fato de quanto antes o teste ser realizado, maiores são a chances de garantia de qualidade, pensando nisso o objetivo geral deste trabalho visa a busca por evidências para a criação de uma abordagem utilizando teste baseado em modelo (TBM) em linhas de produtos de software (LPS) que também contenha gerenciamento de variabilidade utilizando a abordagem SMarty e que possam ser empregadas a nível de sistema a partir da engenharia de domínio. O Objetivo seria a conversão do modelo, que pode ser um caso de uso em um diagrama de sequência, a abordagem SMarty para identificação e classificação das variáveis, posteriormente uma possível conversão em Máquina de estado finito MEF e utilizar TBM para a geração dos casos de teste a partir do modelo SMarty, assim dando origem ao perfil *SMartyTesting*. Com ela esperamos resultados positivos no que tange a qualidade em nível de sistema para todos os produtos derivados em uma LPS e como contribuição a utilização de TBM em SMarty.

Palavras-chave: Linha de produto de *Software*. *SMarty*. Teste Baseado em Modelo. Variabilidade. Qualidade de software.

SMartyTesting: An SMarty Model-Based Testing Approach for Software Product Line

ABSTRACT

The use of test approaches in software development to ensure quality and safety has been growing exponentially among process models in recent decades. Software Product Line is a process model where a group of systems have similar functionalities, since it makes reuse of software components, obtaining greater productivity, reducing time and cost. Since the characteristics that differentiate a product from the others around this nucleus, we call it variability, and this is one of the points that we must take into account when approaching test in LPS, since it becomes a great challenge to cover all the functions variables of a product group, in addition to which exhaustive testing is not feasible. However, it is important to analyze and manage the variability as a whole, since the variability represents different types of variations under different levels with different types of dependencies. On Variability, today, there are already some management approaches, one of which is the Stereotyp-based Management of Variability (SMarty), which guides the user in the identification and representation of variabilities, forming a set of stereotypes and meta-attributes for represent variabilities in UML models of an LPS. TBM is a form of software testing where the test cases are derived from a model that describes aspects (usually functional) of the system being tested. The advantage of using TBM in LPS would be because the sooner the test is performed, the greater the chances of quality assurance, considering the general objective of this work is the search for evidence for the creation of a technique using test based (TBM) in software product lines (LPS) that also contain variability management using the SMarty approach and that can be employed at the system level from domain engineering. The objective would be the conversion of the model, which can be a use case in a sequence diagram, the SMarty approach for identification and classification of variables, later a possible conversion in MEF finite state machine and use TBM to generate the cases of test from the SMarty model, thus giving rise to the textit SMartyTesting profile. With it we expect significant results on the system-level quality for all products derived in a LPS and as contribution to the use of TBM in SMarty.

Keywords: Software Product Line. SMarty. Model-Based Testing. Variability. Software Quality.

LISTA DE FIGURAS

Figura - 2.1	Adaptado de (SEI 2010)	16
Figura - 2.2	<i>Framework</i> de Engenharia de LPS (Pohl et al., 2005). Traduzido por Geraldi (2015)	17
Figura - 2.3	O Processo de Gerenciamento de Variabilidades SMartyProcess e sua Interação entre as Atividades com o Processo de Desenvolvimento de LPS, traduzido de (OliveiraJr et al., 2010b)	20
Figura - 2.4	Estereótipos e Meta-Atributos do Perfil SMartyProfile 5.1 com Suporte a Modelos de Casos de Uso, Classes, Componentes, Atividades e Sequência (Fiori et al., 2012; Marcolino, 2014; OliveiraJr et al., 2010a, 2013a).	21
Figura - 2.5	Esquema simplificado que representa o modelo V	22
Figura - 2.6	Modelo de proposta de implantação do processo de teste de software segundo (Crespo et al., 2004)	23
Figura - 2.7	Exemplo de uma MEF utilizado por Costa L. T., 2016	24
Figura - 2.8	Panorama da abordagem SPLiT-MBt por Costa L. T., 2016	30
Figura - 3.1	Etapas da Metodologia de Desenvolvimento de Pesquisa	34
Figura - 3.2	Guia de direcionamento de processo de TBM em LPS	36
Figura - 3.3	Ciclo da etapa 1	40
Figura - 3.4	Ciclo da etapa 2	41
Figura - 1.1	Visão geral do nosso processo de mapeamento sistemático	60
Figura - 1.2	Processo de pesquisa por SMS	62
Figura - 1.3	Processo Seletivo SMS	67
Figura - 1.4	Processo de extração de SMS	74
Figura - 1.5	Processo de Análise SMS	77
Figura - 1.6	Distribuição de estudos por ano	78
Figura - 1.7	Automação TBM para LPS de abordagens de teste	81
Figura - 1.8	Comparação de níveis e tipos de TBM para LPS	85
Figura - 1.9	Automação TBM para LPS de abordagens de teste	87
Figura - 1.10	Automação de níveis de teste de TBM para LPS	87
Figura - 1.11	Automação TBM para LPS de abordagens de teste	88
Figura - 1.12	Artefatos X Níveis X Modelos	91
Figura - 1.13	Tratamento da variabilidade por tipo de solução	93
Figura - 1.14	Tipos de Avaliação de Propostas e Ambientes	95

Figura - 1.15	TBM4LPS: um roteiro de TBM para LPSs	98
Figura - 1.16	Etapa inicial, mostrando a localização do estudo T31	99
Figura - 1.17	Segunda etapa, mostrando a localização do estudo T31	100
Figura - 1.18	Etapa final mostrando a localização do estudo T31	100
Figura - 2.1	Versão 5.2 do <i>SMartyProfile</i> , com suporte a Componentes, Portas, Interfaces e Operações (Bera, 2015).	103
Figura - 2.2	Visão Geral SMarty 5.2 (Bera, 2015).	106
Figura - 2.3	Diagrama de Classes da LPS AGM (OliveiraJr et al., 2010b).	107

LISTA DE TABELAS

Tabela - 2.1	Trabalhos selecionados para leitura e extração de dados	26
Tabela - 1.1	String de pesquisa geral do SMS	62
Tabela - 1.2	Fontes de pesquisa eletrônicas definidas	63
Tabela - 1.3	Conferências Definidas e Workshops para Pesquisa Manual	63
Tabela - 1.4	Diários Definidos para Pesquisa Manual	63
Tabela - 1.5	Pesquisadores que avaliaram o protocolo de SMS	65
Tabela - 1.6	Fontes eletrônicas e sequências de pesquisa adaptadas	66
Tabela - 1.7	Número de estudos de fontes eletrônicas	66
Tabela - 1.8	Número de estudos de fontes manuais	67
Tabela - 1.9	Artigos selecionados para leitura completa	68
Tabela - 1.10	Estudos removidos de acordo com os critérios de exclusão	70
Tabela - 1.11	Estudo do Snowballing	72
Tabela - 1.12	Lista final de estudos	73
Tabela - 1.13	Número de estudos por ano	77
Tabela - 1.14	Avaliação de Qualidade de Estudos	79
Tabela - 1.15	TBM de domínios de aplicativos LPS	80
Tabela - 1.16	Teste de TBM de tipos de solução LPS	83
Tabela - 1.17	Propostas principais de TBM para LPS	84
Tabela - 1.18	Abordagens de TBM para LPS	84
Tabela - 1.19	TBM dos níveis e tipos de testes de LPS	85
Tabela - 1.20	TBM de automação LPS	86
Tabela - 1.21	Artefatos Usados Durante TBM de LPS	88
Tabela - 1.22	Comparação de ferramentas e artefatos primários ou secundários .	90
Tabela - 1.23	TBM de LPS Tipos de Modelos	91
Tabela - 1.24	Estudos que tratam da variabilidade durante as atividades da TBM	92
Tabela - 1.25	Tempo de Ligação de Variabilidade em TBM de LPS	92
Tabela - 1.26	Método de Evidenciação da Solução TBM para LPS	94
Tabela - 1.27	Ambientes de evidência de soluções TBM para LPS	94
Tabela - 1.28	Estudos com Suporte de Rastreabilidade para TBM de LPSs . . .	96

LISTA DE SIGLAS E ABREVIATURAS

AGM: *Arcade Game Maker*

FSM: *Finite State Machine*

LPS: Linha de Produto de Software

RSL: Revisão Sistemática de Literatura

SMarty: *Stereotype-based Management of Variability*

SEI: *Software Engineering Institute*

TBM: Teste Baseado em Modelo

UML: *Unified Modeling Language*

SPLiT-MbT: *Software Product Line Testing Method Based on System Models*

SUMÁRIO

1	Introdução	11
1.1	Contextualização	11
1.2	Motivação	12
1.3	Objetivo	13
1.4	Método de Desenvolvimento	13
1.4.1	Porque Diagramas de Sequência?	13
1.5	Organização do Texto	14
2	Fundamentação Teórica	15
2.1	Considerações Iniciais	15
2.2	Linha de Produto de Software e SMarty	15
2.3	Teste de Linha de Produto de Software (LPS)	22
2.4	Teste baseado em Modelos para LPS	22
2.5	A Abordagem SPLit-MbT	30
2.6	Considerações Finais	31
3	SMartyTesting	32
3.1	Considerações Iniciais	32
3.2	Caracterização da Abordagem SMartyTesting	32
3.2.1	Motivação de Pesquisa	32
3.3	Metodologia de Desenvolvimento	33
3.3.1	Ciclo de Desenvolvimento	35
3.3.2	RoadMap teste TBM em LPS	35
3.3.3	Conversão de Sequência para Atividade	35
3.3.4	Geração de Sequências de Teste	35
3.3.5	Ferramentas Adaptadas (SPLit-MbT)	35
3.3.6	Ferramentas e Caminho Escolhido *****	35
3.3.7	Desafios Encontrados	37
3.4	Especificação da Abordagem SMartyTesting	38
3.4.1	Etapa 1 - Mapeamento de Diagrama de Sequência para Diagrama de Atividade	38
3.4.2	ATLs de Conversão UML	38
3.4.3	Etapa 2 - Geração de Sequência de Teste com apoio da ferramenta SPLit-MbT	40
3.4.4	Solução da Variabilidade na Engenharia de Domínio	41

3.5	Exemplo de Aplicação	42
3.6	Considerações Finais	42
4	Avaliação Empírica de SMartyTesting	43
4.1	Considerações Iniciais	43
4.2	Objetivo	43
4.3	Planejamento	43
4.4	Execução	44
4.5	Análise e Interpretação	44
4.6	Disseminação	44
4.7	Ameaças à Validade	44
4.8	Melhorias Identificadas	44
4.9	Considerações Finais	44
5	Melhorias Identificadas e Lições Aprendidas	45
5.1	Considerações Iniciais	45
5.2	Melhorias	45
5.3	Aprendizado	45
5.4	Considerações Finais	45
6	Conclusão	46
6.1	Contribuições	46
6.2	Limitações	46
6.3	Trabalhos Futuros	46
REFERÊNCIAS		47
A	Teste Baseado em Modelos para LPS:um Mapeamento Sistemático da Literatura	59
A.1	Planejamento da MSL	59
A.1.1	Objetivo da Pesquisa	59
A.2	Metodologia de Pesquisa	59
A.2.1	Objetivo e Questões de Pesquisa	60
A.2.2	Processo de pesquisa	61
A.2.3	Processo de seleção	66
A.2.4	Processo de Extração	74
A.2.5	Processo de Análise	77
A.3	Resultados	77

A.3.1	Caracterização dos Estudos	77
A.3.2	Resultados em questões de pesquisa	80
A.3.3	Procedimentos de compartilhamento de dados do SMS	96
A.4	TBM4LPS:um roteiro para testes baseados em modelos de pesquisa de LPSs	97
A.4.1	Rota Primária Baseada no Estudo	99
A.4.2	Rota baseada em critérios	99
A.5	Observações Conclusivas	100
B	Anexo: Abordagem <i>SMarty</i>	102

Introdução

O mercado se torna cada vez mais competitivo, indicadores de investimento contabilizam e determinam a rentabilidade ou apresentam resultados desastrosos. Baseado nesse cenário, agilidade no desenvolvimento e qualidade de um produto se tornam um fator decisivo. Reutilização de código, diminuição de etapas no processo final são meios que podem ser utilizados para se aumentar a produtividade e rentabilidade, mas até que ponto isso é levado em prática.

1.1 Contextualização

Diversas abordagens têm sido desenvolvidas com propósito de aumentar a reusabilidade de software e, consequentemente, o retorno de investimento (ROI) (Delamaro et al., 2017). Entre essas abordagens, está Linha de Produto de Software (LPS). LPS é uma alternativa de utilização como processo de desenvolvimento baseado em reuso de software, para proporcionar maior produtividade, redução de custo, tempo, risco e proporcionar maior qualidade ao produto.

Uma LPS é um conjunto de sistemas de software que compartilham características (*feature*) comuns e gerenciáveis, que satisfazem as necessidades de um segmento particular ou de uma missão (Clements e Northrop, 2002). Esse conjunto de sistemas é denominado também, família de produtos.

Uma LPS possui artefato que podem ser reutilizados, assim, tendo em vista esse novo desafio de gerenciamento durante o desenvolvimento de software nota-se que em publicados ***verificar fontes das publicações e adicionar aqui**** onde, os artefatos

produzidos nos processos tradicionais de desenvolvimento de software já não são tão eficientes para o contexto de LPS, pois em sua maior parte não possui suporte a variabilidade. Assim, visando contornar este problema, **algumas abordagens foram propostas***citar as abordagens***** para o gerenciamento de variabilidade em LPS.

Uma das abordagens é a *Stereotype-based Management of Variability (SMarty)* (OliveiraJr et al., 2010a) , que é composta de um perfil UML***exemplificar melhor essa parte***, o *SMartyProfile*, e do processo denominado *SMartyProcess*. *SMarty* tem como objetivo permitir que as variabilidades de uma LPS possam ser gerenciadas de forma clara e explícita em modelos UML, e guia o usuário por meio do *SMartyProcess* na identificação e representação de variabilidades em tais modelos ou artefatos, que pode ser de caso de uso, classe, sequência, atividade e componentes de uma LPS.

SMarty possui também um perfil de inspeção de software chamado *SMartyCheck*, que visa a remoção de erros primários ou situações ligadas a requisitos. Embora é feito a verificação se faz importante a validação onde possa garantir uma maior qualidade dos modelos, para isso se faz pertinente testá-los.

1.2 Motivação

Desta forma, teste de software é uma abordagem essencial para a contribuição da geração de qualidade em um produto, o grande desafio é realizar isso em uma LPS, mesmo que ela seja uma abordagem que proporciona um padrão de processo o que permite gerar qualidade. Esse desafio se deve a grande quantidade de derivações de produtos que uma LPS pode permitir, também devemos levar em consideração que teste exaustivo é inviável (do Carmo Machado et al., 2014).

Outro fator que devemos levar em consideração quando se trata de teste e qualidade, é o inicio do ciclo de teste, pois, quanto mais cedo o ciclo se inicia, grandes são as probabilidades de uma maior cobertura de erros. Onde um problema descoberto muito tarde pode gerar um prejuízo muito maior do que se ele fosse detectado no inicio.

Assim, visualizando tais situações citadas, teste baseado em modelo (TBM) pode ser usado em conjunto com LPS, visando a qualidade do produto de software e a garantia de uma estrutura com maior reuso **como garantir esse reuso**. Nesse caso, surge oportunidade de investigar a possibilidade de criação de uma técnica de TBM em associação com a abordagem *SMarty*, em que o teste de software em nível de modelagem onde possa ser reaproveitado em tempo de aplicação se apresente como algo interessante para garantir um certo nível de cobertura do núcleo de artefatos da LPS. Visualizando a utilização de LPS sob o conceito de engenharia de domínio e considerando a abordagem *SMarty*, fica

evidente a possibilidade de utilização de TBM em nível de funcionalidade, sob o domínio de modelagem e posteriormente o de aplicação onde poderia se permitir uma melhor visão sobre as variabilidades do produto de uma LPS.

Teste em LPS é sempre um desafio, devido ao fator variabilidade **explicar melhor porque o teste é um desafio quando se fala em variabilidade** (do Carmo Machado et al., 2014; Chen et al., 2009a; Chen e Babar, 2011; Engström e Runeson, 2011). Neste caso, pensando em TBM, o teste em LPS é iniciando mais cedo, em tempo de modelagem, auxiliando na conversão do modelo em artefatos que possam ser reutilizados mais tarde, permitindo a identificação de variabilidade preservando suas características para possíveis soluções em tempo de aplicação. (Lamancha et al., 2009a, 2010a; Reales et al., 2011a). Por esse motivo, ele se candidata a ser uma técnica promissora para utilização em conjunto com *SMarty*.

1.3 Objetivo

O objetivo deste trabalho é, portanto, especificar uma abordagem de TBM para modelos LPS modelados de acordo com a abordagem *SMarty*. Para tanto, deve-se considerar a variabilidade explícita em tais modelos, especialmente os modelos comportamentais como diagramas de sequência *State Machine*. Espera-se, assim, gerar sequências de teste e derivar cenários de teste para a criação de casos de teste a nível de sistema ou funcionalidade. Sendo assim, espera-se responder à seguinte questão de pesquisa: **É possível utilizar técnicas de TBM para se testar modelos *SMarty* considerando variabilidade?**

1.4 Método de Desenvolvimento

Analizando por essa perspectiva, **acredita-se** que exista a necessidade de se realizar uma pesquisa aprofundada para se obter o estado da arte sobre TBM aplicado em LPS com foco em variabilidade, verificar também a automatização do processo de geração, níveis de cobertura e rastreabilidade.

1.4.1 Porque Diagramas de Sequência?

Um terceiro ponto é a utilização de diagramas de sequência como ponto de partida, Marcolino et al., 2014b estruturou a ampliação do perfil *SMarty* para suporte a diagramas de sequência, um modelo que possui uma maior quantidade de informações comparado

com um diagrama de atividade por exemplo, com ele é possível a representação de laços de repetições assim como condicionais, assim quando convertido para um outro modelo de ele poderá manter as propriedade de estados, esperando assim gerar uma sequencia de teste maior e mais abrangente comparando com outros trabalhos que utilizem artefatos mais alto nível para comparativo com o trabalho de Costa L. T., 2016.

1.5 Organização do Texto

O trabalho está organizado como segue. No capítulo dois é apresentada a fundamentação da área abordada, linha de produto de software e *SMarty*, Teste Baseado em Modelo (TBM) e a abordagem SPLit-MbT que será base para este trabalho. O Capítulo três descreve a proposta assim como as ferramentas e meios para a aplicação e utilização de *SMartyTesting*. O Capítulo quatro apresenta a avaliação realizada com a abordagem e o capítulo cinco finaliza com as conclusões. No apêndice são apresentados um mapeamento sistemático da literatura sobre TBM em LPS e um anexo sobre *SMarty*.

Fundamentação Teórica

..

2.1 Considerações Iniciais

Este capítulo apresenta a fundamentação teórica para a compreensão de Linha de Produto de Software (LPS) e a abordagem de gerenciamento de variabilidade *SMarty*, sobre a abordagem *SMarty* pode ser encontrado mais material no Anexo B. Os conceitos que norteiam este trabalho também são abordados neste capítulo, são eles; teste em LPS, teste baseado em modelo (TBM) para LPS e a abordagem SPLit-MbT que é uma das ferramentas utilizadas nesse trabalho e também é base para comparativo.

2.2 Linha de Produto de Software e *SMarty*

Uma linha de produto de software (LPS) é um conjunto de sistemas que compartilham características comuns e gerenciáveis (Clements e Northrop, 2002) também denominado de família de produtos.

O conceito de LPS tem como principal objetivo o desenvolvimento de produtos de software que se baseia em reutilização, e a migração para uma cultura de desenvolvimento onde novos sistemas são sempre derivados a partir de um conjunto de componentes e artefatos comuns, os quais constituem o núcleo da linha de produtos (Linden et al., 2007).

Dessa maneira, além de componentes do núcleo, uma LPS inclui componentes responsáveis pela implementação de *features* que são necessárias em determinados domínios

ou ambientes de uso. Existem três modelos predominantes para criação de LPS: proativo, reativo e extrativo (Pohl et al., 2005).

- **Proativo** os ativos base são desenvolvidos para depois construir produtos.
- **Reativo** os ativos base já existem e vão sendo evoluídos com incrementos na medida que novos requisitos aparecem.
- **Extrativo** é feita uma análise dos produtos existentes e suas estruturas para poder extrair as características comuns e variáveis para derivar a implantação da LPS.

O *Software Engineering Institute* (SEI) (Institute, 2012), por meio da iniciativa *Product Line Practice* (PLP), define três atividades essenciais em LPS: o desenvolvimento do núcleo de artefatos, correspondente à engenharia de domínio; o desenvolvimento do produto, referente à engenharia de aplicação e o gerenciamento de linha de produto (Figura - 2.1.)



Figura 2.1: Adaptado de (SEI 2010)

- **Engenharia de Domínio** processo em que as similaridades e as variabilidades das LPSs são identificadas e realizadas. No qual, é composto de cinco subprocessos principais, sendo eles: Gerenciamento de Produto, Engenharia de Requisitos do Domínio, Projeto do Domínio, Realização do Domínio e Teste de Domínio;
- **Engenharia de Aplicação** processo em que as aplicações de uma LPS são construídas por meio da reutilização de artefatos de domínio, explorando as variabilidades de uma linha de produto. No qual, é composto pelas subprocessos: Engenharia de Requisitos da Aplicação, Projeto da Aplicação, Realização da Aplicação e Teste da Aplicação.

(Pohl et al., 2005) desenvolveram um *framework* para engenharia de LPS, cujo objetivo é incorporar os conceitos centrais da engenharia de linha de produto tradicional, proporcionando a reutilização de artefatos e a customização em massa por meio de variabilidades (Figura - 2.2).

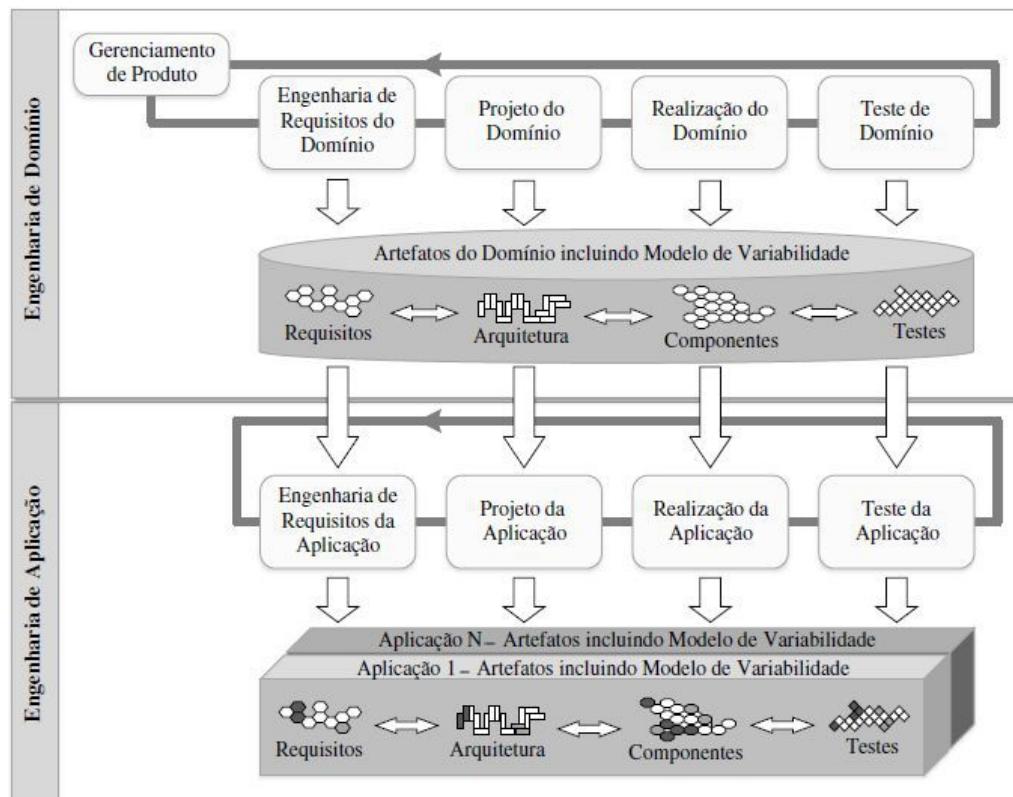


Figura 2.2: *Framework* de Engenharia de LPS (Pohl et al., 2005). Traduzido por Geraldi (2015)

A adoção de LPS traz benefícios em diversos aspectos do processo de desenvolvimento de software, tais como (Linden et al., 2007; Pohl et al., 2005):

- redução dos custos de desenvolvimento, devido ao reúso de artefatos de um núcleo;
- melhoria da qualidade dos produtos, pois os artefatos produzidos são revisados e testados em vários produtos;
- redução do tempo de produção (*time to market*) que é mais alto inicialmente, pois os artefatos comuns devem ser construídos antes. Mas, reduz ao produzir cada novo produto;
- redução do esforço de manutenção, pois quando um artefato do núcleo for modificado, as mudanças podem ser propagadas para todos os produtos que utilizam tal artefato;
- contribuição para evolução, pois ao inserir um novo produto no núcleo da LPS, concede a oportunidade de evolução de todos os tipos de produtos derivados da LPS
- contribuição para redução da complexidade, devido ao crescente número de solicitações de clientes, a complexidade dos produtos aumenta. Dessa forma, mais funcionalidades são adicionadas ao software. O fato de partes comuns serem reusadas por uma LPS ocorre a redução significativa da complexidade;
- melhoria na estimativa de custo, pois a organização pode se concentrar em promover produtos que são fáceis de ser gerados a partir da LPS e produtos que exijam extensões podem ser vendidos por preços mais altos; e
- benefícios para os clientes, pois adquirem produtos adaptados às suas necessidades e expectativas por preços acessíveis devido a abordagem de LPS contribuir na redução de custos de desenvolvimento.

O núcleo de artefatos é composto de um conjunto de características comuns (similaridades) e características variáveis (variabilidades) (Linden et al., 2007). Esse núcleo forma a base da LPS e inclui a arquitetura de LPS, componentes reusáveis, modelos de domínios, requisitos da LPS, planos de testes e modelos de características e de variabilidades.

De acordo com (Apel et al., 2013), "uma característica é um comportamento visível ao usuário final de um sistema de software". Uma característica pode ser obrigatória, opcional ou alternativa. O modelo de características representa as variabilidades de uma LPS. Variabilidades são descritas por: Ponto de variação, o que permite a resolução de variabilidades em artefatos genéricos de uma LPS; Variante, que representa os possíveis elementos que podem ser escolhidos para resolver um ponto de variação e; Restrições entre

variantes que estabelecem os relacionamentos entre uma ou mais variantes com o objetivo de resolver seus respectivos pontos de variação ou variabilidade em um dado tempo de resolução (Apel et al., 2013; Linden et al., 2007; Pohl et al., 2005).

Analizando o contexto apresentado, podemos considerar que a variabilidade é algo muito importante para não ser levado em consideração quando falamos em qualidade de software. Existem várias abordagens para o gerenciamento de variabilidade***citar quais são***, uma delas é a *Stereotype-based Management of Variability (SMarty)*, que realiza o gerenciamento de variabilidades de uma LPS de forma clara e explícita em modelos UML (OliveiraJr et al., 2010a). *SMarty* é composta de um perfil UML, o *SMartyProfile*, e do processo denominado *SMartyProcess*. Ela guia o usuário por meio do *SMartyProcess* na identificação e representação de variabilidades em modelos UML de uma LPS. O perfil *SMartyProfile* é formado por um conjunto de estereótipos e meta-atributos para representar variabilidades em modelos UML de LPS.

Por meio da UML e seu mecanismo de perfil *SMarty* permite a representação explícita de variabilidades. O *SMartyProcess* é um processo sistemático que guia o usuário na identificação, delimitação, representação, rastreamento de variabilidades e análise de configurações de produtos de uma LPS explicar melhor essa figura (Figura - 2.3). e a 2.4 também Nele há um conjunto de diretrizes que permitem ao usuário a aplicação dos estereótipos do *SMartyProfile* de forma clara e objetiva e é possível perceber todos os estereótipos suportados pelo perfil na região inferior da (Figura - 2.4). Mais informações relacionadas a *SMarty* podem ser encontradas no anexo B - Abordagem *SMarty*, assim como um exemplo de modelagem utilizando a abordagem.

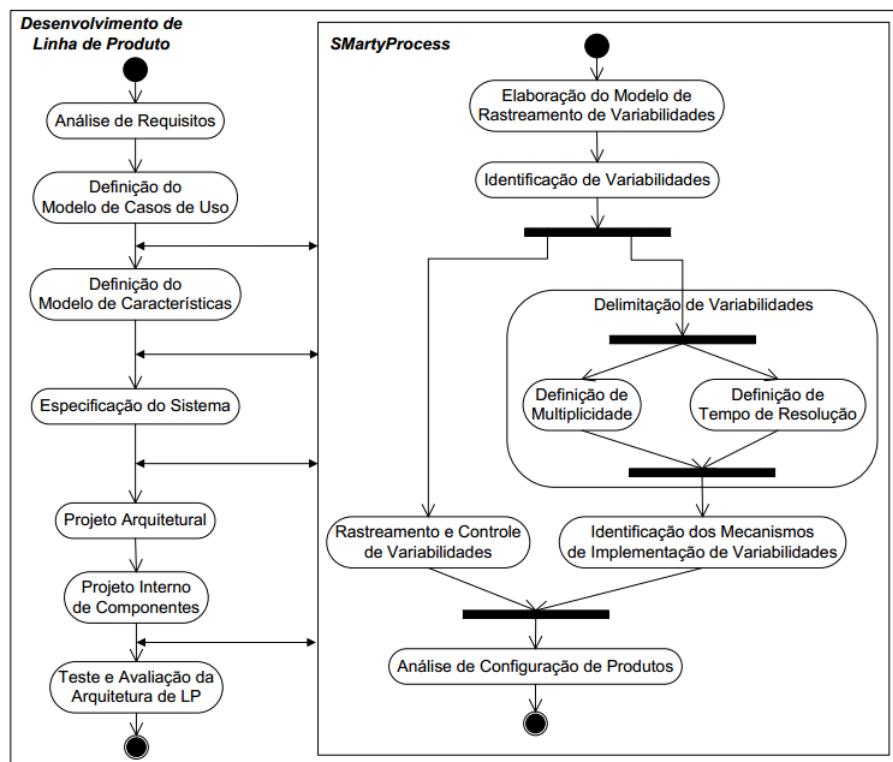


Figura 2.3: O Processo de Gerenciamento de Variabilidades SMartyProcess e sua Interação entre as Atividades com o Processo de Desenvolvimento de LPS, traduzido de (OliveiraJr et al., 2010b)

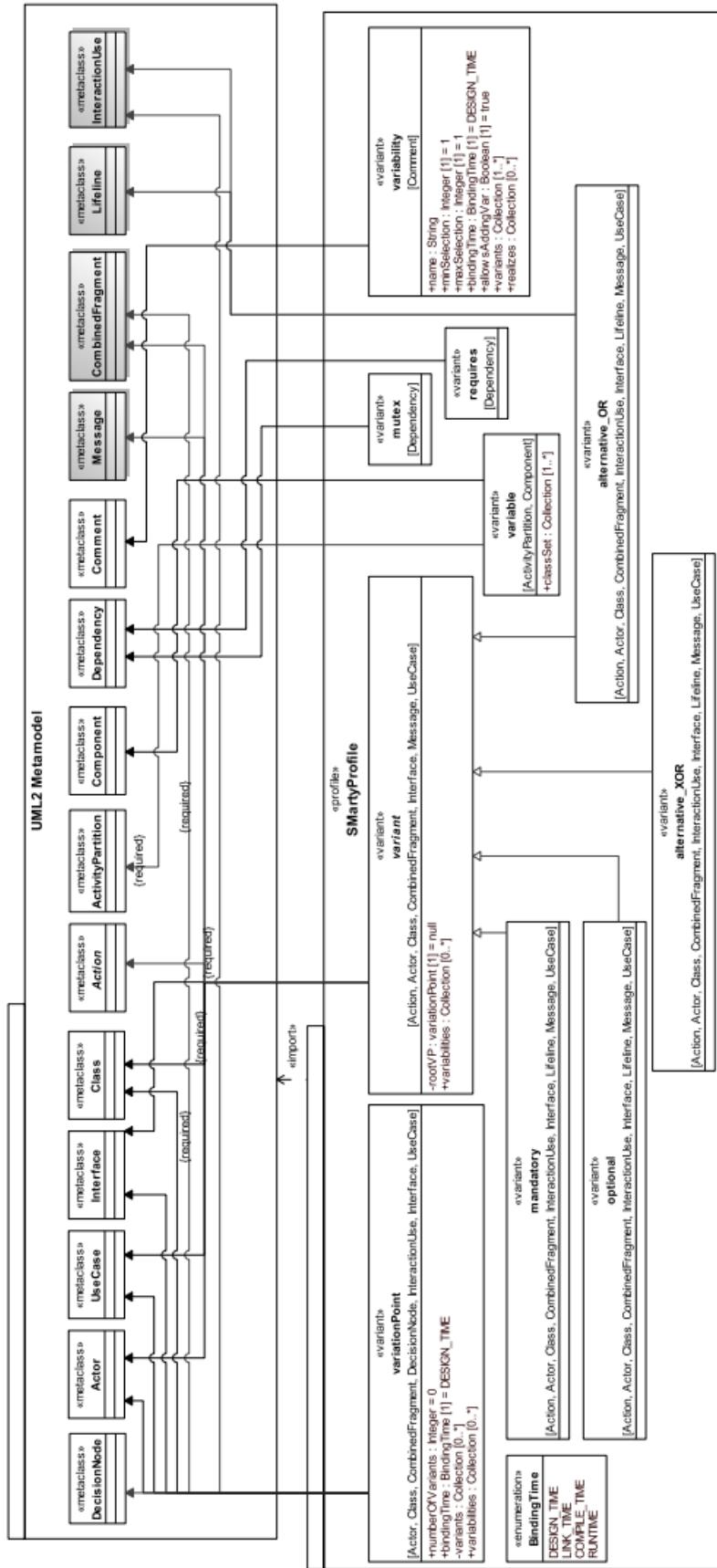


Figura 2.4: Estereótipos e Meta-Atributos do Perfil SMartyProfile 5.1 com Suporte a Modelos de Casos de Uso, Classes, Componentes, Atividades e Sequência (Fiori et al., 2012; Marcolino, 2014; OliveiraJr et al., 2010a, 2013a).

2.3 Teste de Linha de Produto de Software (LPS)

contextualizar aqui os conceitos de teste

citar os principais itens da revisão sistemática que aborda testes em LPS

tentar contextualizar o teste de linha de produto de software com um exemplo em figura

2.4 Teste baseado em Modelos para LPS

LPS é um interesse da indústria pelo o potencial de reuso de artefatos além de aumentar a produtividade. Para alcançar as melhorias pretendidas, a qualidade dos artefatos reutilizáveis deve ser verificado. Portanto, a garantia de qualidade em geral. Testes em particular, ainda é a técnica de garantia de qualidade mais comum na indústria, são cruciais para os esforços de LPS (Delamaro et al., 2017).

Sendo assim, para uma busca completa do desenvolvimento de teste em LPS primeiro temos que entender mais sobre o modelo de processo de teste. Atualmente existem vários modelos, porém, aqui iremos demonstrar um exemplo utilizando o modelo em V Figura - 2.5.

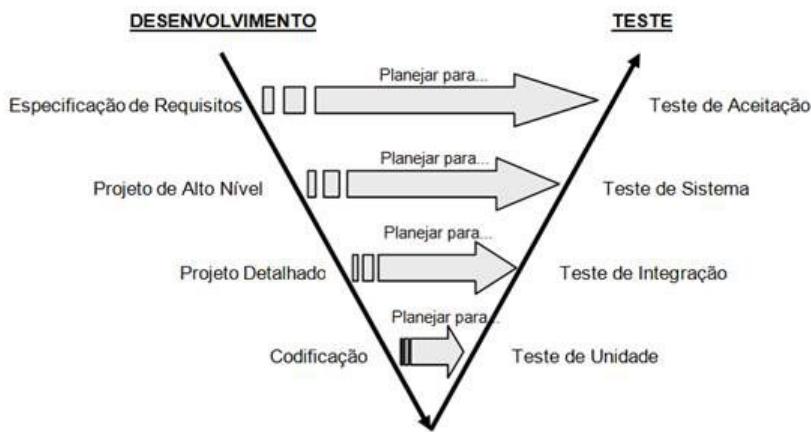


Figura 2.5: Esquema simplificado que representa o modelo V

O modelo V é um modelo conceitual de Engenharia de sistemas/desenvolvimento de produto visto como melhoria ao problema de reatividade do modelo em cascata. Ele permite que, durante a integração de um sistema em seus diversos níveis, os testes sejam feitos contra os próprios requisitos do componente/interface que está sendo testado(a), em contraste com modelos anteriores onde o componente era testado contra a especificação do componente/interface.

Para este trabalho, a intenção é a antecipação dos testes como no modelo V Figura - 2.5 utilizando o modelo de teste consideramos duas etapas, a verificação e a validação. Aqui iremos tratar a validação, e por se tratar de teste de funcionalidade em engenharia de domínio, iremos utilizar a técnica de caixa preta, onde em um processo de teste Figura - 2.6 na preparação do teste sabe-se apenas os parâmetros de entrada não temos acesso a parte interna do processo de execução, apenas o resultado esperado em sua saída para registro posterior do teste.

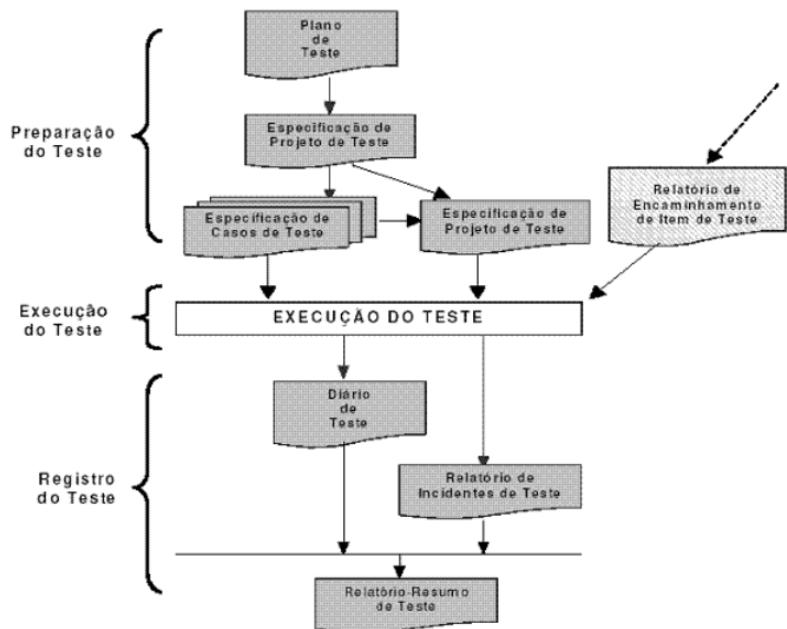


Figura 2.6: Modelo de proposta de implantação do processo de teste de software segundo (Crespo et al., 2004)

Sabendo-se que será utilizando teste caixa-preta, o nível de teste trabalhado poderá ser o de sistema, embora iremos trabalhar com diagramas mais detalhados como o de sequência. Pensando neste sentido de trabalho de teste a nível de engenharia de Domínio, procuramos buscar alternativas de teste que não necessitassem de atuação a nível de código ou caixa-branca.

do Carmo Machado et al., 2014 cita em um trabalho de revisão de literatura relacionando teste em LPS, buscam analisar as estratégias de testes que são abordadas e quais seus potenciais em uma maior taxa de detecção de erros. Nesse caso eles apontam que testes devem ser considerados tanto na engenharia de domínio como na de aplicação. Dentro do interesse de teste dois itens devem ser levados em consideração, o conjunto de requisitos do produto e a qualidade do modelo de variabilidade em teste, isso demonstra que teste a nível de modelagem é uma item a ser pesquisado.

do Carmo Machado et al., 2014 apresentam também um grande número de técnicas para lidar com o aspecto de seleção de produto para teste e o teste real dos produtos. No entanto, faltam relatos reais de experiências industriais que limitam algumas conclusões. O estudo ainda apresenta uma série de estratégias que podem apoiar a seleção e a execução dos testes reais em produtos.

O Teste Baseado em Modelo (TBM) é uma forma de teste de software em que os casos de teste são derivados de um modelo que descreve aspectos (geralmente funcionais) do sistema sendo testado. Tais casos são conhecidos como a suíte abstrata de testes, e seu nível de abstração está intimamente relacionado ao nível de abstração do modelo, segundo do Carmo Machado et al., 2014. As vantagens da abordagem é que a geração de testes começa mais cedo no ciclo do desenvolvimento e pode-se criar casos de teste automaticamente a partir do modelo, isso vai de encontro com nosso interesse, que é a antecipação da etapa de teste.

Os casos de teste podem ser representados por meio de árvores de decisão, *statecharts*, ontologias de domínio ou diagramas de casos de uso e/ou estados da UML (*Unified Modeling Language*) (Isa et al., 2017).

Baseado neste cenário, um dos maiores desafios em teste de LPS se dá em relação as particularidades de cada modelo, para isso TBM busca na criação de modelos dentro da engenharia de domínio, realizar a geração de casos de teste que possam ser reutilizados na engenharia de aplicação, foco da nossa pesquisa. Alguns trabalhos realizados focam na construção da geração antecipada dos teste na modelagem de domínio da LPS, onde, o foco das pesquisas ficam em relação a como conseguir gerar reutilização com uma maior cobertura dos possíveis problemas.

Uma particularidade de TBM é que ele se faz de uso de Máquinas de Estado Finito (MEF) Figura - 2.7, um modelo formal que representa as possíveis configurações do sistema.

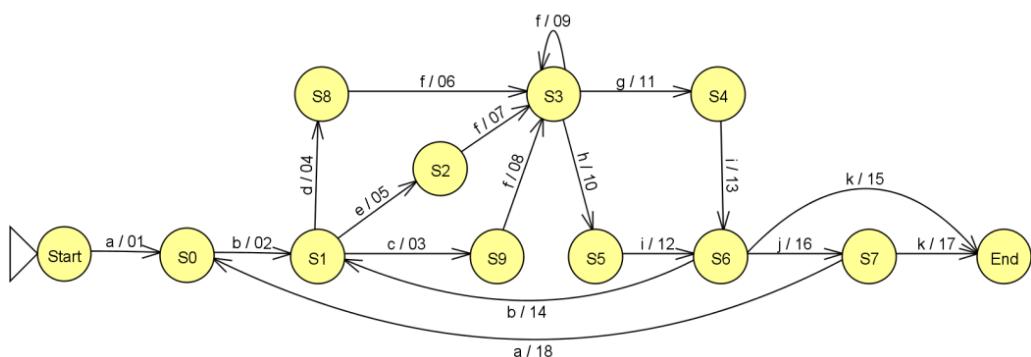


Figura 2.7: Exemplo de uma MEF utilizado por Costa L. T., 2016

Sabendo que TBM possui características na utilização em LPS e que proporciona uma certa relação com variabilidade, preenchendo os principais requisitos para nossa pesquisa, foi conduzido um MSL Seção A, com a finalidade de evidenciar como que TBM esta sendo utilizado em LPS. Onde, ao longo de toda o MSL foram encontrados vários estudos que indicam a utilizam de TBM para LPS com tratativa da variabilidade.

Os estudos **quais estudos reportam do MSL** reportam que, em grande maioria, a partir de um modelo é realizado a conversão para um artefato, normalmente máquina de estado finito estendido, assim, ele pode ser verificado os pontos de variação e variabilidade. Após, gerenciado essa variabilidade com ou sem a utilização de uma ferramenta de apoio. E por fim, a criação do caso de teste, fazendo o uso ou não de rastreabilidade.

Para facilitar a leitura, os estudos utilizados para a extração de dados estão contidos na Tabela - 2.1 que é um conjunto de estudos restante de todo o processo de mapeamento, e que reportam o TBM em LPS, assim foi criado um guia visual na Figura - 1.15 onde apresenta um mapa com o que os trabalhos do MSL reportaram, contido na Anexo A.4 *** **reescrever ***** Porém, caso o leitor queira mais detalhes sobre o guia ou os trabalhos, ele pode consultar o MSL no Anexo A

A Figura - 1.15 Apresenta um compilado de todos os trabalhos resultantes da MS organizados por tópicos chaves, considerando o interesse do leitor, um exemplo, caso o leitor queira saber quais trabalhos que utilizam de gerenciamento de variabilidade ele terá um item no gráfico apontando para este item.

Tabela 2.1: Trabalhos selecionados para leitura e extração de dados

ID	Título	Autor(es)	Ano de Publicação	Fonte	Tipo
T1	Delta-Oriented Model-Based SPL Regression Testing	Sascha Lity, Malte Lochau, Ina Schaefer, Ursula Goltz	2012	ACM	Evento
T2	Industrial Evaluation of Pairwise SPL Testing with MoSo-PoLiTe	Michaela Steffens, Sebastian Oster, Malte Lochau, Thomas Fogdal	2012	ACM	Evento
T3	Model-Based Coverage-Driven Test Suite Generation for Software Product Lines	Harald Cichos, Sebastian Oster, Malte Lochau, Andy Schurr	2011	ACM	Periódico
T4	MoSo-PoLiTe - Tool Support for Pairwise and Model-Based Software Product Line Testing	Sebastian Oster, Ivan Zorcic, Florian Markert, Malte Lochau	2011	ACM	Evento
T5	MPLM - MaTeLo Product Line Manager	Hamza Samih, Ralf Bogusch	2014	ACM	Evento
T6	On the use of test cases in model-based software product line development	Alexander Knapp, Markus Roggenbach, Bernd-Holger Schlingloff	2014	ACM	Evento
T7	Pairwise Feature-Interaction Testing for SPLs: Potentials and Limitations	Sebastian Oster, Malte Lochau, Marius Zink, Mark Grechanik	2011	ACM	Evento
T8	Deriving Usage Model Variants for Model-based Testing: An Industrial Case Study	Hamza Samih, Hélène Le Guen, Ralf Bogusch, Mathieu Acher, Benoit Baudry	2014	IEEE	Evento
T9	Model-based Software Product Line Testing by Coupling Feature Models with Hierarchical Markov Chain Usage Models	Ceren Sahin Gebizli, Hasan Sozer	2016	IEEE	Evento
T10	Model-Based Test Design of Product Lines: Raising Test Design to the Product Line Level	Hartmut Lackner, Martin Thomas, Florian Wartenberg, Stephan Weileder	2014	IEEE	Periódico
T11	Requirements-Based Delta-Oriented SPL Testing	Michael Dukaczewski, Ina Schaefer, Remo Lachmann, Malte Lochau	2013	IEEE	Evento

ID	Título	Autor(es)	Ano de Publicação	Fonte	Tipo
T12	Using Feature Model to Support Model-Based Testing of Product Lines: An Industrial Case Study	Shuai Wang, Shaukat Ali, Tao Yue, Marius Lliaen	2013	IEEE	Periódico
T13	An automated Model-based Testing Approach in Software Product Lines Using a Variability Language	Boni García, Rodrigo García-Carmona, Álvaro Navas, Hugo A. Parada-Gélvez, Félix Cuadrado, Juan C. Dueñas	2010	Politécnica Arquivo digital UPM	Evento
T14	Automated Product Line Methodologies to Support Model-Based Testing	Shuai Wang, Shaukat Ali, Arnaud Gotheb	2013	CEUR Event Proceedings	Evento
T15	Behavioural Model Based Testing of Software Product Lines	Xavier Devroey	2014	ACM	Evento
T16	Feature Model-based Software Product Line Testing	SEBASTIAN OSTER	2012	TUprints	Periódico
T17	Model-based pairwise testing for feature interaction coverage in software product line engineering	Malte Lochau, Sebastian Oster, Ursula Goltz, Andy Schurr	2011	Springer	Periódico
T18	Model-based Test Generation for Software Product Line	Xinying Cai, Hongwei Zeng	2013	IEEE	Evento
T19	Model-Based Testing for Software Product Lines	Erika Mir Olimpiew	2008	Springer	Evento
T20	PLETS - A Product line of model-based testing tools	Elder de Macedo Rodrigues	2013	PUC-RS	Evento
T21	'Top-Down and Bottom-Up' Approach for Model-Based Testing of Product Lines	Stephan Weileder, Hartmut Lackner	2013	EPTCS	Evento
T22	A Product Line Modeling and Configuration Methodology to Support Model-Based Testing: An Industrial Case Study	Shaukat Ali, Tao Yue, Lionel Briand, Suneth Walawege	2012	Springer	Periódico
T23	Coverage Criteria for Behavioural Testing of Software Product Lines	Xavier Devroey, Gilles Perronin, Axel Legay, Maxime Cordy, Pierre-Yves Schobbens, Patrick Heymans	2014	Springer	Evento
T24	A Model Based Testing Approach for Model-Driven Development and Software Product Lines	Beatrix Pérez Lamancha, Macario Polo Usoala, Mario Piattini Veltius	2010	Springer	Evento

ID	Título	Autor(es)	Ano de Publicação	Fonte	Tipo
T25	A Vision for Behavioural Model-Driven Validation of Software Product Lines	Xavier Devroey, Maxime Cordy, Gilles Perrouin, Eun-Young Kang, Pierre-Yves Schobbens, Patrick Heymans, Axel Legay, Benoit Baudry	2012	Springer	Evento
T26	Abstract Test Case Generation for Behavioural Testing of Software Product Lines	Xavier Devroey, Gilles Perrouin, Pierre-Yves Schobbens	2014	ACM	Evento
T27	Applying Incremental Model Slicing to Product-Line Regression Testing	Sascha Lity, Thomas Morbach, Thomas Th um, Ina Schaefer	2016	Springer	Periódico
T28	Automated Testing of Software-as-a-Service Configurations using a Variability Language	Sachin Patel, Vipul Shah	2015	ACM	Evento
T29	Delta-Oriented FSM-Based Testing	Mahsa Varshosaz, Harsh Beohar, Mohammad Reza Mousavi	2015	Springer	Evento
T30	Incremental Model-Based Testing of Delta-oriented Software Product Lines	Malte Lochau, Ina Schaefer, Jochen Kamischke, Sascha Lity	2012	Springer	Periódico
T31	Model Based Testing in Software Product Lines	Pedro Reales, Macario Polo, Danilo Caivano	2011	Springer	Evento
T32	Model-Based Testing	Malte Lochau, Sven Peldszus, Matthias Kowal, Ina Schaefer	2014	Springer	Evento
T33	Parameterized Preorder Relations for Model-Based Testing of Software Product Lines	Malte Lochau, Jochen Kamischke	2012	Springer	Evento
T34	Poster: VTBeS, Transition System Mutation Made Easy	Xavier Devroey, Gilles Perrouin, Pierre-Yves Schobbens, Patrick Heymans	2015	IEEE	Evento
T35	Spinal Test Suites for Software Product Lines	Harsh Beohar, Mohammad Reza Mousavi	2014	EPTCS	Evento
T36	Automated model-based testing using the UML testing profile and QVT	Beatriz Pérez Lamancha, Pedro Reales Mateo, Ignacio Rodríguez de Guzmán, Macario Polo Usaola, Mario Piattini Vethius	2009	ACM	Evento

ID	Título	Autor(es)	Ano de Publicação	Fonte	Tipo
T37	Relating Variability Modeling and Model-Based Testing for Software Product Lines Testing	Hamza Samih	2012	ICTSS	Evento
T38	An Evaluation of Model-Based Testing in Embedded Applications	Stephan Weileder, Holger Schlingloff	2014	IEEE	Evento
T39	Assessing Software Product Line Testing Via Model-Based Mutation An Application to Similarity Testing	Christopher Henard, Mike Papadakis, Gilles Perrouin, Jacques Klein, Yves Le Traon	2013	IEEE	Evento
T40	Automated and Scalable T-wise Test Case Generation Strategies for Software Product Lines	Gilles Perrouin, Sagar Sen, Jacques Klein, Benoit Baudry, Yves le Traon Lassy	2010	IEEE	Evento
T41	Model-based Testing of System Requirements using UML Use Case Models	Bill Hasling, Helmut Goetz, Klaus Beetz	2008	IEEE	Evento
T42	Successive refinement of models for model-based testing to increase system test effectiveness	Ceren Sahin Gebizli, Hasan Sozer, Ali Ozier Ercan	2016	IEEE	Evento
T43	A Software Product Line for Model-Based Testing Tools	Elder M. Rodrigues, Avelino F. Zorzo, Itana M. Gimenez, Elisa Y. Nakagawa, Flavio M. Oliveira, José C. Maldonado	2012	PUC-RS	Outro
T44	Reusing State Machines for Automatic Test Generation in Product Lines	Stephan Weileder, Dehla Sokenou, Bernd-Holger Schlingloff	2008	MoTip	Evento

2.5 A Abordagem SPLiT-MbT

(Costa L. T., 2016) apresenta uma abordagem denominada *Software Product Line Testing Method Based on System Models* (SPLiT-MbT) onde é possível gerar casos de teste funcional e *scripts* para testar os produtos derivados de uma LPS, onde os casos de teste comuns entre os produtos derivados são gerados com base no reuso inerente a LPS.

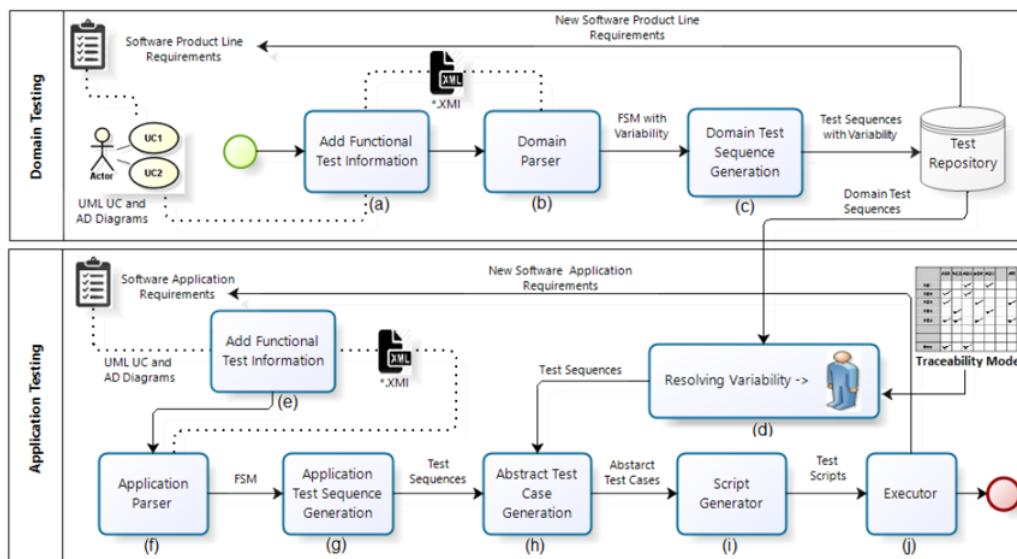


Figura 2.8: Panorama da abordagem SPLiT-MbT por Costa L. T., 2016

O objetivo principal da abordagem é prover o reuso de artefatos de teste baseado na adaptação do uso de teste baseado em modelo (TBM) para geração automática de casos de teste funcional e *scripts* para modelos considerando a variabilidade.

Para fornecer o reuso ele é aplicado em duas etapas vide Figura - 2.8, onde na primeira é realizada a anotação em modelo de sistema, as informações de variabilidade e teste, em seguida utilizadas para gerar sequências de teste usando diferentes métodos formais, como o HSI, UIO,DS ou TT. O modelo formal utilizado é Máquina de Estado Finito e que são estendidas para lidar com as informações sobre variabilidade, uma vez que os modelos de entrada esperado é que já esteja modelado com base na abordagem *SMarty*.

Na engenharia de domínio um diagrama de atividade é convertido em uma Máquina de estado Finitos (MEF) onde é estendida para lidar com informações sobre variabilidade, em seguida se faz a utilização de um modelo formal de geração de sequência de teste denominado HSI. Então essas sequências de teste são armazenadas em um repositório para reutilização, porém, ainda sem a resolução da variabilidade.

Já na segunda etapa, na engenharia de aplicação, os casos de teste são reaproveitados e assim é gerado uma solução para a variabilidade, os modelos de teste e as sequências geradas são utilizadas para gerar os *scripts* de teste que podem ser executados por diferentes ferramentas de teste funcional. Outro fator que deve ser levado em consideração é a rastreabilidade dos casos de teste, que proporciona uma verificação posterior de quais casos de teste foram reaproveitados e se sofreram atualizações. As diferenças entre este trabalho para com o SPLiT-MBt é que *SMartyTesting* dará suporte ao diagrama de sequência que também é suportado por *SMarty*, a escolha por este tipo de modelo é por proporcionar um maior teor de detalhes relacionado aos processos.

2.6 Considerações Finais

Nesta seção não foi abordado alguns temas como tipos, técnicas ou níveis de teste, pois entendemos que o leitor que venha buscar por este assunto já está familiarizado com os princípios básicos de teste de software.

3

SMarty Testing

3.1 Considerações Iniciais

Neste capítulo é apresentado uma abordagem que auxilia na geração de sequências de teste em tempo de modelagem, dentro da engenharia de domínio a partir de modelos UML. Tais sequências de teste poderão ser utilizadas para a geração de casos de teste, que em um segundo momento poderão ser aplicados na engenharia de aplicação. Denominada *SMartyTesting*, um perfil de validação *SMarty*, tem por objetivo auxiliar o arquiteto de software com a geração de sequências de teste a partir de diagramas de sequência, onde o diagrama é convertido para máquina de estado, após este processo é realizado a geração da sequência de teste, isso é feito para todos os produtos derivados de uma linha de produto de software, considerando suas variabilidades.

Para esta seção ser mais produtiva recomendamos a leitura da Seção 2, logo em seguida conhecer mais sobre a abordagem de gerenciamento de variabilidade *SMarty*,

3.2 Caracterização da Abordagem SMartyTesting

3.2.1 Motivação de Pesquisa

Apesar de teste de software ser altamente difundido, assim como o conceito de LPS, sempre encontramos pontos onde devemos tomar atenção com relação ao resultado final do produto, isso impacta diretamente na qualidade e custo de um produto de software, principalmente em se tratando de uma LPS (Engstrom et al., 2011). Além do fator teste

em uma LPS, devemos considerar as particularidades individuais de cada produto gerado, e assim temos um ponto que demanda atenção, que é a variabilidade. Além de um sério problema na questão de gerenciamento das mesmas, impactando diretamente nas métricas e formas de medição de qualidade (OliveiraJr et al., 2013).

Como foi relatado na Seção 1 existe uma desafio muito grande na geração de testes para produtos derivados de LPS, tanto pela quantidade de produtos semelhantes, como também por suas particularidades que diferem um dos outros. Sobre esse ponto, consideramos que a variabilidade será gerenciada utilizando *SMarty*, visto que ele já possui um perfil de verificação, assim motivando a criação de um perfil voltado para a validação, completando o ciclo de teste da abordagem *SMarty*.

Outro ponto que motiva a pesquisa por uma abordagem de geração de testes a partir de modelo é a geração automatizada de código, que também faz uso de modelos UML. Um exemplo é o trabalho de Kundu et al., 2013, assim com também a geração de casos de teste a partir de modelos (Panahi e Mohapatra, 2013; Sarma et al., 2007; Sokenou et al., 2006), porém nestes casos não são apresentados evidências de que sejam voltados para LPS e isso pode ser um fator de implicação para o não aproveitamento de tais trabalhos.

Conforme a motivação apresentada, este projeto de dissertação tem como objetivo especificar a abordagem *SMartyTesting* que permita utilizar TBM em nível de sistema, considerando especificamente diagramas de sequência pelo alto grau de representação de variabilidade onde tais artefatos sejam modelados por *SMarty*. Com objetivos específicos tem-se:

- investigar na literatura soluções que envolvam tais artefatos UML em TBM para LPS;
- especificar uma abordagem de transformação de modelos e geração de sequências de testes, e;
- avaliar empiricamente a abordagem *SMartyTesting*.

3.3 Metodologia de Desenvolvimento

Para atingir o objetivo proposto neste trabalho, será necessário realizar as etapas apresentadas na Figura - 3.1.

- **Revisão Bibliográfica:** Estudo dos conceitos sobre os principais temas que farão parte do objetivos da pesquisa, como: Perfis UML, Diagramas de caso de uso e



Figura 3.1: Etapas da Metodologia de Desenvolvimento de Pesquisa

sequência, teste em LPS, TBM, conceitos de LPS, gerenciamento de variabilidade, a abordagem *SMarty* que está no Anexo B.

- **Estudo secundário:** Realização de uma Revisão Sistemática de Literatura (RSL), de estudos primários, onde o tema é TBM para LPS, que encontra-se no Apêndice A. A análise dos estudos da RSL permitiu a identificação de formatos diferenciados de aplicação de TBM em LPS.
- **Proposta da Técnica *SMartyTesting*:** Elaborar e apresentar a proposta da técnica *SMartyTesting* que visa a utilização de TBM para LPS modeladas usando *SMarty* para geração de casos de teste em nível de sistema a partir de um modelo formal.
- **Exemplo de aplicação da Técnica *SMartyTesting*:** Utilização da LPS acadêmica *Arcade Game Maker*(AGM) para aplicar a técnica *SMartyTesting* para a aprendizagem dos conceitos de SPL através de uma abordagem prática. Esta LPS pode ser usada para derivar três jogos eletrônicos diferentes; Bowling, Brickles e Pong, que são usados pela comunidade científica para avaliar e validar suas abordagens **colocar uma referencia de onde ele pode obter a lps AGM** (Costa L. T., 2016).
- **Avaliação Qualitativa da Técnica *SMartyTesting*:** trata dos estudos empíricos que serão conduzidos com a finalidade de validar *SMartyTesting* em relação a geração de casos de teste a partir de modelos *SMarty* utilizando-se de diagrama de caso de uso e sequência. Nas avaliações serão utilizadas as LPS *Mobile Media*(MM) e M-SPLlearning ** ou plets.
- **Redação e Publicação de Resultados:** é escrita e submissão de artigos sobre a técnica proposta e os estudos empíricos conduzidos, bem a escrita e publicação sobre a RSL gerada como estudo secundário, além da escrita e defesa da dissertação.

3.3.1 Ciclo de Desenvolvimento

contextualizar o ciclo de teste pesquisado
posicionar o leitor onde esta nossa abordagem

3.3.2 RoadMap teste TBM em LPS

apresenta os caminhos possíveis encontrados no mapeamento e o qual foi o escolhido e porque

3.3.3 Conversão de Sequência para Atividade

apresentar o motivo de diagrama de sequencia ser o escolhido
porque a conversão para atividade
possibilidade de conversão para maquina de estado
apresentar de forma detalhada o mapeamento

3.3.4 Geração de Sequências de Teste

falar o que é geração de sequencia de teste
falar sobre os metodos de geração de sequencia W, HSI etc
falar sobre a ferramenta plavis
apresentar como é o processo da split mbt
explicar o porque foi escolhido a split

3.3.5 Ferramentas Adaptadas (SPLit-MbT)

porque foi selecionado a split e todos os detalhes de seu funcionamento

3.3.6 Ferramentas e Caminho Escolhido *****

Neste trabalho estamos em análise para algumas ATLs de conversão, uma para a primeira etapa *Convert UML Sequence Diagram to UML State Machine* que está sendo testada, e uma para a segunda, já a segunda deverá ser implementada, devido as particularidades da geração de sequência de teste contendo variabilidade, além da ATL, outras ferramentas estão sendo utilizadas, como o Eclipse, UML *Designer*. A Figura - 3.2 apresenta o caminho que optamos dentro do contexto levantado pelo mapeamento sistemático da literatura apresentado no Anexo A.

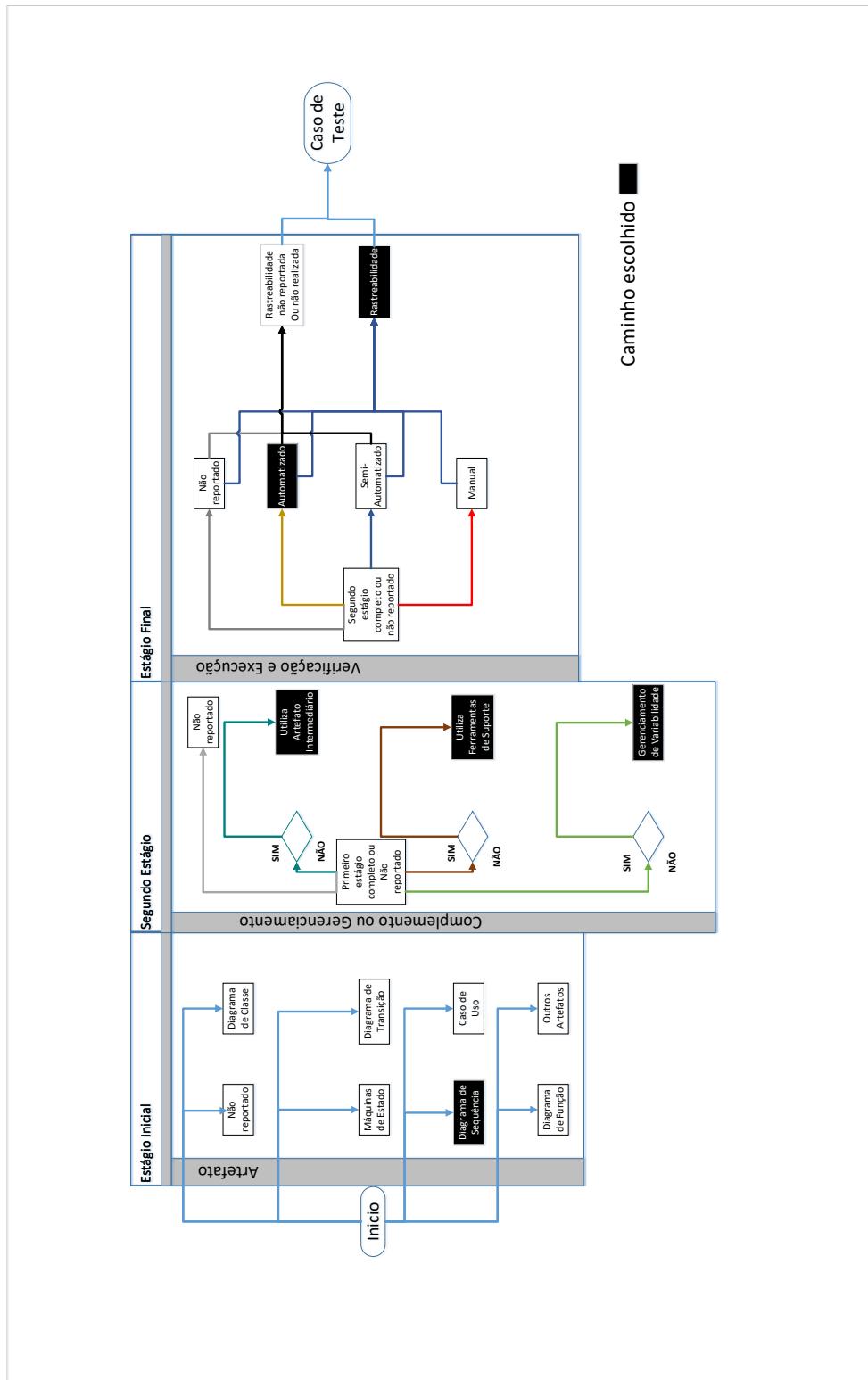


Figura 3.2: Guia de direcionamento de processo de TBM em LPS

3.3.7 Desafios Encontrados

A literatura apresenta muitas soluções para teste de LPS, porém muitas delas não possuem foco na variabilidade quando o assunto é uma LPS. Assim, um dos fatores motivantes é considerar variabilidade na decisão de teste. Outro ponto motivante é a abordagem *SMarty* (OliveiraJr et al., 2010a) que permite a representação de variabilidade em modelos iniciais do ciclo de vida de LPS. Quando se fala em teste, a literatura apresenta várias soluções voltadas a LPS, dentre elas o TBM, embora trabalhe com MEF, seria interessante que pudesse ser trabalhado o teste com artefatos do tipo casos de uso, diagramas de sequência e de atividades, além de diagramas de estado que já é bastante utilizado. Assim, entende-se como uma oportunidade que pesquise o TBM em conjunto *SMarty* para utilização em casos de uso e diagrama de sequência, para se obter casos de teste que possam ser utilizados a nível de sistema.

Costa L. T., 2016; Olimpiew, 2008a apresentam alguns desafios na geração de testes a partir de modelos em uma LPS, um ciclo de criação que perfaz desde o modelo inicial até o caso de teste. Devido as particularidades de cada modelo, se faz necessário adaptações a respeito, um exemplo é onde Costa L. T., 2016 estende a máquina de estado finito, pois ela não possui suporte para variabilidade e para garantir que a variabilidade fosse mantida isso se fez necessário. Olimpiew, 2008a também faz de forma semelhante a em sua abordagem CaDeT, modificando parte dos itens para poder contemplar o objetivo da proposta.

SMarty Testing não seria diferente, houve a necessidade de se encontrar ferramentas que fizessem o apoio automatizado do processo. Sendo assim buscamos procurar a linguagem de transformação de modelos é usada para definir como um conjunto de modelos de origem é mapeado com o objetivo de criar um conjunto de modelos de destino. A linguagem define como as operações básicas sobre os modelos podem ser realizadas usando um conjunto específico de construções de linguagem. A criação de linguagens específicas para transformação de modelos é uma abordagem que foi tomada pela indústria de software e comunidade de pesquisa. Como resultado, diversas linguagens de transformação de modelos têm sido propostas Allilaire et al., 2006.

Limitações de Uso

apresentar as limitações de uso em relação a pesquisa e os suportes que as ferramentas possuem

limitações do mapeamento e da split

Ameaças ao Funcionamento

apresentar as ameaças relacionadas as limitações de uso e interpretação do usuário

Mesmo havendo um embasamento de todo o estudo apresentado aqui a partir da MSL, ainda podemos constatar que podem surgir ameaças ao funcionamento da abordagem quando o assunto é a conversão dos artefatos. Por optarmos trabalhar com diagramas de sequência como ponto de partida, a ATL de conversão trabalha com um metamodelo UML, se o arquiteto de software modelar diagramas que não atendem a quantidade mínima de informações necessárias no diagrama, isso pode mascarar itens, ou criar processos equivocados. Um exemplo seria a representação de um laço de repetição de forma diferente do convencional, que da forma como for representado o retorno, poderá gerar uma sequencia de teste como uma outra operação diferente de um laço de repetição. Sendo assim, devemos apresentar ao utilizador um padrão mínimo para que a seja obtido o resultado esperado.

3.4 Especificação da Abordagem *SMartyTesting*

Nesta Seção é apresentado de forma mais detalhada a abordagem *SMartyTesting*, para um bom entendimento se faz obrigatório a leitura da Seção 3.2.

3.4.1 Etapa 1 - Mapeamento de Diagrama de Sequência para Diagrama de Atividade

explicar de forma detalhada o mapeamento de sequencia para atividade

Como mencionado na Seção 1.4.1 a opção por diagramas de sequência se deve ao fato de ele possibilitar uma maior riqueza de informações a respeito do modelo. A escolha por diagrama de estado na geração de sequência de teste se deve ao fato de ele ser um dos modelos formais mais utilizados em TBM. Além disso ele é uma boa alternativa para projetar componente de teste, pois podem ser aplicáveis em qualquer modelo de especificação descrevendo um número finito de estados, outro ponto é que ele também é o mais adequado para gerar sequências de teste, uma vez que optamos por esse caminho (Costa L. T., 2016).

3.4.2 ATLs de Conversão UML

falar sobre algumas ATLs de conversão e os limitadores da utilização deste recurso

Foram encontradas duas ATLs de conversão a primeira *UML Sequence Diagrams to Statechart Diagram*¹ onde a documentação de utilização é extremamente confusa e de difícil configuração, foram feitos tentes de conversão de diagramas de exemplos que estavam no pacote, mas não foi obtido o resultado esperado. Já a segunda também *open-source*, é chamada de *Convert UML Sequence Diagram to UML State Machine*¹ que possui uma documentação um pouco mais detalhadas em comparação a anterior, foram realizados teste iniciais com os exemplos contidos no pacote onde ouve exito de conversão (Hennicker e Knapp, 2007).

Este pacote utiliza uma linguagem de conversão *Query/View/Transformation* (QVT), QVT é uma linguagem usada para transformar (meta) modelos. Ele usa OCL (Linguagem de Restrição de Objeto) estendida em combinação com uma linguagem específica de domínio: Relações, Core ou Operacional. Este último é uma linguagem imperativa, enquanto os outros dois são declarativos, ela é implementada no Eclipse.

Convert UML Sequence Diagram to UML State Machine possui suporte a arquivos com extensão UML, sendo assim a quantidade de ferramentas que poderiam auxiliar na concepção de diagramas que pudessem ser convertidos ficou um pouco limitada, devido a outras existentes utilizarem formatos próprios de saída, a princípio não realizamos uma busca por conversões de formato para obter uma abrangência maior, ao invés disso procuramos uma ferramenta que fosse robusta e pudesse entregar o proposto. Sendo assim, optamos por realizar testes como o *UML Designer*².

Convert UML Sequence Diagram to UML State Machine

Convert UML Sequence Diagram to UML State Machine se mostra compatível com *SMarty*, pois em sua conversão apresenta um arquivo de extensão *notation* ainda não foram realizados testes com artefatos modelados por *SMarty* mas baseado nas documentações o resultado pode ser satisfatório. Na Figura - 3.3 apresenta uma visão do ciclo da primeira etapa de forma mais resumida, sendo assim na primeira etapa ficou definido a seguinte sequência de ações, onde grande parte automatizada.

Partindo do pressuposto que o artefato seja diagrama de sequência e também modelado no formato *SMarty*, é feito a entrada do artefato em extensão UML, na IDE eclipse através da ATL *Convert UML Sequence Diagram to UML State Machine* é realizado as configurações pré-definidas e após a conversão é gerado dois arquivos, um com extensão *notation* e outro UML.

¹Disponível em:<http://www.st.ewi.tudelft.nl/basgraaf/> - Acessado em 20/03/2019

¹Disponível em: <https://github.com/slashburn/mdd-project> - Acessado em 10/04/2019

²Disponível em: <http://www.umldesigner.org/download/> - Acessado em 10/04/2019

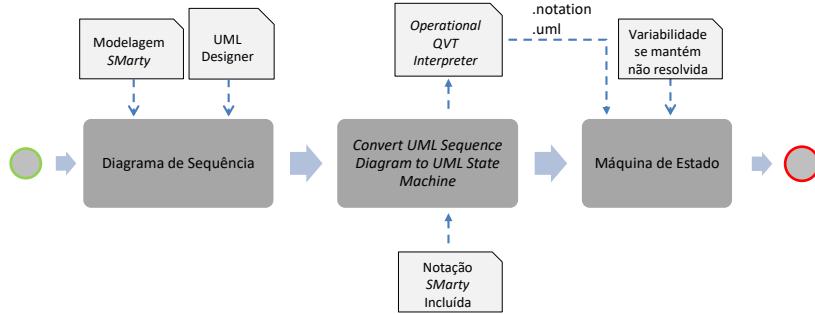


Figura 3.3: Ciclo da etapa 1

3.4.3 Etapa 2 - Geração de Sequência de Teste com apoio da ferramenta SPLiT-MbT

explicar de forma detalhada a geração da sequencia de teste utilizando a split

Após a conversão do artefato para máquina de estado contendo a variabilidade, idealizamos a geração de sequências de teste, onde será possível verificar todas as possíveis combinações e coberturas do artefato. O desafio inicial é como realizar a geração de sequência de teste a partir de máquina de estado, para isso desenvolvemos uma pesquisa a respeito. Realizamos também uma análise para entender se seria vantagem a geração de caso de teste diretamente do artefato de máquina de estado, onde poucos trabalhos foram reportados (Aggarwal e Sabharwal, 2012; Dalepiane, 2014; Samuel et al., 2008), porém, nenhuma deles aborda LPS.

Na busca por modelos de conversão Costa L. T., 2016 relata os principais métodos com suporte a diagrama de máquinas de estado, TT, UIO, DS, W or HSI, porém alguns deles não seriam possíveis de utilização em LPS, no estudo ele relata que o HSI foi o método que contém mais características favoráveis a utilização em LPS. Também um dos métodos menos restritivos em relação às propriedades que as máquinas de estado devem

ter, outro detalhe é que, iremos realizar uma avaliação para comparar os dados obtidos com o trabalho de Costa L. T., 2016.

Um outro exemplo é que o HSI é capaz de interpretar MEF completas e parciais.sendo assim, iremos buscar implementar tal método para geração de sequências de teste através de diagramas de máquinas de estado.



Figura 3.4: Ciclo da etapa 2

3.4.4 Solução da Variabilidade na Engenharia de Domínio

explicar que a variabilidade fica para ser resolvida na engenharia de aplicação

Um dos objetivos deste trabalho é o gerenciamento da variabilidade na geração dos casos de teste. Analisando estudos resultantes da MSL no Anexo A observamos que quando se abordado produtos derivados a variabilidade sempre é deixada para ser resolvida na engenharia de aplicação, principalmente se houver o interesse no reaproveitamento do caso de teste. Pois um dos principais argumento é que se houver o reaproveitamento pode haver leves variações no ponto de variabilidade o que implicaria em um retrabalho na solução da variabilidade já resolvida.

Segundo essa premissa, o caso de teste será gerado contendo a variabilidade, sem a solução. Estará documentado a variabilidade e suas particularidades, mas sem a solução

devida, ficando a cargo do responsável em qual etapa da engenharia de aplicação será realizada a solução.

3.5 Exemplo de Aplicação

ilustrar a utilização com um diagrama

3.6 Considerações Finais

4

Avaliação Empírica de SMartyTesting

4.1 Considerações Iniciais

4.2 Objetivo

4.3 Planejamento

- Estudo Qualitativo - Utilização do modelo proposto em uma LPS real com foco na criação de produtos mobile para ensino de programação, uma vez a LPS constituída, os produtos modelados em engenharia de domínio previamente modelados notados em *SMarty* seria submetidos a *SMartyTesting*, podendo até ser comparada com outras abordagens como SPLiT- MBT.
- Experimento controlado - Avaliar o modelo proposto com base em experimentos controlados, a partir de modelos *SMarty* utilizando-se de diagrama de caso de uso e sequência. Nas avaliações serão utilizadas a LPS a *Product Line of Model-Based Testing Tools* (PLETS) e M-SPLearning..

4.4 Execução

4.5 Análise e Interpretação

4.6 Disseminação

4.7 Ameaças à Validade

4.8 Melhorias Identificadas

4.9 Considerações Finais

5

Melhorias Identificadas e Lições Aprendidas

5.1 Considerações Iniciais

5.2 Melhorias

quais a melhorias que podem ser propostas

5.3 Aprendizado

qual aprendizado com esta pesquisa

5.4 Considerações Finais

6

Conclusão

6.1 Contribuições

contribuições desta pesquisa

6.2 Limitações

limitantes encontrados durante a pesquisa

6.3 Trabalhos Futuros

o que pode ser feito e qual o caminho

REFERÊNCIAS

- AGGARWAL, M.; SABHARWAL, S. Test case generation from uml state machine diagram: A survey. In: *2012 Third International Conference on Computer and Communication Technology*, IEEE, 2012, p. 133–140.
- AL-HAJJAJI, M.; KRIETER, S.; THÜM, T.; LOCHAU, M.; SAAKE, G. Incling: efficient product-line testing using incremental pairwise sampling. In: *ACM SIGPLAN Notices*, ACM, 2016, p. 144–155.
- ALI, S.; YUE, T.; BRIAND, L.; WALAWEGE, S. A product line modeling and configuration methodology to support model-based testing: an industrial case study. In: *International Conference on Model Driven Engineering Languages and Systems*, Springer, 2012, p. 726–742.
- ALLILAIRE, F.; BÉZIVIN, J.; JOUAUT, F.; KURTEV, I. Atl-eclipse support for model transformation. In: *Proceedings of the Eclipse Technology eXchange workshop (eTX) at the ECOOP 2006 Conference, Nantes, France*, Citeseer, 2006.
- ALVES, V.; SCHNEIDER, D.; BECKER, M.; IESE, F.; PLATZ, F.; BENCOMO, N.; GRACE, P. Comparitive study of variability management in software product lines and runtime adaptable systems. In: *International Workshop on Variability Modelling of Software-Intensive Systems*, 2009, p. 9–17.
- APEL, S.; BATORY, D.; KÄSTNER, C.; SAAKE, G. *Feature-oriented software product lines*. Springer, 2013.
- BEOHAR, H.; MOUSAVI, M. R. Input-output conformance testing based on featured transition systems. In: *Proceedings of the 29th Annual ACM Symposium on Applied Computing*, ACM, 2014a, p. 1272–1278.
- BEOHAR, H.; MOUSAVI, M. R. Spinal test suites for software product lines. *arXiv preprint arXiv:1403.7260*, 2014b.

- BEOHAR, H.; VARSHOSAZ, M.; MOUSAVI, M. R. Basic behavioral models for software product lines: Expressiveness and testing pre-orders. *Science of Computer Programming*, v. 123, p. 42–60, 2016.
- BERA, M. H. G. *Smartycomponents: um processo para especificação de arquiteturas de linha de produto de software baseadas em uml*. Dissertação de Mestrado, Universidade Estadual de Maringá, Departamento de Informática, Programa de Pós Graduação em Ciência da Computação, 2015.
- BERA, M. H. G.; OLIVEIRAJR, E.; COLANZI, T. E. Evidence-based smarty support for variability identification and representation in component models. In: *Proceedings of the 17th International Conference on Enterprise Information Systems - Volume 2: ICEIS*, 2015, p. 295–302.
- BERNARDINO, M.; RODRIGUES, E. M.; ZORZO, A. F.; MARCHEZAN, L. Systematic mapping study on mbt: tools and models. *IET Software*, v. 11, n. 4, p. 141–155, 2017.
- BRINGMANN, E.; KRÄMER, A. Model-based testing of automotive systems. In: *2008 1st international conference on software testing, verification, and validation*, IEEE, 2008, p. 485–493.
- CAI, X.; ZENG, H. Model-based test generation for software product line. In: *Computer and Information Science (ICIS), 2013 IEEE/ACIS 12th International Conference on*, IEEE, 2013, p. 347–351.
- DO CARMO MACHADO, I.; MCGREGOR, J. D.; CAVALCANTI, Y. C.; DE ALMEIDA, E. S. On strategies for testing software product lines: A systematic literature review. *Information and Software Technology*, v. 56, n. 10, p. 1183–1199, 2014.
- CHEN, L.; ALI BABAR, M.; ALI, N. Variability management in software product lines: a systematic review. In: *Proceedings of the 13th International Software Product Line Conference*, Carnegie Mellon University, 2009a, p. 81–90.
- CHEN, L.; ALI BABAR, M.; ALI, N. Variability management in software product lines: A systematic review. In: *Proceedings of the 13th International Software Product Line Conference*, SPLC '09, Pittsburgh, PA, USA: Carnegie Mellon University, 2009b, p. 81–90 (SPLC '09,).
- Disponível em <http://dl.acm.org/citation.cfm?id=1753235.1753247>

- CHEN, L.; BABAR, M. A. A systematic review of evaluation of variability management approaches in software product lines. *Information and Software Technology*, v. 53, n. 4, p. 344–362, 2011.
- CICHOS, H.; OSTER, S.; LOCHAU, M.; SCHÜRR, A. Model-based coverage-driven test suite generation for software product lines. In: *International Conference on Model Driven Engineering Languages and Systems*, Springer, 2011, p. 425–439.
- CLEMENTS, P.; NORTHRUP, L. *Software product lines*. Addison-Wesley, 2002.
- COSTA L. T., L. T. *Split-mbt: A model-based testing method for software product lines*. Dissertação de Mestrado, Pontifícia Universidade Católica do Rio Grande do Sul, 2016.
- CRESPO, A. N.; SILVA, O. J.; BORGES, C. A.; SALVIANO, C. F.; ARGOLLO, M.; JINO, M. Uma metodologia para teste de software no contexto da melhoria de processo. *Simpósio Brasileiro de Qualidade de Software*, p. 271–285, 2004.
- DALEPIANE, M. F. Implementation of a model based test case generator for uml state machines. 2014.
- DAMIANI, F.; FAITELSON, D.; GLADISCH, C.; TYSZBEROWICZ, S. A novel model-based testing approach for software product lines. *Software & Systems Modeling*, v. 16, n. 4, p. 1223–1251, 2017.
- DAMIANI, F.; GLADISCH, C.; TYSZBEROWICZ, S. Refinement-based testing of delta-oriented product lines. In: *Proceedings of the 2013 International Conference on Principles and Practices of Programming on the Java Platform: Virtual Machines, Languages, and Tools*, ACM, 2013, p. 135–140.
- DELAMARO, M.; JINO, M.; MALDONADO, J. *Introdução ao teste de software*. Elsevier Brasil, 2017.
- DEVROEY, X. Behavioural model based testing of software product lines. 2014.
- DEVROEY, X. *Behavioural model-based testing of software product lines*. Tese de Doutoramento, UniversitÃ de Namur, 2017.
- DEVROEY, X.; CORDY, M.; PERROUIN, G.; KANG, E.-Y.; SCHOBBIENS, P.-Y.; HEYMANS, P.; LEGAY, A.; BAUDRY, B. A vision for behavioural model-driven validation of software product lines. In: *International Symposium On Leveraging Applications of Formal Methods, Verification and Validation*, Springer, 2012, p. 208–222.

- DEVROEY, X.; PERROUIN, G.; LEGAY, A.; CORDY, M.; SCHOBBENS, P.-Y.; HEYMANS, P. Coverage criteria for behavioural testing of software product lines. In: *International Symposium On Leveraging Applications of Formal Methods, Verification and Validation*, Springer, 2014a, p. 336–350.
- DEVROEY, X.; PERROUIN, G.; SCHOBBENS, P.-Y. Abstract test case generation for behavioural testing of software product lines. In: *Proceedings of the 18th International Software Product Line Conference: Companion Volume for Workshops, Demonstrations and Tools-Volume 2*, ACM, 2014b, p. 86–93.
- DEVROEY, X.; PERROUIN, G.; SCHOBBENS, P.-Y.; HEYMANS, P. Vibes, transition system mutation made easy. In: *Proceedings of the 37th International Conference on Software Engineering-Volume 2*, IEEE Press, 2015, p. 817–818.
- DUKACZEWSKI, M.; SCHAEFER, I.; LACHMANN, R.; LOCHAU, M. Requirements-based delta-oriented spl testing. In: *Product Line Approaches in Software Engineering (PLEASE), 2013 4th International Workshop on*, IEEE, 2013, p. 49–52.
- ENGSTRÖM, E.; RUNESON, P. Software product line testing—a systematic mapping study. *Information and Software Technology*, v. 53, n. 1, p. 2–13, 2011.
- ENGSTROM, E.; ET AL. Testing software product lines. *IEEE software*, v. 28, n. 5, p. 16–20, 2011.
- ESCALONA, M.; GUTIERREZ, J. J.; MEJÍAS, M.; ARAGÓN, G.; RAMOS, I.; TORRES, J.; DOMÍNGUEZ, F. An overview on test generation from functional requirements. *Journal of Systems and Software*, v. 84, n. 8, p. 1379–1393, 2011.
- FARRAG, M. *Colored model based testing for software product lines (cmbt-swpl)*. Tese de Doutoramento, Technical University of Ilmenau, 2010.
- FELDT, R.; ZIMMERMANN, T.; BERGERSEN, G. R.; FALESSI, D.; JEDLITSCHKA, A.; JURISTO, N.; MÜNCH, J.; OIVO, M.; RUNESON, P.; SHEPPERD, M.; SJØBERG, D. I. K.; TURHAN, B. Four commentaries on the use of students and professionals in empirical software engineering experiments. *Empirical Software Engineering*, v. 23, n. 6, p. 3801–3820, 2018.

Disponível em <https://doi.org/10.1007/s10664-018-9655-0>

- FENG, Y.; LIU, X.; KERRIDGE, J. A product line based aspect-oriented generative unit testing approach to building quality components. In: *Annual International Computer Software and Applications Conference*, 2007, p. 403–408.
- FRAGAL, V. H.; SIMAO, A.; ENDO, A. T.; MOUSAVI, M. R. Reducing the concretization effort in fsm-based testing of software product lines. In: *2017 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*, IEEE, 2017, p. 329–336.
- GARCÍA GUTIÉRREZ, B.; GARCIA CARMONA, R.; NAVAS BALTASAR, A.; PARADA GÉLVEZ, H. A.; CUADRADO LATASA, F.; DUEÑAS LÓPEZ, J. C. An automated model-based testing approach in software product lines using a variability language. In: *Workshop on Model-Driven Tool and Process Integration*, 2010, p. 1–10.
- GBIZLI, C. S.; SÖZER, H. Model-based software product line testing by coupling feature models with hierarchical markov chain usage models. In: *Software Quality, Reliability and Security Companion (QRS-C), 2016 IEEE International Conference on*, IEEE, 2016, p. 278–283.
- GBIZLI, C. S.; SÖZER, H.; ERCAN, A. Ö. Successive refinement of models for model-based testing to increase system test effectiveness. In: *Software Testing, Verification and Validation Workshops (ICSTW), 2016 IEEE Ninth International Conference on*, IEEE, 2016, p. 263–268.
- GERALDI, R. T.; OLIVEIRAJR, E.; CONTE, T.; STEINMACHER, I. Checklist-based inspection of smarty variability models - proposal and empirical feasibility study. In: *Proceedings of the 17th International Conference on Enterprise Information Systems - Volume 2: ICEIS*,, 2015, p. 268–276.
- HASLING, B.; GOETZ, H.; BEETZ, K. Model based testing of system requirements using uml use case models. In: *Software Testing, Verification, and Validation, 2008 1st international conference on*, IEEE, 2008, p. 367–376.
- HENARD, C.; PAPADAKIS, M.; PERROUIN, G.; KLEIN, J.; LE TRAON, Y. Assessing software product line testing via model-based mutation: An application to similarity testing. In: *Software Testing, Verification and Validation Workshops (ICSTW), 2013 IEEE Sixth International Conference on*, IEEE, 2013, p. 188–197.

- HENNICKER, R.; KNAPP, A. Activity-driven synthesis of state machines. In: *International Conference on Fundamental Approaches to Software Engineering*, Springer, 2007, p. 87–101.
- INSTITUTE, S. S. E. A framework for software product line practice, version 5.0. *SEI.-2007-<http://www.sei.cmu.edu/productlines/index.html>*, 2012.
- ISA, M. A. B.; RAZAK, S. B. A.; JAWAWI, D. N. B. A.; FUH, O. L. Model-based testing for software product line: A systematic literature review. *International Journal of Software Engineering and Technology*, v. 2, n. 2, 2017.
- KAKARONTZAS, G.; STAMELOS, I.; KATSAROS, P. Product line variability with elastic components and test-driven development. In: *International Conference on Computational Intelligence for Modelling Control Automation*, 2008, p. 146–151.
- KEELE, S.; ET AL. Guidelines for performing systematic literature reviews in software engineering. In: *Technical report, Ver. 2.3 EBSE Technical Report. EBSE*, sn, 2007.
- KITCHENHAM, B. Procedures for performing systematic reviews. *Keele, UK, Keele University*, v. 33, n. 2004, p. 1–26, 2004.
- KITCHENHAM, B. A.; BUDGEN, D.; BRERETON, P. *Evidence-based software engineering and systematic reviews*, v. 4. CRC press, 2015.
- KNAPP, A.; ROGGENBACH, M.; SCHLINGLOFF, B.-H. On the use of test cases in model-based software product line development. In: *Proceedings of the 18th International Software Product Line Conference- Volume 1*, ACM, 2014, p. 247–251.
- KUNDU, D.; SAMANTA, D.; MALL, R. Automatic code generation from unified modelling language sequence diagrams. *IET Software*, v. 7, n. 1, p. 12–28, 2013.
- LACHMANN, R.; BEDDIG, S.; LITY, S.; SCHULZE, S.; SCHAEFER, I. Risk-based integration testing of software product lines. In: *Proceedings of the Eleventh International Workshop on Variability Modelling of Software-intensive Systems*, ACM, 2017, p. 52–59.
- LACKNER, H.; THOMAS, M.; WARTENBERG, F.; WEISSLEDER, S. Model-based test design of product lines: Raising test design to the product line level. In: *Software Testing, Verification and Validation (ICST), 2014 IEEE Seventh International Conference on*, IEEE, 2014, p. 51–60.

- LAMANCHA, B. P.; MATEO, P. R.; DE GUZMÁN, I. R.; USAOLA, M. P.; VELTHIUS, M. P. Automated model-based testing using the uml testing profile and qvt. In: *Proceedings of the 6th International Workshop on Model-Driven Engineering, Verification and Validation*, ACM, 2009a, p. 6.
- LAMANCHA, B. P.; MATEO, P. R.; DE GUZMÁN, I. R.; USAOLA, M. P.; VELTHIUS, M. P. Automated model-based testing using the uml testing profile and qvt. In: *International Workshop on Model-Driven Engineering, Verification and Validation*, ACM, 2009b, p. 6.
- LAMANCHA, B. P.; USAOLA, M. P.; VELTHIUS, M. P. A model based testing approach for model-driven development and software product lines. In: *International Conference on Evaluation of Novel Approaches to Software Engineering*, Springer, 2010a, p. 193–208.
- LAMANCHA, B. P.; USAOLA, M. P.; VELTHIUS, M. P. A model based testing approach for model-driven development and software product lines. In: *International Conference on Evaluation of Novel Approaches to Software Engineering*, Springer, 2010b, p. 193–208.
- LEE, J.; KANG, S.; LEE, D. A survey on software product line testing. In: *International Software Product Line Conference, SPLC '12*, New York, NY, USA: ACM, 2012, p. 31–40 (*SPLC '12*).
- LI, Q. A novel likert scale based on fuzzy sets theory. *Expert Systems with Applications*, v. 40, n. 5, p. 1609–1618, 2013.
- LINDEN, F.; SCHMID, K.; ROMMES, E. The product line engineering approach. *Software Product Lines in Action*, p. 3–20, 2007.
- LITY, S.; LOCHAU, M.; SCHAEFER, I.; GOLTZ, U. Delta-oriented model-based spl regression testing. In: *International Workshop on Product Line Approaches in Software Engineering*, IEEE Press, 2012, p. 53–56.
- LITY, S.; MORBACH, T.; THÜM, T.; SCHAEFER, I. Applying incremental model slicing to product-line regression testing. In: *International Conference on Software Reuse*, Springer, 2016, p. 3–19.
- LOCHAU, M.; KAMISCHKE, J. Parameterized preorder relations for model-based testing of software product lines. In: *International Symposium On Leveraging Applications of Formal Methods, Verification and Validation*, Springer, 2012, p. 223–237.

LOCHAU, M.; OSTER, S.; GOLTZ, U.; SCHÜRR, A. Model-based pairwise testing for feature interaction coverage in software product line engineering. *Software Quality Journal*, v. 20, n. 3-4, p. 567–604, 2012a.

LOCHAU, M.; PELDSZUS, S.; KOWAL, M.; SCHAEFER, I. Model-based testing. In: *Advanced Lectures of the 14th International School on Formal Methods for Executable Software Models - Volume 8483*, New York, NY, USA: Springer-Verlag New York, Inc., 2014, p. 310–342.

Disponível em http://dx.doi.org/10.1007/978-3-319-07317-0_8

LOCHAU, M.; SCHAEFER, I.; KAMISCHKE, J.; LITY, S. Incremental model-based testing of delta-oriented software product lines. In: *International Conference on Tests and Proofs*, Springer, 2012b, p. 67–82.

MARCOLINO, A.; OLIVEIRA, E.; GIMENES, I.; BARBOSA, E. F. Empirically based evolution of a variability management approach at uml class level. In: *2014 IEEE 38th Annual Computer Software and Applications Conference*, 2014a, p. 354–363.

MARCOLINO, A.; OLIVEIRAJR, E.; GIMENES, I. Towards the effectiveness of the smarty approach for variability management at sequence diagram level. In: *Proceedings of the 16th International Conference on Enterprise Information Systems - Volume 2: ICEIS*, 2014b, p. 249–256.

MARCOLINO, A.; OLIVEIRAJR, E.; GIMENES, I. M. S.; MALDONADO, J. C. Towards the effectiveness of a variability management approach at use case level. In: *The 25th International Conference on Software Engineering and Knowledge Engineering, Boston, MA, USA, June 27-29, 2013.*, 2013, p. 214–219.

DE MORAIS, S. R.; AUSSEM, A. A novel markov boundary based feature subset selection algorithm. *Neurocomputing*, v. 73, n. 4-6, p. 578–584, 2010.

OLIMPIEW, E. M. *Model-based testing for software product lines*. George Mason University, 2008a.

OLIMPIEW, E. M. *Model-based testing for software product lines*. Tese de Doutorado, George Mason University, Fairfax, VA, USA, aAI3310145, 2008b.

OLIMPIEW, E. M.; GOMAA, H. Model-based testing for applications derived from software product lines. In: *ACM SIGSOFT Software Engineering Notes*, ACM, 2005, p. 1–7.

- OLIVEIRAJR; GIMENES, I. M.; MALDONADO, J. C. Systematic management of variability in uml-based software product lines. *Journal of Universal Computer Science*, v. 16, n. 17, p. 2374–2393, 2010a.
- OLIVEIRAJR, E.; GIMENES, I. M.; MALDONADO, J. C.; MASIERO, P. C.; BARROCA, L. Systematic evaluation of software product line architectures. *Journal of Universal Computer Science*, v. 19, n. 1, p. 25–52, 2013.
- OLIVEIRAJR, E.; GIMENES, I. M. S.; MALDONADO, J. C. Systematic management of variability in uml-based software product lines. *Journal of Universal Computer Science (JUCS)*, v. 16, n. 17, p. 2374–2393, 2010b.
- OSTER, S. *Feature model-based software product line testing*. Tese de Doutoramento, Technische Universität, 2012.
- OSTER, S.; ZINK, M.; LOCHAU, M.; GRECHANIK, M. Pairwise feature-interaction testing for spls: potentials and limitations. In: *Proceedings of the 15th International Software Product Line Conference, Volume 2*, ACM, 2011a, p. 6.
- OSTER, S.; ZORCIC, I.; MARKERT, F.; LOCHAU, M. Moso-polite: tool support for pairwise and model-based software product line testing. In: *Proceedings of the 5th Workshop on Variability Modeling of Software-Intensive Systems*, ACM, 2011b, p. 79–82.
- PANTHI, V.; MOHAPATRA, D. P. Automatic test case generation using sequence diagram. In: *Proceedings of International Conference on Advances in Computing*, Springer, 2013, p. 277–284.
- PATEL, S.; SHAH, V. Automated testing of software-as-a-service configurations using a variability language. In: *Proceedings of the 19th International Conference on Software Product Line*, ACM, 2015, p. 253–262.
- PÉREZ, A. M.; KAISER, S. Bottom-up reuse for multi-level testing. *Journal of Systems and Software*, v. 83, n. 12, p. 2392–2415, 2010.
- PERROUIN, G.; SEN, S.; KLEIN, J.; BAUDRY, B.; LE TRAON, Y. Automated and scalable t-wise test case generation strategies for software product lines. In: *Software Testing, Verification and Validation (ICST), 2010 Third International Conference on*, IEEE, 2010, p. 459–468.

- PETERSEN, K.; VAKKALANKA, S.; KUZNIARZ, L. Guidelines for conducting systematic mapping studies in software engineering: An update. *Information and Software Technology*, v. 64, p. 1–18, 2015.
- POHL, K.; BÖCKLE, G.; VAN DER LINDEN, F. J. *Software product line engineering: foundations, principles and techniques*. Springer Science & Business Media, 2005.
- REALES, P.; POLO, M.; CAIVANO, D. Model based testing in software product lines. In: *International Conference on Enterprise Information Systems*, Springer, 2011a, p. 270–283.
- REALES, P.; POLO, M.; CAIVANO, D. Model based testing in software product lines. In: *International Conference on Enterprise Information Systems*, Springer, 2011b, p. 270–283.
- REIS, S.; METZGER, A.; POHL, K. Integration testing in software product line engineering: A model-based technique. In: DWYER, M. B.; LOPES, A., eds. *Fundamental Approaches to Software Engineering*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, p. 321–335.
- REUYS, A.; KAMSTIES, E.; POHL, K.; REIS, S. Model-based system testing of software product families. In: *International Conference on Advanced Information Systems Engineering*, Springer, 2005, p. 519–534.
- RODRIGUES, E. d. M. Plets: a product line of model-based testing tools. 2013.
- RODRIGUES, E. d. M.; ZORZO, A. F.; NAKAGAWA, E.; GIMENEZ, I.; MALDONADO, J.; DE OLIVEIRA, F. A software product line for model-based testing tools. *Journal of Systems and Software*, p. 1–26, 2012.
- SAMIH, H.; BAUDRY, B. Relating variability modeling and model-based testing for software product lines testing. *Proceedings of the ICTSS*, p. 18–22, 2012.
- SAMIH, H.; BOGUSCH, R. Mplm-matelo product line manager:[relating variability modelling and model-based testing]. In: *Proceedings of the 18th International Software Product Line Conference: Companion Volume for Workshops, Demonstrations and Tools-Volume 2*, ACM, 2014a, p. 138–142.
- SAMIH, H.; BOGUSCH, R. Mplm-matelo product line manager:[relating variability modelling and model-based testing]. In: *Proceedings of the 18th International Software*

Product Line Conference: Companion Volume for Workshops, Demonstrations and Tools-Volume 2, ACM, 2014b, p. 138–142.

SAMIH, H.; LE GUEN, H.; BOGUSCH, R.; ACHER, M.; BAUDRY, B. An approach to derive usage models variants for model-based testing. In: MERAYO, M. G.; DE OCA, E. M., eds. *Testing Software and Systems*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2014a, p. 80–96.

SAMIH, H.; LE GUEN, H.; BOGUSCH, R.; ACHER, M.; BAUDRY, B. Deriving usage model variants for model-based testing: an industrial case study. In: *Engineering of Complex Computer Systems (ICECCS), 2014 19th International Conference on*, IEEE, 2014b, p. 77–80.

SAMUEL, P.; MALL, R.; BOTHRA, A. K. Automatic test case generation using unified modeling language (uml) state diagrams. *IET software*, v. 2, n. 2, p. 79–93, 2008.

SARMA, M.; KUNDU, D.; MALL, R. Automatic test case generation from uml sequence diagram. In: *15th International Conference on Advanced Computing and Communications (ADCOM 2007)*, IEEE, 2007, p. 60–67.

SOKENOU, D.; ET AL. Generating test sequences from uml sequence diagrams and state diagrams. In: *GI Jahrestagung (2)*, Citeseer, 2006, p. 236–240.

STEFFENS, M.; OSTER, S.; LOCHAU, M.; FOGDAL, T. Industrial evaluation of pairwise spl testing with moso-polite. In: *Proceedings of the Sixth International Workshop on Variability Modeling of Software-Intensive Systems*, ACM, 2012a, p. 55–62.

STEFFENS, M.; OSTER, S.; LOCHAU, M.; FOGDAL, T. Industrial evaluation of pairwise spl testing with moso-polite. In: *Proceedings of the Sixth International Workshop on Variability Modeling of Software-Intensive Systems*, ACM, 2012b, p. 55–62.

VARSHOSAZ, M.; BEOHAR, H.; MOUSAVI, M. R. Delta-oriented fsm-based testing. In: *International Conference on Formal Engineering Methods*, Springer, 2015, p. 366–381.

WANG, S.; ALI, S.; GOTLIEB, A. Automated product line methodologies to support model-based testing. In: *Demos/Posters/StudentResearch@ MoDELS*, 2013a, p. 56–60.

WANG, S.; ALI, S.; YUE, T.; LIAAEN, M. Using feature model to support model-based testing of product lines: an industrial case study. In: *Quality Software (QSIC), 2013 13th International Conference on*, IEEE, 2013b, p. 75–84.

- WEISSLEDER, S.; LACKNER, H. Top-down and bottom-up approach for model-based testing of product lines. *arXiv preprint arXiv:1303.1011*, 2013a.
- WEISSLEDER, S.; LACKNER, H. Top-down and bottom-up approach for model-based testing of product lines. *arXiv preprint arXiv:1303.1011*, 2013b.
- WEISSLEDER, S.; SCHLINGLOFF, H. An evaluation of model-based testing in embedded applications. In: *Software Testing, Verification and Validation (ICST), 2014 IEEE Seventh International Conference on*, IEEE, 2014, p. 223–232.
- WEISSLEDER, S.; SOKENOU, D.; SCHLINGLOFF, B.-H. Reusing state machines for automatic test generation in product lines. In: *1st workshop on model-based testing in practice (MoTiP)*, 2008, p. 19.
- ZHANG, M.; ALI, S.; YUE, T.; NORGRE, R. Uncertainty-wise evolution of test ready models. *Information and Software Technology*, v. 87, p. 140–159, 2017.



Teste Baseado em Modelos para LPS:um Mapeamento Sistemático da Literatura

A.1 Planejamento da MSL

A.1.1 Objetivo da Pesquisa

O objetivo deste mapeamento sistemático da literatura é identificar estudos sobre Teste Baseado em Modelo (TBM) para Linha de Produto de Software (LPS) com foco em variabilidade, domínio de aplicação, ferramenta, modelos de LPS, entre outros.

A MSL apresentada aqui segue padrões definidos como principais orientações para a execução de um mapeamento sistemático da literatura por (Kitchenham, 2004) e também por (Keele et al., 2007).

A.2 Metodologia de Pesquisa

Elaboramos a metodologia de pesquisa seguindo as diretrizes de Petersen et al., 2015 e Kitchenham et al., 2015.

A Figura - 1.1 mostra a visão geral do nosso processo de mapeamento sistemático.

Começamos nosso estudo definindo questões de pesquisa (Seção A.2.1). Em seguida, realizamos o processo de pesquisa (Seção A.2.2) para reunir um conjunto inicial de estudos

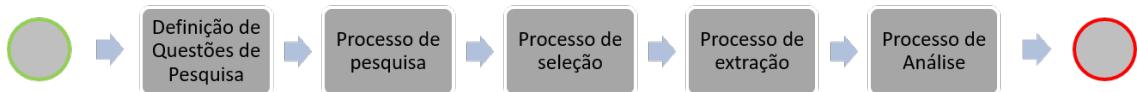


Figura 1.1: Visão geral do nosso processo de mapeamento sistemático

primários de várias fontes diferentes. Com este conjunto de estudos, realizamos o processo de seleção (Seção A.2.3), filtrando estudos de acordo com diferentes critérios. O processo de extração (Seção A.2.4) foi realizado nos estudos filtrados para suportar o processo de análise. No processo de análise (Seção A.2.5) respondemos as questões de pesquisa definidas para este estudo.

A.2.1 Objetivo e Questões de Pesquisa

Este SMS **tem como objetivo** coletar evidências da literatura, **com o propósito de** caracterizar Testes Baseados em Modelo, **com relação a** sua aplicação em Linhas de Produto de Software, **do ponto de vista** de Pesquisadores de LPS, **no contexto de** diferentes fontes digitais de estudo primário.

Nós formulamos seis questões de pesquisa com base no objetivo principal deste SMS, como segue:

- **RQ.1:Para quais domínios do aplicativo LPS, tipos de solução e propostas o TBM é usado?** Estamos interessados em reunir evidências de quais domínios de LPS são mais frequentes, como, por exemplo, software, aeroespacial e automotivo, bem como que tipo de soluções foram propostas para TBM de LPSS;
- **RQ.2:Quais abordagens de TBM, níveis de teste e artefatos foram usados para testar LPSS?** Nesta questão, o foco principal é identificar técnicas de TBM, níveis de teste e automação de teste;

- **RQ.3:Como a variabilidade e o tempo de ligação são tratados durante o TBM das LPSs?** Nesta questão, gostaríamos de compreender quais estudos primários levam em conta a variabilidade e como eles realizam o gerenciamento da variabilidade nas atividades de TBM da LPS, bem como se o tempo de ligação afeta as atividades do TBM nas LPSs;
- **RQ.4:O TBM suporta testes de requisitos não-funcionais de LPS?** Estamos interessados em coletar evidências se os estudos primários dependem do teste de requisitos não-funcionais de LPS;
- **RQ.5:Como as propostas de TBM de LPS são avaliadas?** As soluções de TBMs de LPSs devem ser avaliadas como um meio de fornecer evidência de sua viabilidade. Portanto, gostaríamos de entender que tipo de avaliação é realizada, como experimentos controlados e estudos de caso;
- **RQ.6:Como a rastreabilidade é considerada durante atividades de TBM para LPSs?** Rastreabilidade é um conceito fundamental de TBM e LPS. Nesta questão, reunimos evidências de como as soluções consideram tal conceito para atividades de TBM em LPS;

A.2.2 Processo de pesquisa

A Figura - 1.2 mostra o processo de busca do SMS.

A estratégia de busca para encontrar estudos primários relevantes definidos para este SMS é baseada principalmente na seleção de termos e sinônimos relacionados ao TBM aplicado ao LPS. Esses termos e sinônimos levam em consideração algumas das palavras-chave de trabalhos relacionados e são apresentadas da seguinte forma:

- ***software***;
- ***product line***:*product-line*, *product family*, *product-family*, *product-families*, *family of products*;
- ***model-based testing***:*model based testing*, *TBM*.

Uma vez que tivemos esses termos e seus sinônimos, associamos esses sinônimos ao operador lógico “ OR ” e termos com “ AND ”. Assim, criamos nossa *string* de pesquisa geral, conforme apresentado na Tabela - 1.1.

Além disso, selecionamos fontes de pesquisa (eletrônica e manual), idioma dos estudos primários e tipo de publicação da seguinte forma:

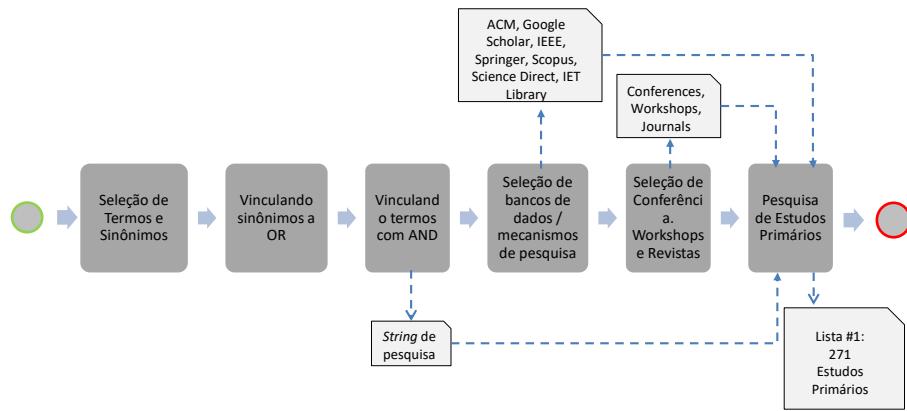


Figura 1.2: Processo de pesquisa por SMS

Tabela 1.1: String de pesquisa geral do SMS

software **AND** (“product line” **OR** “product-line” **OR**
“product family” **OR** “product-family” **OR** “product-families” **OR**
“family of products” **OR** variability) **AND** (“model-based testing” **OR**
“model based testing” **OR** TBM)

- **Fontes de Pesquisa:** bases de dados eletrônicas indexadas (IEEE, ACM, Science-Direct, Scopus, Biblioteca TET IET, Google Scholar, Springer) listadas na Tabela - 1.2 e pesquisa manual em revistas especializadas, conferências e workshops como na Tabela - 1.3 e Tabela - 1.4;
- **Linguagem de Estudos:** Inglês, devido à sua abrangência na área de Ciência da Computação;
- **Tipo de Publicação:** estudos primários revisados por pares publicados em periódicos, conferências / workshops ou capítulos de livros.

As razões para escolher essas fontes de pesquisa estão listadas:

- fonte de ciência da computação mundialmente conhecida;
- fornece um mecanismo de pesquisa avançado capaz de lidar com consultas avançadas;

- indexado internacionalmente;
- índices de origem conferências e periódicos mais relevantes em Engenharia de Software;
- índices de fontes de artigos com fator de alto impacto (JCR e / ou índice H);
- Fonte não cessou a publicação.

Tabela 1.2: Fontes de pesquisa eletrônicas definidas

Fonte Eletrônica	URL
ACM Digital Library	http://dl.acm.org
IEEE Xplore	http://ieeexplore.ieee.org
ScienceDirect	http://www.sciencedirect.com
Scopus	http://www.info.sciverse.com/scopus
Springer	http://www.springer.com
Google Scholar	https://scholar.google.com
IET Digital Library	http://digital-library.theiet.org/content/journals/iet-sen

Tabela 1.3: Conferências Definidas e Workshops para Pesquisa Manual

Acrônimo	Conferência/Workshop
ICECCS	Engineering of Complex Computer Systems
AISE	Advanced Information Systems Engineering
ICST	International Conference on Software Testing
ICIS	Computer and Information Science
ACM SIGSOFT	Software Engineering Nâotés
MSIS	Workshop on Variability Modeling of Software-Intensive Systems
ILPS	International Software Product Line Conference
ICSTSS	International Conference on Testing Software and Systems
QA&TEST	International Conference on Software QA and Testing
IFIP	International Conference on Testing Software and Systems

Tabela 1.4: Diários Definidos para Pesquisa Manual

Acrônimo	Journal
STVR	Software Testing, Verification & Reliability
ESE	Empirical Software Engineering
JSS	Journal of Systems and Software
IST	Information and Software Technology
ASC	Applied Soft Computing
SQJ	Software Quality Journal
SCP	Science of Computer Programming

Como última atividade do processo de busca, realizamos a busca por estudos primários aplicando a string de busca a fontes eletrônicas, e buscando estudos primários manualmente em congressos, workshops e periódicos, conforme apresentado nas próximas seções. Um total de 271 estudos (Lista # 1) foram recuperados no final deste processo.

Avaliação de protocolo

Elaboramos um questionário¹ para avaliar nosso protocolo SMS para coletar a opinião de pesquisadores que trabalharam com TBM e / ou LPS, para facilitar a construção da pesquisa. O questionário é composto por oito itens, sete questões em escala de Likert como (Li, 2013) e uma questão aberta, como segue:

1. Este estudo baseia-se em uma importante questão de pesquisa em *Model-Based Testing* (TBM) de linhas de produto de software (LPS).
2. A cadeia de pesquisa é adequadamente derivada das questões de pesquisa.
3. As fontes de pesquisa definidas são suficientes para cobrir estudos relevantes.
4. Os critérios de inclusão e exclusão são adequados e adequadamente descritos.
5. A pesquisa pode avaliar a qualidade / validade dos estudos selecionados.
6. O processo de extração aborda adequadamente as questões de pesquisa.
7. O processo de análise é apropriado para responder às questões de pesquisa.

Cada pergunta do tipo likert tem as seguintes opções como respostas:

- **Eu discordo totalmente:** quando o protocolo não atende aos critérios da pergunta de forma alguma;
- **Discordo Parcialmente** quando o protocolo não atende a alguns critérios da questão;
- **Neutro:** quando o protocolo não deixa claro se ele atende ou não aos critérios da questão;
- **Eu Concordo Parcialmente** quando o protocolo atende a alguns critérios da questão; e
- **Eu concordo plenamente:** quando o protocolo atende totalmente aos critérios da pergunta.

¹<https://drive.google.com/open?id=1U0zfLS6JLf06krrFkbpwL0mf1CtrNkVBJ5cGKDjWfJw>

para os pesquisadores anotarem onde quer que julguem importante sobre o protocolo

Na parte inferior do questionário, uma opção foi disponibilizada para sugestões gratuitas sobre melhorias do protocolo. O objetivo principal desta avaliação foi refinar o protocolo com base nas respostas e sugestões dos pesquisadores.

Enviamos o questionário de avaliação para sete pesquisadores na área de Sistemas de Informação, Engenharia de Software e Engenharia Elétrica (ver Tabela - 1.5). Disponibilizamos o questionário por 20 dias. Nós tivemos cinco respostas.

Tabela 1.5: Pesquisadores que avaliaram o protocolo de SMS

ID	Nível de educação	Instituição	Área Especialista
R.1	Ph.D.	Universidade Federal de Tecnologia do Paraná	Sistemas de informação
R.2	Ph.D.	Universidade Federal do Pampa	Engenharia de software
R.3	Ph.D.	Universidad Politécnica de València Espanha	Engenharia elétrica
R.4	Ph.D.	Universidade Federal do Paraná	Engenharia de software
R.5	M.Sc.	Instituto SENAI de Tecnologia em Metalmecânica	Engenharia elétrica

Os pesquisadores convidados fizeram uma avaliação muito produtiva. Todos consideram que a pesquisa pode gerar dados satisfatórios com base no contexto apresentado sobre o tema. A maioria também concorda que o Search String é adequado com base nas questões de pesquisa.

Outro ponto de concordância é que as fontes e os tipos de pesquisa podem abranger estudos relevantes, mas com relação aos critérios de inclusão e exclusão, alguns discordaram, por exemplo, que a seleção de artigos está limitada a apenas os últimos 10 anos de pesquisa. Nesse caso, a justificativa seria evitar o estudo duplicado que foi atualizado ou continuado.

Uma sugestão foi realizar uma avaliação baseada no número de citações de um determinado estudo. Essa sugestão foi aplicada em conjunto com o questionário de critérios.

Busca Eletrônica

Realizamos a busca eletrônica de agosto / 2017 a outubro / 2017. Nós derivamos nossa *string* de busca geral para cada uma das fontes eletrônicas, como apresentado em Tabela - 1.6.

A busca eletrônica retornou 251 estudos, como podemos ver na Tabela - 1.7, coluna “Retornado”. A Science Direct foi a fonte mais voltada, com 125 estudos, seguida pela Scopus, com 62 estudos.

Tabela 1.6: Fontes eletrônicas e sequências de pesquisa adaptadas

Fonte Eletrônica	String de pesquisa adaptada
ACM	("software" + "product line" "product-line" "product family" "product-family" "product-families" "family of products" "variability" + "model-based testing" "model based testing" "TBM")
Google Scholar	("Software") AND ("product line" or "product-line" or "product family" or "product-family" or "product-families" or "family of products" or "variability") AND ("model-based testing" or "model based testing" or "TBM")
IEEE	(("software") AND ("product line" OR "product-line" OR "product family" OR "product-family" OR "product-families" OR "family of products" OR "variability") AND ("model-based testing" OR "model based testing" OR "TBM"))
Springer	((software) AND ("product line" or "product-line" or "product family" or "product-family" or "product-families" or "family of products" or "variability") AND ("model-based testing" or "model based testing" or TBM))
Scopus	(TITLE-ABS-KEY (software) AND TITLE-ABS-KEY (product line OR product-line OR product family OR product-family OR product-families OR family of products OR variability) AND TITLE-ABS-KEY (model-based testing OR model based testing OR TBM)) AND (PUBYEAR > 2005)
Science Direct	("software") AND ("product line" OR "product-line" OR "product family" OR "product-family" OR "product-families" OR "family of products" OR "variability") AND ("model-based testing" OR "model based testing" OR "TBM")
IET Digital Library	(("software") AND ("product line" OR "product-line" OR "product family" OR "product-family" OR "product-families" OR "family of products" OR "variability") AND ("model-based testing" OR "model based testing" OR "TBM"))

Tabela 1.7: Número de estudos de fontes eletrônicas

Fonte Eletrônica	Retornou	Aceitos	Duplicados	Rejeitados
Science Direct	125	07	00	118
Scopus	62	10	21	31
IEEE Xplore	25	00	00	25
ACM	20	16	01	03
IET Digital Library	09	00	00	09
Springer	07	05	01	01
Google Scholar	03	02	00	01
Total	251	40	23	188

Busca Manual

A busca manual retornou 20 estudos de conferências, oficinas e periódicos, conforme apresentado na Tabela - 1.8, coluna “ Retornado ”.

A.2.3 Processo de seleção

A Figura - 1.3 mostra o processo de seleção do SMS.

As próximas seções apresentam como realizamos o processo de seleção.

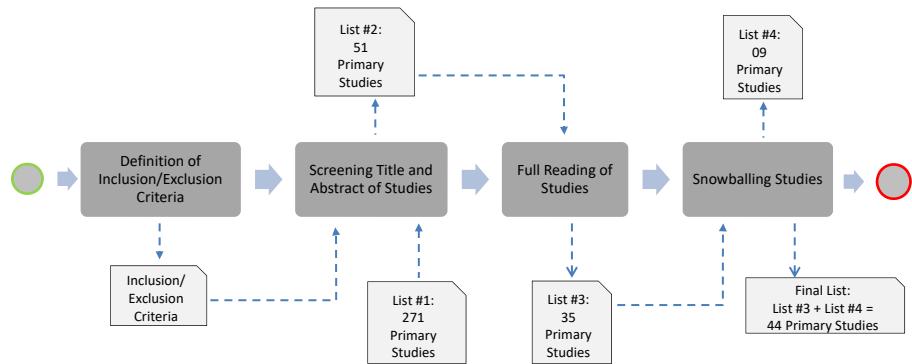
Definição de Critérios de Inclusão e Exclusão

Para selecionar estudos relevantes e contribuir para responder às questões de pesquisa deste estudo, definimos os seguintes critérios de inclusão e exclusão:

- Critério de Inclusão:

Tabela 1.8: Número de estudos de fontes manuais

Fonte	Retornados	Aceitos	Rejeitados
Software & Systems Modeling	01	01	00
Model Driven Engineering Languages and Systems	01	00	01
Software Quality Journal	01	01	00
Software Testing, Verification and Validation (ICST)	01	01	00
Computer and Information Science (ICIS)	01	01	00
ACM SIGSOFT Software Engineering Notes	01	01	00
Workshop on Variability Modeling of Software-Intensive Systems	01	00	01
International Software Product Line Conference	01	00	01
IEEE Software	01	00	01
Google Scholar	11	06	05
Total	20	11	09

**Figura 1.3:** Processo Seletivo SMS

- IC.1 - estudos que discutem atividades de TBM realizadas exclusivamente no domínio LPS.

- **Critérios de exclusão:**

- EC.1 - estudos não abordam atividades de TBM realizadas exclusivamente no domínio LPS;
- EC.2 - estudos com texto incompleto;
- EC.3 - estudos em um idioma diferente do inglês para promover a disseminação e reproduzibilidade internacional;
- EC.4 - estudos com menos de quatro páginas, pois acreditamos que eles não têm espaço para contribuições relevantes;
- EC.5 - estudos duplicados;
- EC.6 - estudos indisponíveis, mesmo em contato com os autores.

Títulos de triagem e resumos de estudos

Depois de definir os critérios de inclusão e exclusão, aplicamos os mesmos aos estudos lendo títulos e resumos de Lista # 1 (271 estudos). Tal leitura filtrou Lista # 1 para aceitar 51 estudos (Lista # 2), onde 40 estudos provêm da busca eletrônica (Tabela - 1.7, coluna “ Aceito ”) e 11 da busca manual (Tabela - 1.8, coluna “ Aceito ”).

Tabela 1.9: Artigos selecionados para leitura completa

Título	Ano	Local	Tipo Pub.
Delta-Oriented Model-Based LPS Regression Testing (Lity et al., 2012)	2012	ACM	Evento
Industrial Evaluation of Pairwise LPS Testing with MoSo-PoLiTe (Steffens et al., 2012a)	2012	ACM	Evento
Model-Based Coverage-Driven Test Suite Generation for Software Product Lines (Cichos et al., 2011)	2011	ACM	Journal
MoSo-PoLiTe - Tool Support for Pairwise and Model-Based Software Product Line Testing (Oster et al., 2011b)	2011	ACM	Evento
MPLM - MaTeLo Product Line Manager (Samih e Bogusch, 2014a)	2014	ACM	Evento
On the use of test cases in model-based software product line development (Knapp et al., 2014)	2014	ACM	Evento
Pairwise Feature-Interaction Testing for LPSs: Potentials and Limitations (Oster et al., 2011a)	2011	ACM	Evento
Deriving Usage Model Variants for Model- based Testing: An Industrial Case Study (Samih et al., 2014b)	2014	IEEE	Evento
Model-based Software Product Line Testing by Coupling Feature Models with Hierarchical Markov Chain Usage Models (Gebizli e Sözer, 2016)	2016	IEEE	Evento
Model-Based Test Design of Product Lines: Raising Test Design to the Product Line Level (Lackner et al., 2014)	2014	IEEE	Journal
Requirements-Based Delta-Oriented LPS Testing (Dukaczewski et al., 2013)	2013	IEEE	Evento
Using Feature Model to Support Model-Based Testing of Product Lines: (Wang et al., 2013b) An Industrial Case Study	2013	IEEE	Journal
An automated Model-based Testing Approach in Software Product Lines (García Gutiérrez et al., 2010) Using a Variability Language	2010	Politécnica Arquivo digital UPM	Evento
Automated Product Line Methodologies to Support Model-Based Testing (Wang et al., 2013a)	2013	CEUR Proceedings	Evento
Behavioural Model Based Testing of Software Product Lines (Devroey, 2017)	2014	ACM	Evento
Feature Model-based Software Product Line Testing (Oster, 2012)	2012	TUprints Darmstadt publication service	Journal
Model-based pairwise testing for feature interaction coverage in software product line engineering (Lochau et al., 2012a)	2011	Springer	Journal
Model-based Test Generation for Software Product Line (Cai e Zeng, 2013)	2013	IEEE	Evento
Model-Based Testing for Software Product Lines (Olimpiew, 2008b)	2008	Springer	Evento
PLETS - A Product Line of Model- Based Testing Tools (Rodrigues et al., 2012)	2013	PUC-RS	Evento
Top-Down and Bottom-Up Approach for Model-Based Testing of Product Lines (Weißleder e Lackner, 2013a)	2013	EPTCS	Evento
A Product Line Modeling and Configuration Methodology to Support Model-Based Testing: An Industrial Case Study (Ali et al., 2012)	2012	Springer	Journal
Coverage Criteria for Behavioural Testing of Software Product Lines (Devroey et al., 2014a)	2014	Springer	Evento

Tabela - 1.9 Continuação da página anterior

Título	Ano	Local	Tipo Pub.
A Model Based Testing Approach for Model-Driven Development and Software Product Lines (Lamancha et al., 2010b)	2010	Springer	Evento
A Vision for Behavioural Model-Driven Validation of Software Product Lines (Devroey et al., 2012)	2012	Springer	Evento
Abstract Test Case Generation for Behavioural Testing of Software Product Lines (Devroey et al., 2014b)	2014	ACM	Evento
Applying Incremental Model Slicing to Product-Line Regression Testing (Lity et al., 2016)	2016	Springer	Journal
Automated Testing of Software-as-a-Service Configurations using a Variability Language (Patel e Shah, 2015)	2015	ACM	Evento
Delta-Oriented FSM-Based Testing (Varshosaz et al., 2015)	2015	Springer	Evento
Incremental Model-Based Testing of Delta-oriented Software Product Lines (Lochau et al., 2012b)	2012	Springer	Journal
Model Based Testing in Software Product Lines (Reales et al., 2011b)	2011	Springer	Evento
Model-Based Testing (Lochau et al., 2014)	2014	Springer	Evento
Parameterized Preorder Relations for Model-Based Testing of Software Product Lines (Lochau e Kamischke, 2012)	2012	Springer	Evento
Poster:VIBeS, Transition System Mutation Made Easy (Devroey et al., 2015)	2015	IEEE	Evento
Spinal Test Suites for Software Product Lines (Beohar e Mousavi, 2014b)	2014	EPTCS	Evento
Automated model-based testing using the UML testing profile and QVT (Lamancha et al., 2009b)	2009	ACM	Evento
Relating Variability Modeling and Model-Based Testing for Software Product Lines Testing (Samih e Baudry, 2012)	2012	ICTSS	Evento
An Evaluation of Model-Based Testing in Embedded Applications (Weißleder e Schlingloff, 2014)	2014	IEEE	Evento
Assessing Software Product Line Testing Via Model-Based Mutation (Henard et al., 2013)	2013	IEEE	Evento
An Application to Similarity Testing			
Automated and Scalable T-wise Test Case Generation Strategies (Perrouin et al., 2010) for Software Product Lines	2010	IEEE	Evento
Model-based Testing of System Requirements using UML Use Case Models (Hasling et al., 2008)	2008	IEEE	Evento
Successive refinement of models for model-based testing to increase system test effectiveness (Gebizli et al., 2016)	2016	IEEE	Evento
A Software Product Line for Model-Based Testing Tools (Rodrigues et al., 2012)	2012	PUC-RS	Evento
Reusing State Machines for Automatic Test Generation in Product Lines (Weißleder et al., 2008)	2008	MoTip	Evento
A Nâovel Markov Boundary Based Feature Subset Selection Algorithm (de Moraes e Aussem, 2010)	2010	Elsevier	Journal
A Nâovel Model-Based Testing Approach for Software Product Lines (Damiani et al., 2017)	2016	Springer	Evento
Abstract Test Case Generation for Behavioural Testing of Software Product Lines (Devroey et al., 2014b)	2014	ACM	Evento
An Overview on Test Generation from Functional Requirements (Escalona et al., 2011)	2011	Elsevier	Journal
Basic Behavioral Models for Software Product Lines:Expressiveness and Testing (Beohar et al., 2016)	2016	Elsevier	Journal
Bottom-up reuse for multi-level testing (Pérez e Kaiser, 2010)	2010	Elsevier	Journal
Colored Model Based Testing for Software Product Lines (Farrag, 2010)	2010	semanticscholar	Gray Literature
IncLing:Efficient Product-Line Testing Using Incremental Pairwise Sampling (Al-Hajjaji et al., 2016)	2016	ACM	Evento
Input-output conformance testing based on featured transition systems (Beohar e Mousavi, 2014a)	2014	ACM	Evento
Model-based testing of automotive systems (Bringmann e Krämer, 2008)	2008	IEEE	Evento
Model-based system testing of software product families (Reuys et al., 2005)	2005	Springer	Evento
Model-based testing for applications derived from software product lines (Olimpiew e Gomaa, 2005)	2005	ACM	Evento
Reducing the Concretization Effort in FSM-Based Testing of Software Product Lines (Fragal et al., 2017)	2016	IEEE	Evento
Refinement-based testing of delta-oriented product lines (Damiani et al., 2013)	2013	ACM	Evento

Tabela - 1.9 Continuação da página anterior

Título	Ano	Local	Tipo Pub.
Risk-based integration testing of software product lines (Lachmann et al., 2017)	2016	ACM	Evento
Uncertainty-wise evolution of test ready models (Zhang et al., 2017)	2016	Elsevier	Journal

Leitura Completa de Estudos

Dada Lista # 2, lemos cada um dos 51 estudos. Rejeitamos 16 deles (veja Tabela - 1.10) com base nos critérios de exclusão da Seção A.2.3, fornecendo assim Lista # 3 com 35 estudos.

Tabela 1.10: Estudos removidos de acordo com os critérios de exclusão

Título	EC.1	EC.2	EC.3	EC.4	EC.5	EC.6
A Nãovel Markov Boundary Based Feature Subset Selection Algorithm (de Morais e Aussem, 2010)	X	-	-	-	-	-
A Nãovel Model-Based Testing Approach for Software Product Lines (Damiani et al., 2017)	-	-	-	-	X	-
Abstract Test Case Generation for Behavioural Testing of Software Product Lines(Devroey et al., 2014b)	X	-	-	-	-	-
An Overview on Test Generation from Functional Requirements (Escalona et al., 2011)	X	-	-	-	-	-
Basic Behavioral Models for Software Product Lines:Expressiveness and Testing Pre-Orders (Beohar et al., 2016)	-	-	-	X	-	-
Bottom-up reuse for multi-level testing (Pérez e Kaiser, 2010)	X	-	-	-	-	-
Colored Model Based Testing for Software Product Lines (Farrag, 2010)	X	-	-	-	X	-
IncLing:Efficient Product-Line Testing Using Incremental Pairwise Sampling (Al-Hajjaji et al., 2016)	X	-	-	-	-	-
Input-output conformance testing based on featured transition systems (Beohar e Mousavi, 2014a)	X	-	-	-	-	-
Model-based testing of automotive systems (Bringmann e Krämer, 2008)	-	-	-	X	-	-
Model-based system testing of software product families (Reuys et al., 2005)	-	-	-	-	X	-
Model-based testing for applications derived from software product lines (Olimpiew e Gomaa, 2005)	-	-	-	-	X	-
Reducing the Concretization Effort in FSM-Based Testing of Software Product Lines (Fragal et al., 2017)	X	-	-	-	-	-
Refinement-based testing of delta-oriented product lines (Damiani et al., 2013)	X	-	-	-	-	-
Risk-based integration testing of software product lines (Lachmann et al., 2017)	X	-	-	-	-	-
Uncertainty-wise evolution of test ready models (Zhang et al., 2017)	X	-	-	-	-	-

Estudos derivados do Snowballing

Realizamos bolas de neve invertidas em Lista # 3, onde avaliamos as referências dos estudos. Dez estudos retornaram de bola de neve, e os critérios de inclusão e exclusão foram aplicados a eles. Portanto, um deles foi duplicado conforme apresentado em Tabela - 1.11. Os nove estudos restantes compuseram Lista # 4.

Tabela 1.11: Estudo do Snowballing

ID	Fonte do estudo	ID	Estudo do Snowballing	Estatus
S24	A Model-Based Testing Approach for Model-Driven Development and Software Product Lines (Lamancha et al., 2010b)	S36	Automated model-based testing using the UML testing profile and QVT (Lamancha et al., 2009b)	Aceito
S8	Deriving Usage Model Variants for Model-based Testing: An Industrial Case Study (Samih et al., 2014b)	S37	Relating Variability Modeling and Model-Based Testing for Software Product Lines Testing (Samih e Baudry, 2012)	Aceito
S10	Model-Based Test Design of Product Lines: Raising Test Design to the Product Line Level (Lackner et al., 2014)	S38	An Evaluation of Model-Based Testing in Embedded Applications (Weißleder e Schlingloff, 2014)	Aceito
S10	Model-Based Test Design of Product Lines: Raising Test Design to the Product Line Level (Lackner et al., 2014)	S39	Assessing Software Product Line Testing Via Model-Based Mutation An Application to Similarity Testing (Henard et al., 2013)	Aceito
S5	MPLM-MaTeLo Product Line Manager (Samih e Bogusch, 2014a)	S40	Automated and Scalable T-wise Test Case Generation Strategies for Software Product Lines (Perrouin et al., 2010)	Aceito
S20	PLETS - a Product Line of Model-Based Testing Tools	S41	Model based testing of system requirements using UML use case models (Hasling et al., 2008)	Aceito
S9	Model-based Software Product Line Testing by Coupling Feature Models with Hierarchical Markov Chain Usage Models (Gebizli e Sözer, 2016)	S42	Successive refinement of models for model-based testing to increase system test effectiveness (Gebizli et al., 2016)	Aceito
S20	PLETS - a Product Line of Model-Based Testing Tools ()	S43	A Software Product Line for Model-Based Testing Tools (Rodrigues et al., 2012)	Aceito
S21	Top-Down and Bottom-Up Approach for Model-Based Testing of Product Lines (Weißleder e Lackner, 2013a)	S44	Reusing State Machines for Automatic Test Generation in Product Lines (Weißleder et al., 2008)	Aceito
S5	MPLM-MaTeLo Product Line Manager (Samih e Bogusch, 2014a)	–	An approach to derive usage models variants for model-based (Samih et al., 2014a)	Duplicata

A Tabela - 1.12 apresenta o conjunto final de estudos selecionados para este SMS, composto por 44 estudos (Lista Final), dos quais:os primeiros 35 são de pesquisas eletrônicas e manuais e os restantes nove são do *Snowballing*.

Tabela 1.12: Lista final de estudos

ID	Título	Ano	Local	Tipo de Pub.
S1	Delta-Oriented Model-Based LPS Regression Testing (Lity et al., 2012)	2012	ACM	Evento
S2	Industrial Evaluation of Pairwise LPS Testing with MoSo-PoLiTe (Steffens et al., 2012a)	2012	ACM	Evento
S3	Model-Based Coverage-Driven Test Suite Generation for Software Product Lines (Cichos et al., 2011)	2011	ACM	Journal
S4	MoSo-PoLiTe - Tool Support for Pairwise and Model-Based Software Product Line Testing (Oster et al., 2011b)	2011	ACM	Evento
S5	MPLM - MaTeLo Product Line Manager (Samihi e Bogusch, 2014a)	2014	ACM	Evento
S6	On the use of test cases in model-based software product line development (Knapp et al., 2014)	2014	ACM	Evento
S7	Pairwise Feature-Interaction Testing for LPSs:Potentials and Limitations (Oster et al., 2011a)	2011	ACM	Evento
S8	Deriving Usage Model Variants for Model- based Testing: An Industrial Case Study (Samihi et al., 2014b)	2014	IEEE	Evento
S9	Model-based Software Product Line Testing by Coupling Feature Models with Hierarchical Markov Chain Usage Models (Gebizli e Sözer, 2016)	2016	IEEE	Evento
S10	Model-Based Test Design of Product Lines: Raising Test Design to the Product Line Level (Lackner et al., 2014)	2014	IEEE	Journal
S11	Requirements-Based Delta-Oriented LPS Testing (Dukaczewski et al., 2013)	2013	IEEE	Evento
S12	Using Feature Model to Support Model-Based Testing of Product Lines: An Industrial Case Study (Wang et al., 2013b)	2013	IEEE	Journal
S13	An automated Model-based Testing Approach in Software Product Lines Using a Variability Language (García Gutiérrez et al., 2010)	2010	Politécnica Arquivo digital UPM	Evento
S14	Automated Product Line Methodologies to Support Model-Based Testing (Wang et al., 2013a)	2013	CEUR Proceedings	Evento
S15	Behavioural Model Based Testing (Devroey, 2017) of Software Product Lines	2014	ACM	Evento
S16	Feature Model-based Software Product Line Testing (Oster, 2012)	2012	TUprints Darmstadt publication service	Journal
S17	Model-based pairwise testing for feature interaction coverage in software product line engineering (Lochau et al., 2012a)	2011	Springer	Journal
S18	Model-based Test Generation for Software Product Line (Cai e Zeng, 2013)	2013	IEEE	Evento
S19	Model-Based Testing for Software Product Lines (Olimpiew, 2008b)	2008	Springer	Evento
S20	PLETS - A Product Line of Model- Based Testing Tools (Rodrigues et al., 2012)	2013	PUC-RS	Evento
S21	Top-Down and Bottom-Up Approach for Model-Based Testing of Product Lines (Weißleder e Lackner, 2013a)	2013	EPTCS	Evento
S22	A Product Line Modeling and Configuration Methodology to Support Model-Based Testing: An Industrial Case Study (Ali et al., 2012)	2012	Springer	Journal
S23	Coverage Criteria for Behavioural Testing of Software Product Lines (Devroey et al., 2014a)	2014	Springer	Evento
S24	A Model Based Testing Approach for Model-Driven Development and Software Product Lines (Lamancha et al., 2010b)	2010	Springer	Evento
S25	A Vision for Behavioural Model-Driven Validation of Software Product Lines (Devroey et al., 2012)	2012	Springer	Evento

Tabela - 1.12 Continuação da página anterior

ID	Título	Ano	Local	Tipo de Pub.
S26	Abstract Test Case Generation for Behavioural Testing of Software Product Lines (Devroey et al., 2014b)	2014	ACM	Evento
S27	Applying Incremental Model Slicing to Product-Line Regression Testing (Lity et al., 2016)	2016	Springer	Journal
S28	Automated Testing of Software-as-a-Service Configurations using a Variability Language (Patel e Shah, 2015)	2015	ACM	Evento
S29	Delta-Oriented FSM-Based Testing (Varshosaz et al., 2015)	2015	Springer	Evento
S30	Incremental Model-Based Testing of Delta-oriented Software Product Lines (Lochau et al., 2012b)	2012	Springer	Journal
S31	Model Based Testing in Software Product Lines (Reales et al., 2011b)	2011	Springer	Evento
S32	Model-Based Testing (Lochau et al., 2014)	2014	Springer	Evento
S33	Parameterized Preorder Relations for Model-Based Testing of Software Product Lines (Lochau e Kamischke, 2012)	2012	Springer	Evento
S34	Poster: VIBeS, Transition System Mutation Made Easy (Devroey et al., 2015)	2015	IEEE	Evento
S35	Spinal Test Suites for Software Product Lines (Beohar e Mousavi, 2014b)	2014	EPTCS	Evento
S36	Automated model-based testing using the UML testing profile and QVT (Lamancha et al., 2009b)	2009	ACM	Evento
S37	Relating Variability Modeling and Model-Based Testing for Software Product Lines Testing (Samih e Baudry, 2012)	2012	ICTSS	Evento
S38	An Evaluation of Model-Based Testing in Embedded Applications (Weißleder e Schlingloff, 2014)	2014	IEEE	Evento
S39	Assessing Software Product Line Testing Via Model-Based Mutation An Application to Similarity Testing (Henard et al., 2013)	2013	IEEE	Evento
S40	Automated and Scalable T-wise Test Case Generation Strategies (Perrouin et al., 2010) for Software Product Lines	2010	IEEE	Evento
S41	Model-based Testing of System Requirements using UML Use Case Models (Hasling et al., 2008)	2008	IEEE	Evento
S42	Successive refinement of models for model-based testing to increase system test effectiveness (Gebizli et al., 2016)	2016	IEEE	Evento
S43	A Software Product Line for Model-Based Testing Tools (Rodrigues et al., 2012)	2012	PUC-RS	Evento
S44	Reusing State Machines for Automatic Test Generation in Product Lines (Weißleder et al., 2008)	2008	MoTip	Evento

A.2.4 Processo de Extração

A Figura - 1.4 mostra o processo de extração do SMS.

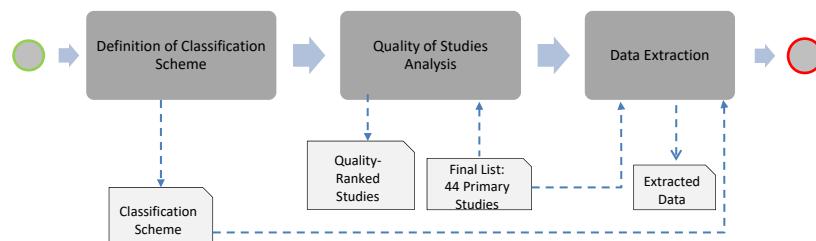


Figura 1.4: Processo de extração de SMS

Definição do Esquema de Classificação

Análise de Qualidade de Estudos

Para avaliação da qualidade dos estudos, definimos um questionário e aplicamos em cada estudo. Portanto, elaboramos as três perguntas a seguir:

- O par de estudos é revisado?
- O objetivo do estudo é claro?
- A proposta do estudo foi avaliada / validada?

Para cada questão, havia três alternativas, nas quais apenas uma delas poderia ser escolhida: “Não”, “Sim” e “Parcialmente”. Estudos com “Não” para quaisquer questões são automaticamente descartados. Nós cuidadosamente lemos novamente os estudos com ”Parcialidade” como uma resposta para qualquer pergunta.

Adotamos este procedimento, assim como a aplicação dos critérios de inclusão e exclusão para garantir contribuições mínimas dos estudos selecionados para o processo de extração.

Realizamos a extração de dados para coletar informações necessárias para responder às questões de pesquisa, bem como analisar os estudos dos critérios de seleção. Portanto, definimos os seguintes critérios de qualidade (QC) para garantir a qualidade mínima dos estudos:

- QC.1 - O estudo descreve claramente o propósito da pesquisa?
- QC.2 - O campo de ação é compatível com a área de pesquisa?
- QC.3 - Como o estudo é avaliado?
- QC.4 - Houve uma coleta de dados apropriada?
- QC.5 - Houve uma análise de dados apropriada?
- QC.6 - O estudo apresenta resultados consistentes com seus objetivos?
- QC.7 - Os resultados contribuem para o processo realizado para a pesquisa?

Extração de dados

Os próximos itens apresentam a lista de dados que definimos como extraídos de cada estudo primário:

- Dados bibliométricos:Título, Autor(es), Ano de Publicação, Fonte de Publicação, Tipo de Publicação, Tipo de Documento;
- Domínio da Solução;
- Inclui variabilidade ?;
- *Feature Interaction*?
- Método de Execução;
- Item de teste usado;
- Nível de teste aplicado;
- Abordagem de Teste Usada;
- Nível de cobertura;
- Trate a rastreabilidade ?;
- Finalidade do TBM;
- Teste de Requisito Não-Funcional é Suportado ?;
- Artefato de Origem;
- Artefato Intermediário;
- Uso de Ferramentas;
- Tempo de Ligação;
- Atividades do TBM avaliadas;
- Resultados da pesquisa;
- Método de Coleta de Evidências;
- Local de validação;
- Tipo de contribuição.

A.2.5 Processo de Análise

A Figura - 1.5 mostra o processo de análise do SMS.

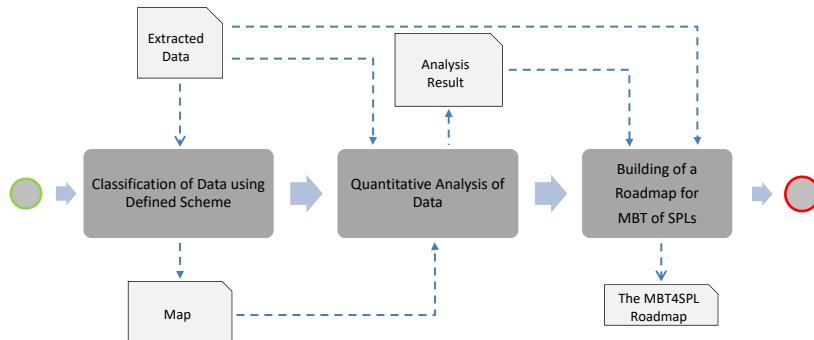


Figura 1.5: Processo de Análise SMS

Classificação de dados usando o esquema definido

Análise Quantitativa de Dados

Criação de um roteiro para TBM de LPSs

A.3 Resultados

A.3.1 Caracterização dos Estudos

Estudos por ano

É importante entender como o tema de pesquisa do TBM aplicado ao LPS se comportou ao longo dos últimos anos. Portanto, Tabela - 1.13 apresenta o número de estudos por ano e a Figura - 1.6 mostra a distribuição desses estudos ao longo dos anos.

Tabela 1.13: Número de estudos por ano

Tipo de local	2008	2009	2010	2011	2012	2013	2014	2015	2016	Count
Conferência	1	1	1	2	-	3	6	3	2	19
Workshop	2	1	1	1	4	3	1	-	-	13
Journal	-	-	-	2	3	1	1	-	1	8
Simpósio	-	-	-	-	2	-	2	-	-	4
Total	3	2	2	5	9	7	10	3	3	44

Com base na Figura - 1.6, observamos um número crescente de estudos ao longo dos anos. Especial atenção é dada a 2014 com 10 estudos. Além disso, com exceção de 2006 e 2007, sem ocorrência, os estudos são bem distribuídos na maioria dos anos, como em 2008, 2009, 2010, 2011, 2015 e 2016.

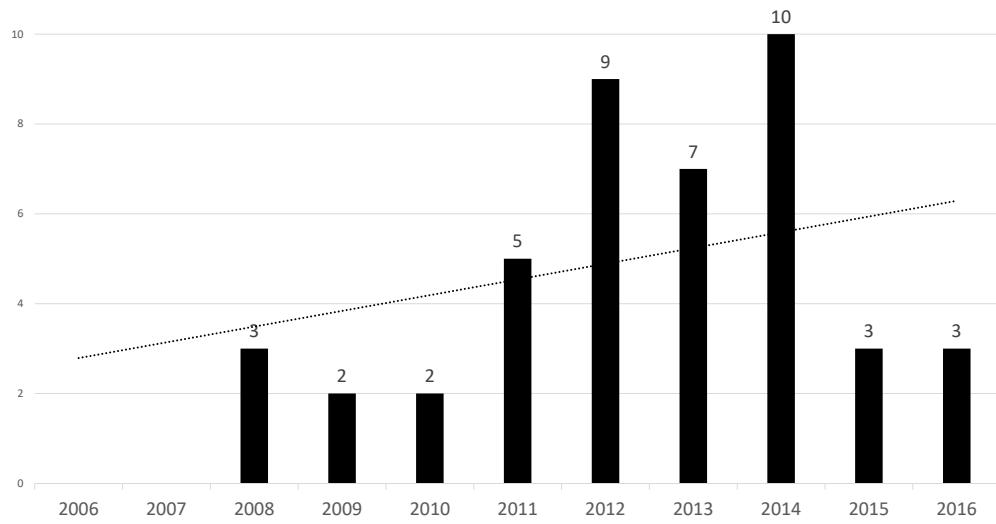


Figura 1.6: Distribuição de estudos por ano

Qualidade de Estudos

Para garantir a qualidade mínima dos estudos, realizamos uma avaliação de qualidade conforme planejado na seção A.2.4, levando em consideração a aplicação dos critérios de qualidade da Seção A.2.4.

Todos os estudos da Lista Final têm qualidade mínima com base em tais critérios, como podemos ver na Tabela - 1.14 com a maioria das respostas “ Sim ”.

Tabela 1.14: Avaliação de Qualidade de Estudos

ID do Estudo	QC.1-Propósito	QC.2-Compatibilidade	QC.3-Avaliação	QC.4-Coleção de dados	QC.5-Análise de dados	QC.6-Consistência dos Resultados	QC.7-Contribuição
S1	Sim	Não	Sim	Sim	Sim	Sim	Sim
S2	Sim	Não	Sim	Sim	Sim	Sim	Sim
S3	Sim	Não	Sim	Sim	Sim	Sim	Sim
S4	Sim	Não	Não Informado	Sim	Sim	Sim	Sim
S5	Sim	Não	Não Informado	Não Informado	Não Informado	Não Informado	Sim
S6	Sim	Não	Não Informado	Sim	Sim	Sim	Sim
S7	Sim	Não	Sim	Sim	Sim	Sim	Sim
S8	Sim	Não	Sim	Sim	Sim	Sim	Sim
S9	Sim	Não	Sim	Sim	Sim	Sim	Sim
S10	Sim	Não	Sim	Sim	Sim	Sim	Sim
S11	Sim	Não	Não	Sim	Sim	Sim	Sim
S12	Sim	Não	Sim	Sim	Sim	Sim	Sim
S13	Sim	Não	Não Informado	Sim	Sim	Sim	Sim
S14	Sim	Não	Sim	Sim	Sim	Sim	Sim
S15	Sim	Não	Não Informado	Sim	Sim	Sim	Sim
S16	Sim	Não	Sim	Sim	Sim	Sim	Sim
S17	Sim	Não	Sim	Sim	Sim	Sim	Sim
S18	Sim	Não	Não Informado	Sim	Sim	Sim	Sim
S19	Sim	Não	Sim	Sim	Sim	Sim	Sim
S20	Sim	Não	Não Informado	Sim	Sim	Sim	Sim
S21	Sim	Não	Sim	Sim	Sim	Sim	Sim
S22	Sim	Não	Sim	Sim	Sim	Sim	Sim
S23	Sim	Não	Sim	Sim	Sim	Sim	Sim
S24	Sim	Não	Não Informado	Sim	Sim	Sim	Sim
S25	Sim	Não	Sim	Sim	Sim	Sim	Sim
S26	Sim	Não	Sim	Sim	Sim	Sim	Sim
S27	Sim	Não	Sim	Sim	Sim	Sim	Sim
S28	Sim	Não	Sim	Sim	Sim	Sim	Sim
S29	Sim	Não	Sim	Sim	Sim	Sim	Sim
S30	Sim	Não	Sim	Sim	Sim	Sim	Sim
S31	Sim	Não	Sim	Não Informado	Não Informado	Não Informado	Sim
S32	Sim	Não	Sim	Não Informado	Não Informado	Não Informado	Sim
S33	Sim	Não	Sim	Não	Não Informado	Não Informado	Sim
S34	Sim	Não	Sim	Não Informado	Não Informado	Não Informado	Sim
S35	Sim	Não	Sim	Não Informado	Não Informado	Não Informado	Sim
S36	Sim	Não	Sim	Não Informado	Não Informado	Não Informado	Sim
S37	Sim	Não	Sim	Não Informado	Não Informado	Não Informado	Sim
S38	Sim	Não	Sim	Não Informado	Não Informado	Não Informado	Sim
S39	Sim	Não	Sim	Não Informado	Não Informado	Não Informado	Sim
S40	Sim	Não	Sim	Não Informado	Não Informado	Não Informado	Sim
S41	Sim	Não	Sim	Não Informado	Não Informado	Não Informado	Sim
S42	Sim	Não	Sim	Não Informado	Não Informado	Não Informado	Sim
S43	Sim	Não	Sim	Não Informado	Não Informado	Não Informado	Sim
S44	Sim	Não	Sim	Não Informado	Não Informado	Não Informado	Sim

A.3.2 Resultados em questões de pesquisa

Nesta seção apresentamos os resultados com base em cada questão de pesquisa pré-definida.

RQ.1: Para quais domínios do aplicativo LPS, tipos de solução e propostas TBM são usados?

Para responder a essa pergunta, levamos em consideração o TBM aplicado a domínios LPS (por exemplo, software, automotivo), tipos de solução de área de pesquisa LPS (por exemplo, modelagem de recursos) e o tipo de proposta de estudos (por exemplo, abordagem, técnica, ferramenta).

Identificamos sete domínios de aplicação distintos nos quais o TBM for LPS é aplicado. Tabela - 1.15 lista cada domínio e respectivos estudos e a Figura - 1.7 mostra o número de estudos por domínio.

Tabela 1.15: TBM de domínios de aplicativos LPS

Domínio de Aplicação	ID do Estudo	Count
Aeroespacial	S5, S8	02
Automotivo	S1, S3, S4, S7, S10, S11, S16, S17, S30	09
Eletrônico	S2, S9, S42, S44	04
Fabricação	S6	01
Medicina	S41	01
Software	T13, S15, S18, S19, S20, S21, S23, S24, S25, S26, S27, S28, S29, S31, S32, S33, S34, S35, T36, S37, S38, S39, S40, S43	24
Telecomunicação	S12, S14, S22	03
TOTAL		44

O domínio de aplicação *Software* tem a maioria das ocorrências com 24 estudos, seguido por *Automotive* com nove, que utiliza software embutido em seus produtos.

Aerospacial MPLM é uma das poucas ferramentas que concentram o maior esforço na questão da solução de variabilidade na geração de casos de teste, (Samih e Bogusch, 2014a) em seu recurso de estudo a ferramenta Matelo, juntamente com outras ferramentas permite derivar a variante do modelo a partir de um o conjunto desejado e, assim, gerar casos de teste para um projeto experimental na Airbus defense & Space. O TBM auxilia na reutilização para geração dos casos de teste, o outro estudo que trabalha no mesmo domínio é uma extensão deste citado.

Automotivo Lity et al. (Lity et al., 2012) apresenta a abordagem delta, que trabalha com elementos cruzados de um elemento chamado G, onde os casos de teste são definidos com uma cobertura satisfatória. Os artefatos de teste são desenvolvidos de forma

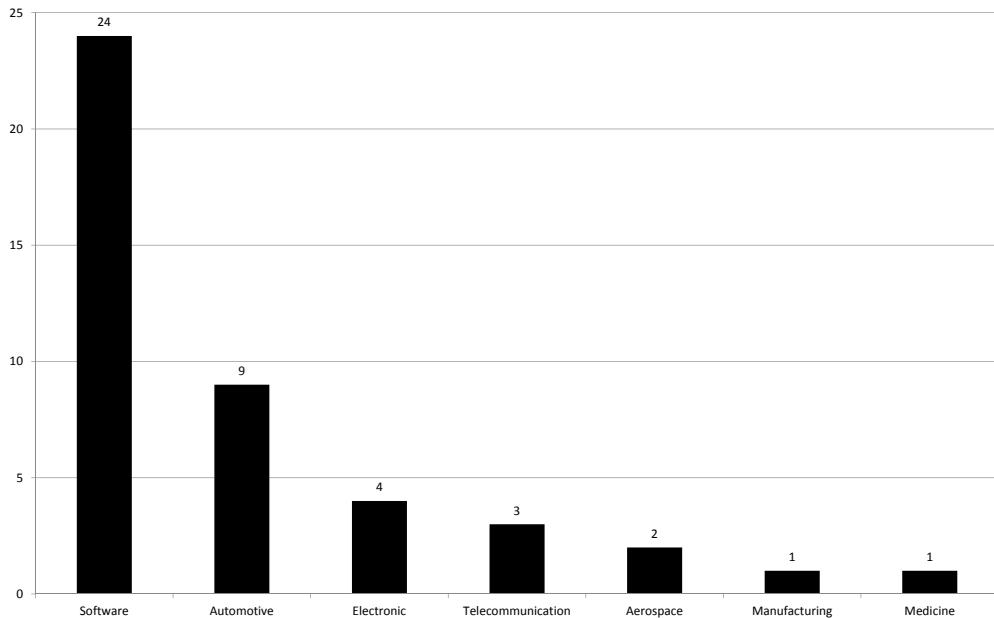


Figura 1.7: Automação TBM para LPS de abordagens de teste

incremental à medida que surgem novas variantes de produtos. Consideramos que o fator de regressão e a adaptabilidade dos artefatos de teste podem ter uma contribuição significativa ao serem trabalhados em um LPS, embora não tenham enfoque no TBM. O uso de software embarcado em produtos automotivos cresce exponencialmente, o TBM tem ajudado a gerar os testes desses produtos que sofrem variações, onde existe um núcleo central, e cada variação pode ou não conter uma especificidade única das demais.

Eletrônico Seguindo esse conceito (Steffens et al., 2012a) usa modelos para gerar subgrupos ou casos de teste desses modelos, aborda a questão da interação de recursos com a cobertura de teste e gerencia as variabilidades como itens cruzados. MoSo-PoLiTe, uma ferramenta que propõe fazer a derivação dos produtos a partir do modelo principal, utilizando o teste combinatório automatizado. O TBM atua na geração de casos de teste para subgrupos de produtos, além de buscar garantir uma maior cobertura de erros de funcionalidade, já que se trata de um domínio eletrônico, a garantia de qualidade deve ser com uma margem significativamente maior.

Fabricação (Knapp et al., 2014) apresenta um procedimento associado a um conjunto de ferramentas para atribuir o resultado de um caso de teste a um membro arbitrário de um LPS, usando modelos de variabilidade baseados em UML e CVL, o modelo é usado para a criação de máquinas de estado são então convertidos em diagrama de classes.

Medicina (Hasling et al., 2008) realiza a conversão dos modelos em diagramas de atividades e caso de uso, após esta conversão é elaborada a criação de casos de teste, onde as variantes dos modelos podem ser consideradas como variáveis. Ao usar o TBM, ele não menciona se a variabilidade é manipulada, simplesmente usada para criar casos de teste a partir de diagramas de atividades.

Software O software é um dos domínios mais relatados, inicialmente porque está diretamente ligado ao tópico de pesquisa, embora os demais domínios também utilizem o TBM no processo de desenvolvimento de algum produto de software específico para um determinado domínio.

(Olimpiew, 2008b) também fornece muitos dados e informações sobre TBM em LPS, embora o foco seja em diagramas de atividades customizados, mas com foco na reutilização do teste em produtos de variante LPS. A abordagem proposta do CADeT transforma os casos de uso em diagramas de atividades, que por sua vez são transformados em casos de teste. As especificações de teste são rastreadas a partir do diagrama de atividades e do caso de uso.

A grande maioria dos estudos relata a criação de alguma abordagem ou ferramenta em TBM para auxiliar no desenvolvimento de LPS, no entanto, muitos não relatam com mais detalhes o uso de TBM em seu estudo.

Telecomunicação O TBM atua no auxílio da derivação de casos de teste de máquinas de estado, o desempenho neste domínio é devido ao uso de mídia embutida nos dispositivos de comunicação. Em um dado momento é possível interpretar que o estudo referente a esse domínio seria manufatura, mas tem uma área de atuação bem definida que seria a divisão de pesquisa de uma grande empresa de telecomunicações.

Ainda sobre o uso de ferramentas (Wang et al., 2013a) retrata o uso de um processo automatizado de seleção de teste de forma mínima e sistematizado. Processos automatizados contribuem para algoritmos de seleção, por mais que estejam em uma manufatura, para criar uma visão sistêmica da seleção de variabilidade.

A Tabela - 1.16 lista os estudos realizados para testar diferentes tipos de solução de LPS, como: *Teste de recurso*, *Teste de modelo LPS* e *Teste PLA* (arquitetura de linha de produto - PLA). Para essa classificação, separamos o PLA dos artefatos gerais do LPS, devido à importância direta do PLA para o sucesso dos LPSs.

Podemos observar na Tabela - 1.16 a maioria dos estudos voltados para o teste de PLA. Isso ocorre principalmente devido ao papel central do PLA para o desenvolvimento de LPSs, uma vez que é uma abstração de todos os potenciais produtos únicos a serem

Tabela 1.16: Teste de TBM de tipos de solução LPS

Tipo de Solução	ID do estudo	Estudos
Feature Testing	S1, S8, S11, S14, S15, S17, S18, S26, S27, S29, S32	11
LPS Model Testing	T2, S3, S4, S5, S9, S12, S13, S19, S20, S21, S24, S30, S36, S37, S39	15
PLA Testing	T6, S7, S10, S16, S22, S23, S25, S28, S31, S33, S34, S35, S38, S40, S41, S42, S43, S44	18
TOTAL		44

desenvolvidos por um LPS. Portanto, 18 estudos (40,9 %) testaram o PLA e seus modelos. Os testes de PLA estão no nível de dados de entrada e saída, por meio de testes de caixa preta e testes de funcionalidade. Ao buscar a reutilização, há uma preocupação com a confiabilidade da arquitetura principal e com a derivação de produtos LPS.

Com relação ao teste de modelos de LPS, encontramos 34 % dos estudos levando em consideração qualquer tipo de nível de teste ou tipo para LPSs para engenharia de domínio e engenharia de aplicação. Testar modelos de LPS é um meio de fornecer reutilização de casos de teste para produtos específicos derivados de um LPS. Como apontado na literatura geral de testes, a detecção de defeitos nos modelos pode ser várias vezes menos custosa do que nas atividades posteriores do NPS. O teste nos modelos de LPS depende muito de qual nível será testado. Na maioria desses estudos, os testes dependem de funcionalidades, portanto, os modelos estão sendo testados de acordo com o resultado esperado dos requisitos especificados.

O teste de recursos é composto por 25 % dos estudos. Embora os recursos estejam no nível do espaço do problema, seus testes são essenciais para o sucesso de um LPS, pois os recursos estão diretamente relacionados aos modelos de LPS em todos os níveis, incluindo o código-fonte. O teste de recurso funcional está de acordo com a cobertura das ações esperadas. Assim, este é um grande desafio, pois com a derivação de novos produtos suas funcionalidades podem sofrer variações, o que pode causar uma combinação exponencial de sequências de teste.

Por uma questão de contribuições de estudos, analisamos e apresentamos na Tabela - 1.17 em termos de seus tipos de propostas.

Podemos observar que a maioria dos estudos (33) apresenta como uma contribuição de proposta uma abordagem. Os estudos restantes relatam de uma a três propostas cada.

RQ.2:Quais abordagens de TBM, níveis de teste, artefatos, ferramentas e modelos foram usados para testar LPSs?

Nesta seção, fornecemos resultados sobre abordagens, níveis de teste, artefatos, ferramentas e modelos usados para aplicar o TBM às LPSs.

Tabela 1.17: Propostas principais de TBM para LPS

Tipo de proposta	ID do estudo	Estudos
Abordagem	T1, S5, S6, S7, S8, S9, S10, S11, S13, S14, S15, S16, S17, S18, S20, S21, S22, S23, S24, S27, S28, S29, S30, S31, S35, S36, S37, S38, S39, S41, S42, S43, S44	33
Algoritmo	S3, S25, S26	03
Kit de ferramentas	S40	01
Estratégia	S33	01
Abordagem de Extensão	S12	01
Ferramenta	S2, S4, S34	03
Metodologia	S19	01
Tutorial	S32	01
TOTAL		44

Os estudos analisados realizam principalmente testes de tempo de projeto. Além disso, para esta questão de pesquisa, analisamos os testes:abordagens, níveis e tipos e automação.

Abordagens Usadas A partir de estudos selecionados, criamos duas abordagens baseadas em "caixa":caixa branca e caixa preta. O primeiro é quando alguém tem acesso ao código-fonte e pode fornecer análises com maior profundidade. Este último é realizado quando não se tem acesso ao código, mas apenas para inserir dados e parâmetros, assim como os resultados produzidos.

Tabela - 1.18 apresenta os estudos classificados em cada uma das abordagens.

Tabela 1.18: Abordagens de TBM para LPS

Abordagem de Teste	ID do estudo	Estudos
Caixa-Preta	S2, S3, S7, S8, S9, S10, S11, S12, S13, S14, S15, S16, S17, S18, S19, S21, S22, S23, S24, S25, S26, S27, S28, S29, S30, S31, S32, S33, S34, S35, S36, S37, S38, S39, S40, S41, S42, S43, S44	39
Caixa-Branca	S1, S4, S20	03
Não Informado	S5, S6	02
TOTAL		44

Podemos observar que a maioria dos estudos (39/44) executa testes de caixa preta, principalmente porque:os testes são realizados em uma abstração de nível superior, como, por exemplo, testes funcionais; e LPS na atividade de engenharia de domínio.

A grande maioria dos estudos procura gerar casos de teste por meio de modelos, portanto, esse é um dos fatores que levam ao uso da abordagem da caixa preta. Steffens et al. (Steffens et al., 2012b) use modelos para gerar casos de teste. Esta prática é feita por quase todos os outros estudos (Cichos et al., 2011; García Gutiérrez et al., 2010; Lackner et al., 2014; Oster et al., 2011a; Samih et al., 2014b).

Níveis de teste Encontramos diferentes níveis e tipos de testes TBM. Da Lista Final, quatro estudos se destacam mais (Tabela - 1.19²), Que são: *Integração*, *Sistema*, *Regressão* e *Funcional*. A Figura - 1.8 descreve os níveis e tipos de TBM de LPSs.

Tabela 1.19: TBM dos níveis e tipos de testes de LPS

Nível / Tipo de Teste	ID do estudo	Estudos
Combinação	S4	01
Estrutural	S20	01
Funcional	S6, S9, S10, S22, S24, S28, S31	07
Performance	S20	01
Regressão	S1, S11, S13, S14, S15, S17, S18, S27, S30,	09
Sistema	S25, S26, S33, S34, S35, S36, S38, S39, S40, S41, S42, S43, S44	13
Integração	S3, S5, S7, S8, S11, S13, S14, S15, S16, S17, S18, S19, S21, S23, S29, S37	16
Unitário	S2	01
Não Informado	S12, S32	02
TOTAL		51

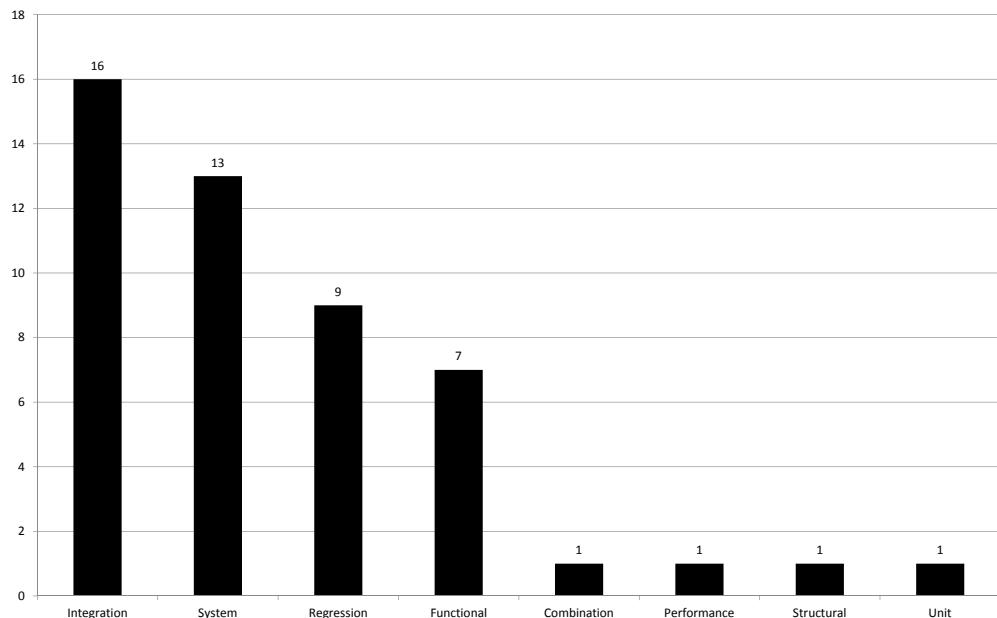


Figura 1.8: Comparação de níveis e tipos de TBM para LPS

Podemos observar que o nível de teste de integração é um dos mais considerados, pois a maioria dos testes é realizada na engenharia de domínio LPS. Além disso, para a reutilização de muitos casos de teste, a derivação de novos produtos requer uma verificação de integração entre modelos (Cai e Zeng, 2013; Cichos et al., 2011; Devroey, 2014; Devroey et al., 2014a; Dukaczewski et al., 2013; García Gutiérrez et al., 2010; Olimpiew, 2008a;

²Um estudo pode suportar mais de um nível ou tipo de teste.

Oster, 2012; Oster et al., 2011a; Samih e Baudry, 2012; Samih e Bogusch, 2014b; Samih et al., 2014b; Varshosaz et al., 2015; Wang et al., 2013a) .

Automação de Testes Ao analisar os estudos selecionados, identificamos o uso de ferramentas totalmente automatizadas e semi-automatizadas para auxiliar no teste de TBMs de LPSs. Também identificamos estudos sem auxílio de ferramentas, desenvolvendo manualmente atividades de testes, como podemos observar na Tabela - 1.20.

Tabela 1.20: TBM de automação LPS

Abordagem de automação	ID do estudo	Estudos
Totalmente automatizado	S2, S3, S4, S5, S6, S7, S8, S9, S10, S11, S12, S13, S14, S15, S16, S17, S18, S20, S22, S23, S24, S26, S27, S28, S31, S40, S41	27
Semi-automatizado	S19, S21, S25, S29, S30, S34, S36, S37, S38, S39, S42, S43, S44	13
Manual	S1, S33, S35	03
Não Informado	S32	01
TOTAL		44

Nãos estudos selecionados, verificamos o uso de ferramentas *fully-automated* em atividades de teste. Segundo estudos selecionados, a maioria deles utiliza algum tipo de automação para realizar tais atividades. As ferramentas usadas são amplamente implementadas pelos autores dos estudos (Devroey et al., 2015; Oster et al., 2011b; Perrouin et al., 2010; Steffens et al., 2012a), mas esses autores também fornecem comparações entre ferramentas de abordagens semelhantes (Devroey et al., 2015; Oster et al., 2011b; Perrouin et al., 2010; Rodrigues, 2013; Steffens et al., 2012a).

Outro conjunto de estudos selecionados usa ferramentas *semi-automated*, nas quais apenas um trecho do processo de teste é automatizado, com ou sem atividades manuais. O Olimpiew (Olimpiew, 2008a) cria casos de uso para permitir especificações de casos de teste reutilizáveis pelo arquiteto de software. De acordo com Weissleder e Lackner (Weißleder e Lackner, 2013b), os modelos são usados em conjunto para modelar recursos e comportamento para a geração de diagramas de máquinas de estado e comportamento por engenheiros de software usando abordagens semi-automatizadas.

Estudos puramente manuais não adotam nenhum tipo de automação das atividades de teste. Tais estudos discutem abordagens teóricas relacionadas com TBM de LPS e, portanto, foram classificados como *Manual*. Em geral, as atividades manuais seriam a conversão de um modelo inicial em um modelo secundário com maior detalhe, depois para casos de teste (Beohar e Mousavi, 2014b; Lity et al., 2016; Lochau e Kamischke, 2012). As abordagens manuais tendem a apresentar apenas conceitos teóricos do modelo de processo desenvolvido, ou buscam apresentar a essência referente a um contexto específico.

O gráfico de bolha da Figura - 1.9 mostra a automação de abordagens. Como podemos observar, o teste caixa-preta tem maior automação, tanto por ferramentas totalmente automatizadas quanto semi-automatizadas, totalizando 39 estudos. As abordagens de caixa branca são menos executadas com o auxílio de ferramentas (semi) automatizadas, apenas três estudos.

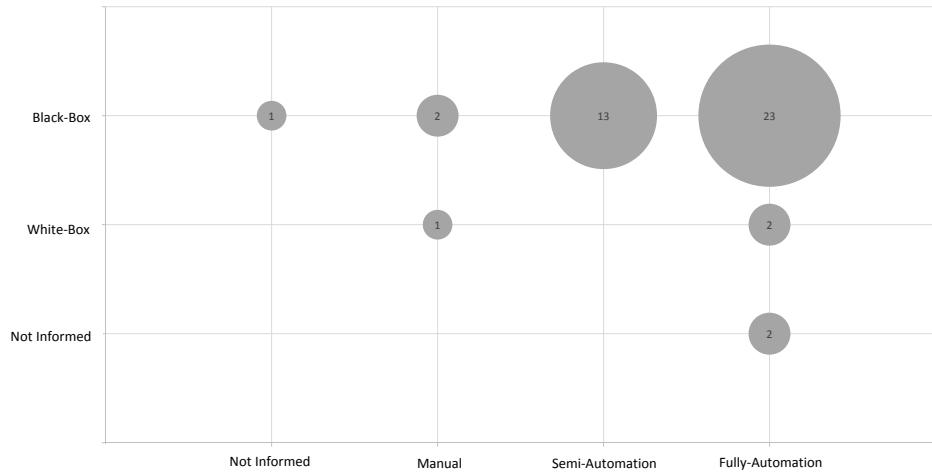


Figura 1.9: Automação TBM para LPS de abordagens de teste

O gráfico de bolha da Figura - 1.10 mostra a automação dos níveis e tipos de teste.

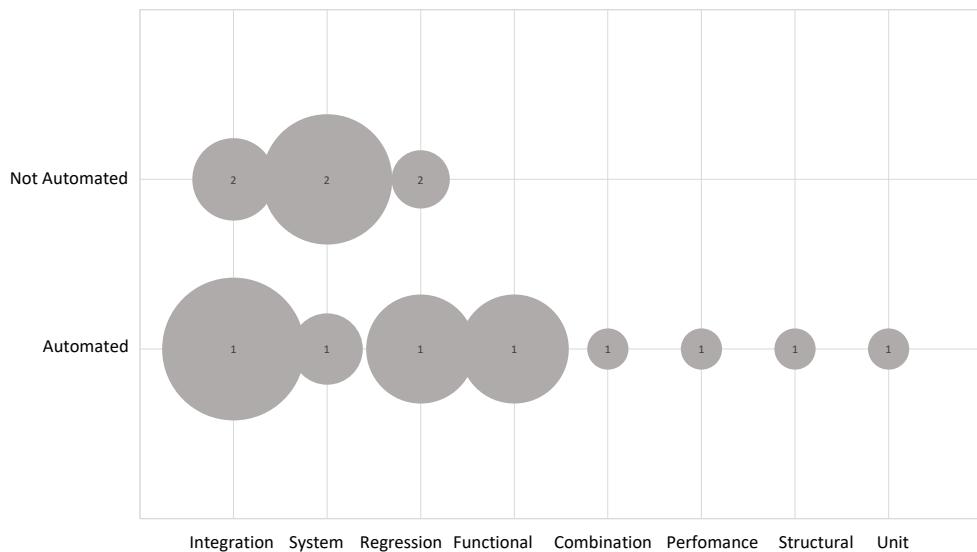


Figura 1.10: Automação de níveis de teste de TBM para LPS

Treze estudos são menos realizados com o auxílio de ferramentas (semi) automatizadas. Apenas três estudos não apresentam nenhum tipo de automação, enquanto um estudo não relata.

Artefatos Usados Para gerar casos de teste, os estudos de TBM usam vários artefatos diferentes, como Máquina de estados, Diagrama de classes, Diagrama de sequência e Modelo de recursos, como podemos observar na Tabela - 1.21. Uma comparação de tais artefatos é fornecida na Figura - 1.11.

Tabela 1.21: Artefatos Usados Durante TBM de LPS

Artefato	ID do estudo	Estudos
Máquina de Estado	S1, S6, S7, S10, S11, S12, S13, S14, S15, S17, S21, T22, S23, S27, S29, S30, S34, S38, S44	19
Modelo original	S8, S9, S42	03
OVM Variability Model	S8	01
Diagrama de Classes	S6, S12, S14, S31	04
Diagrama de atividades	S18, S19, S41	03
Diagrama de casos de uso	S20, S28, S41, S43	04
Diagrama de sequência	S24, S31, S36	03
Diagrama de Transição	S7, S15, S25, S26, S33, S35, S37	07
Diagrama de Funções	S2, S3, S4, S5, S9, S16, S39, S40	08
Não Informado	S32	01
TOTAL		53

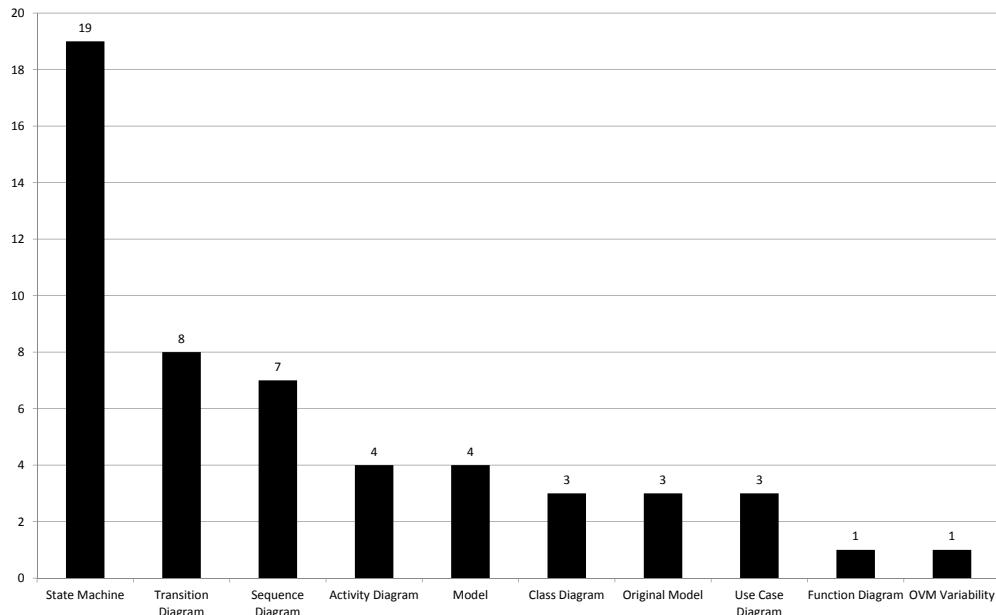


Figura 1.11: Automação TBM para LPS de abordagens de teste

Alguns desses artefatos desempenham o papel do artefato original (inicial) em teste, que são os artefatos dos quais os casos de teste devem ser gerados. Outros artefatos são usados como elementos intermediários para fornecer ao TBM um meio de testar um determinado modelo original.

Ferramentas usadas Figura - 1.12 apresenta ferramentas usadas para executar TBMs de LPSs relacionadas a artefatos. Algumas dessas ferramentas são mais frequentes, pois são usadas em mais de um estudo.

Podemos citar algumas ferramentas com um número maior de citações, como a série Rational IBM, o gerenciador de linha de produtos MaTeLo, variantes PURE, ferramentas CADeT e Eclipse (Beohar e Mousavi, 2014b; Costa L. T., 2016; Lity et al., 2012; Olimpiew, 2008b; Samih e Bogusch, 2014a).

As ferramentas CADeT (Olimpiew, 2008a) ajudam na geração de casos de teste a partir de modelos, bem como na especificação de casos de teste reutilizáveis. Fazendo uso do Delta para extrair os casos de teste com foco em regressão, esta ferramenta, bem como as ferramentas IBM Rational, Eclipse (Lochau et al., 2012b) e MoSo-PoLiTe foram consideradas para resolver o mesmo problema.

Tabela 1.22: Comparação de ferramentas e artefatos primários ou secundários

Modelos LPS Quanto ao tipo de modelo de LPS considerado entre os trabalhos analisados, a maioria foi feita usando *Behavioral-based*, que se caracteriza por ser comportamental, talvez a razão para isso, seja porque os modelos são ricos em detalhes, facilitando interpretação ou configuração, mesmo se o modelo é posteriormente convertido em um artefato, primário ou intermediário. Na Tabela - 1.23 pode-se observar que vários trabalhos utilizam o modelo de modelos baseados em Cenário, a relação se deve ao fato de estes trabalhos fazerem uso de artefatos intermediários, trabalhando com cenários de uso ao invés de comportamento direto.

Tabela 1.23: TBM de LPS Tipos de Modelos

Modelo LPS Considerado	Referência	Estudos
Scenario-based	S3, S11, S19, S20, S24, S31, S38, S40, S41, S42, S43, S44	12
Class-oriented	S6	01
Behavioral-based	S1, S2, S4, S7, S8, S9, S10, S12, S13, S14, S15, S16, S17, S18, S21, S22, S23, S25, S26, S27, S28, S29, S30, S33, S34, S35, S36, S37, S39	29
Não Identificado	S5, S32	02
TOTAL		44



Figura 1.12: Artefatos X Níveis X Modelos

RQ.3:Como a variabilidade e o tempo de ligação são tratados durante o TBM de LPSs?

Como a variabilidade é a essência da LPS, assim como é diretamente relacionada ao tempo de vinculação, nesta seção apresentamos resultados nos quais os estudos levam em conta tais conceitos e como eles os tratam.

Tabela - 1.24 classifica os estudos que:consideram a variabilidade (linha “ *Sim* ”) durante o TBM do LPS e não consideram a variabilidade (linha “ *Não* ”). Em vários estudos, não conseguimos identificar se as variabilidades são tratadas.

Tabela 1.24: Estudos que tratam da variabilidade durante as atividades da TBM

Trate a variabilidade?	ID do Estudo	Estudos
Sim	S1, S2, S4, S5, S6, S7, S8, S9, S10, S12, S13, S14, S15, S16, S17, S18, S19, S21, S22, S24, S25, S27, S28, S29, S30, S31, S33, S34, S37, S40	30
Não	S39, S43	02
Não Identificado	S3, S11, S20, S23, S26, S32, S35, S36, S38, S41, S42, S44	12
TOTAL		44

Apenas dois estudos (S39, S43) não tratam a variabilidade durante atividades de TBM. Isso ocorre porque eles aplicam testes em produtos específicos de LPS na engenharia de aplicativos.

Como podemos observar na Tabela - 1.24, 68,1 % (30/44) dos estudos tratam a variabilidade durante qualquer atividade de TBM de LPSs.

Muitas estratégias diferentes são usadas para lidar com a variabilidade durante o TBM. A variabilidade pode ser modelada e resolvida em diagramas UML, com a geração de diagramas de classes e máquinas de estados (Wang et al., 2013a) a serem testadas.

Idiomas específicos também podem ser usados para especificar a variabilidade dos casos de teste (Devroey, 2014). A Figura - 1.13 resume quais estudos tratam ou não a variabilidade por tipo de solução.

O tempo de ligação é a capacidade de resolver a variabilidade em qualquer ponto do ciclo de vida do LPS. É obrigatório o aumento significativo dos níveis de variabilidade (Chen et al., 2009b).

O tempo de vinculação no LPS é tradicionalmente em tempo de design, pré-compilação, compilação ou tempo de vinculação, lidando com a variabilidade estática. Já nos sistemas que lidam com a variabilidade dinâmica, pode ser em tempo de carregamento quando um sistema é implementado e carregado na memória e, em tempo de execução, após o sistema ter iniciado a execução de (Alves et al., 2009).

Tabela - 1.25 lista estudos dos quais 86,3 % (38/44) lidam com a variabilidade de vinculação em tempo de design. Os estudos restantes não informam seus tempos de ligação.

Tabela 1.25: Tempo de Ligação de Variabilidade em TBM de LPS

Binding Time	ID do Estudo	Estudos
Design Time	S7, S8, S9, S10, S11, S12, S13, S14, S15, S16, S17, S18, S19, S20, S21, S22, S23, S24, S25, S26, S27, S28, S29, S30, S31, S32, S33, S34, S35, S36, S37, S38, S39, S40, S41, S42, S43, S44	38
Não Identificado	S1, S2, S3, S4, S5, S6	06
TOTAL		44

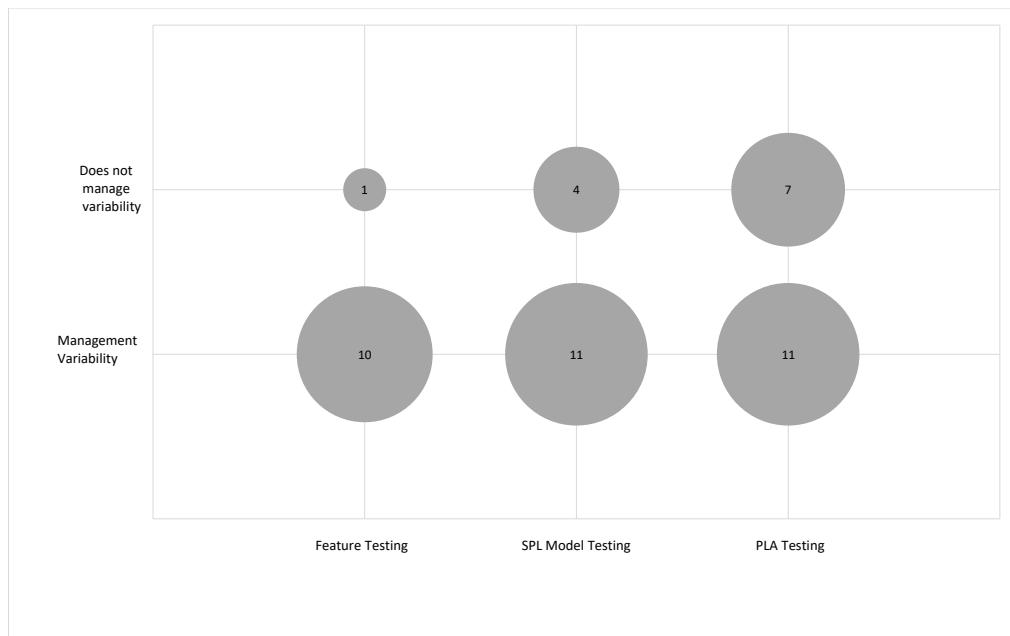


Figura 1.13: Tratamento da variabilidade por tipo de solução

RQ.4:O TBM suporta testes de requisitos não funcionais do LPS?

Nenhum dos 44 estudos selecionados deste SMS apresenta ou menciona qualquer tipo de teste de requisitos de LPS não funcional. Embora o LPS e os produtos testados estejam geralmente em tempo de design, nada impede a criação dos requisitos não funcionais do plano de teste (Lee et al., 2012).

Feng et al. (Feng et al., 2007) e Reis et al. (Reis et al., 2007), por exemplo, propõe a criação de casos de teste para a variabilidade considerando requisitos não-funcionais. Kakarontzas et al. (Kakarontzas et al., 2008) propõe casos de teste hierárquicos para requisitos não funcionais específicos.

RQ.5:Como as propostas de TBM de LPS são avaliadas?

Analisamos a avaliação da proposta de estudos de TBM de LPS com base no tipo de avaliação e no ambiente em que as avaliações são realizadas.

Tabela - 1.26 lista estudos de acordo com seus tipos de avaliação, que são:*Experiment* e *Case Study*. Dos estudos selecionados, 79,5 % (35/44) deles realizaram um desses tipos

de avaliação. Em nove estudos, nem a avaliação foi realizada nem eles puderam ser identificados.

Tabela 1.26: Método de Evidenciação da Solução TBM para LPS

Método de avaliação	ID do Estudo	Estudos
Experimento	S1, S3, S4, S16, S20, S21, S23, S24, S25, S28, S39, S40, S42	13
Estudo de Caso	S2, S6, S7, S8, S9, S10, S11, S12, S13, S14, S15, S17, S19, S22, S26, S27, S29, S30, S33, S38, S43, S44	22
Não Informado	S5, S18, S31, S32, S34, S35, S36, S37, S41	09
TOTAL		44

Os experimentos são responsáveis por 29,5 % (13/44) das avaliações realizadas. Nós não percebemos qualquer replicação experimental como um meio de impor resultados originais em uma determinada proposta. Além disso, poucos trabalhos apresentam elementos experimentais essenciais, como validade do estudo, perfil dos participantes, formulação e teste de hipóteses. Embora a maioria dos estudos relacionados à indústria não use participantes humanos, o tamanho da amostra não é relatado na maioria dos estudos. Cerca de 60 % dos estudos disponibilizam seu pacote experimental ou parte dele, incluindo, por exemplo, diagramas e algoritmos, por meio de uma URL pública. Não entanto, a maioria das URLs não está mais disponível . Acreditamos que isso ocorre porque a maioria das avaliações é realizada em conjuntos de setores (consulte Tabela - 1.27).

Os estudos de caso são o tipo de avaliação mais realizado das propostas de TBM para LPS, com 50 % (22/44) dos estudos e podem ter questões confidenciais.

Um aspecto importante relacionado à avaliação de propostas é o ambiente em que tais avaliações ocorrem. Portanto, analisamos quais estudos são realizados na academia e quais são realizados na indústria. Tabela - 1.27 lista estudos por ambiente de avaliação.

Tabela 1.27: Ambientes de evidência de soluções TBM para LPS

Environment	Study ID	Count
Industria	S1, S2, S3, S4, S5, S6, S7, S8, S9, S10, S11, S12, S13, S14, S15, S16, S17, S20, S22, S29, S30, S38, S41, S42, S43, S44	26
Academia	S19, S23, S24, S25, S26, S27, S28, S31, S32, S33, S34, S35, S36, S37, S39, S40	16
Não Informado	S18, S21	02
TOTAL		44

A maioria dos estudos ocorre na indústria, 59 % (26/44), o que é de grande importância devido ao seu realismo e participação de profissionais de testes reais. Além disso, o conjunto da indústria fornece dados reais de projetos de LPS. Por outro lado, em estudos realizados em ambiente acadêmico, geralmente recruta os estudantes como participantes e

usam NPSs pedagógicos ou não comerciais. Nãoote que o uso de estudantes na engenharia de software baseada em evidências não é uma ameaça tão amplamente discutida (Feldt et al., 2018).

A Figura - 1.14 mostra um gráfico de bolhas dos tipos de avaliação e seus ambientes.

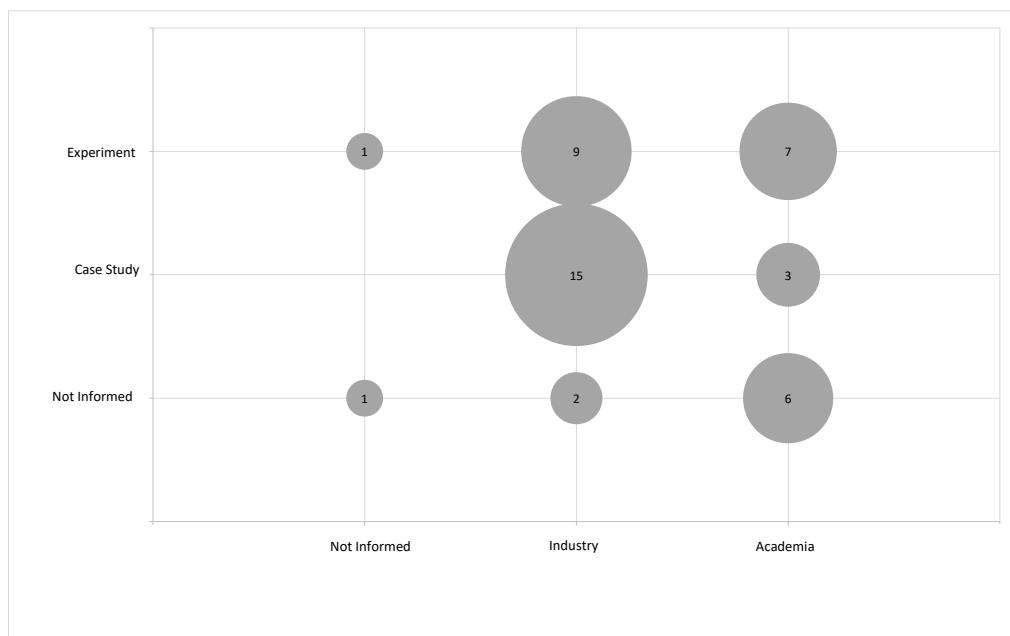


Figura 1.14: Tipos de Avaliação de Propostas e Ambientes

Como podemos observar na Figura - 1.14, a maioria dos estudos, nove experimentos e 15 estudos de caso, foram avaliados no conjunto da indústria, o que contribui para aumentar a confiabilidade das evidências das propostas fornecidas.

RQ.6:Como a rastreabilidade é considerada durante atividades de TBM para LPSs?

A rastreabilidade ainda é uma das muitas lacunas de pesquisa no desenvolvimento de software, especialmente para LPSs. Como a rastreabilidade é responsável por identificar elementos importantes para derivar produtos específicos de LPS e para a evolução do LPS, é diretamente necessário durante as atividades do TBM (Bernardino et al., 2017; Costa L. T., 2016). Além disso, a rastreabilidade fornece suporte essencial para garantia de qualidade de LPSs.

Nesta seção, analisamos quais estudos levam em consideração qualquer tipo de rastreabilidade durante as atividades de TBM. Assim, Tabela - 1.28 lista os estudos e se fornecem algum suporte para a rastreabilidade.

Tabela 1.28: Estudos com Suporte de Rastreabilidade para TBM de LPSSs

Suporte a Rastreabilidade?	ID do estudo	Estudos
Sim	S5, S7, S15, S16, S17, S19, S20, S24, S31, S36, S37, S41	12
Não	S1, S2, S3, S4, S6, S8, S9, S10, S11, S12, S13, S14, S18, S21, S22, S23, S25, S26, S27, S28, S29, S30, S32, S33, S34, S35, S38, S39, S40, S42, S43, S44	32
TOTAL		44

Poucos estudos (12/44) relatam ou mencionam a rastreabilidade como suporte. Além disso, nenhum deles declara como eles usam ou tratam a rastreabilidade. Eles só alegam que a rastreabilidade é fundamental para a geração de casos de teste, que devem ser o mais automatizados possível.

A.3.3 Procedimentos de compartilhamento de dados do SMS

Realizamos procedimentos para promover a abertura e a reprodutibilidade deste estudo. Portanto, nós:

- fornece um formato de arquivo CSV com dados de todos os estudos deste SMS;
- fornece um formato de arquivo de metadados CSV explicando o arquivo CSV dos estudos;
- fornece arquivos PDF para todas as figuras e gráficos deste documento;
- fornece o formato de arquivo TEX para todas as tabelas deste documento;
- fornece formato de arquivo BIBTEX e CSV com referências dos estudos da Lista Final para uso em diferentes ferramentas bibliográficas;
- Fornece item de dados (CSV, PDF, TEX, arquivos BibTeX) em um repositório confiável compartilhamento de dados público aberto e permanente, chamado Zenodo³, sob DOI xxxx/aaaa/zzzz.

³zenodo.org

A.4 TBM4LPS:um roteiro para testes baseados em modelos de pesquisa de LPSs

Esta seção apresenta um roteiro, denominado TBM4LPS, como resultado do mapeamento da Lista Final de estudos com relação às fases do processo de TBM para LPSs. Três fases são consideradas:Baseado em artefato, Gerenciamento e Verificação de Variabilidade e Execução de TBM.

A Figura - 1.15 mostra o TBM4LPS sobre como os estudos são divididos de acordo com as fases e atividades do TBM. Para ilustrar o readomap, duas rotas serão apresentadas:a primeira baseada principalmente em um estudo primário específico, guia os usuários do modelo de estudo até a geração de casos de teste; e o segundo, considerando as três fases e atividades do TBM para LPS, dando aos usuários estudos que suportam tais fases.

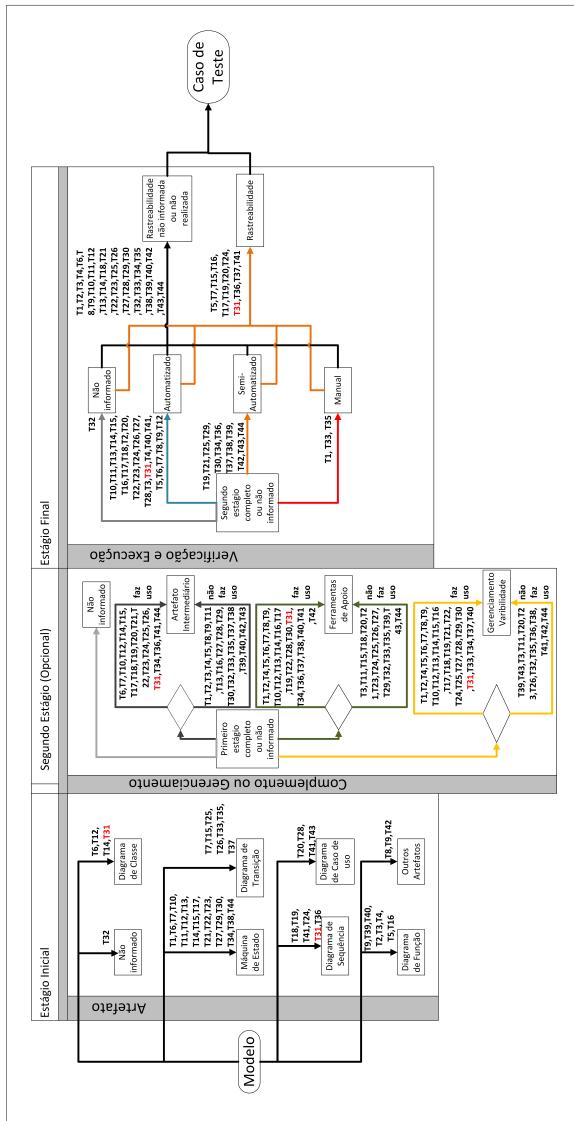


Figura 1.15: TBM4LPS: um roteiro de TBM para LPSs

As seções a seguir apresentam exemplos de aplicativos sobre como usar o TBM4LPS.

A.4.1 Rota Primária Baseada no Estudo

Digamos que, ao selecionar um estudo de Tabela - 1.12 do seu ID, você queira verificar o Figura - 1.15 qual caminho percorre, identificando seus estágios e atividades. Assim, inicialmente analisamos o primeiro estágio representado em Figura - 1.16, que é um excerto de Figura - 1.15.

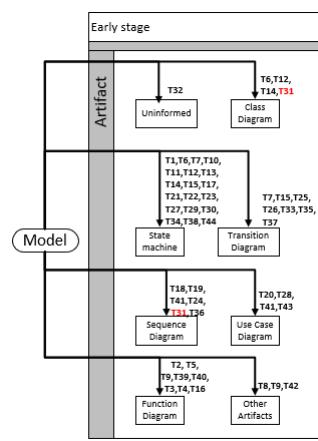


Figura 1.16: Etapa inicial, mostrando a localização do estudo T31

Tomemos, por exemplo, o estudo T31. A partir de um modelo, o T31 faz uma conversão para o artefato do diagrama de sequência, então podemos considerar o T31. Com o estágio inicial concluído, o segundo estágio (opcional) é iniciado e o T31 considera o gerenciamento de variabilidade, suporte de ferramentas e geração de artefatos intermediários, pois nesse estágio abordamos o gerenciamento de variabilidade, uso de ferramentas, geração de artefatos intermediários e se não executar qualquer uma dessas atividades automaticamente, pode ser na geração do caso de teste.

Ao apresentar o estágio final em Figura - 1.18 pode-se analisar que o estudo T31 faz uso de processo automatizado e menciona a rastreabilidade. Isso conclui o curso do guia para o estudo T31, identificando, assim, quais etapas o estudo passou.

A.4.2 Rota baseada em critérios

O segundo cenário executa o tema TBM no LPS por assunto. Ao analisar o Figura - 1.18, por exemplo, pode-se verificar quais trabalhos fazem uso da rastreabilidade. Portanto, se

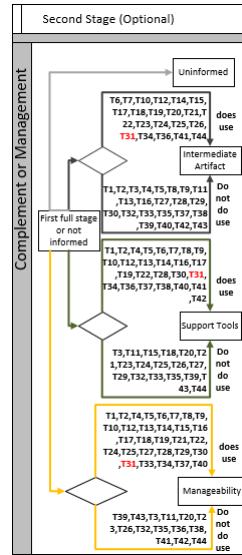


Figura 1.17: Segunda etapa, mostrando a localização do estudo T31

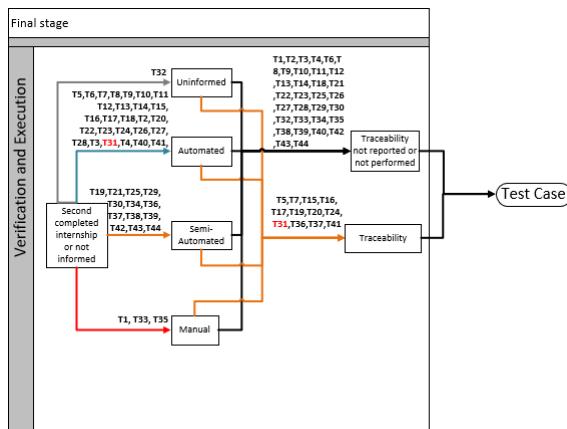


Figura 1.18: Etapa final mostrando a localização do estudo T31

o interesse do leitor é por obras que fazem uso da rastreabilidade, basta consultar essa atividade para obter a lista de tais obras.

A.5 Observações Conclusivas

Este mapeamento sistemático foi realizado para obter uma visão geral sobre testes baseados em modelo para linhas de software, e identificamos quarenta e quatro trabalhos publicados entre 2006 e 2016. Há um desafio muito grande ao abordar o teste em LPS, onde os principais pontos levantados são o gerenciamento de variabilidade, a cobertura de erros, a automação de processos e a rastreabilidade.

O objetivo principal da pesquisa foi verificar se o TBM é aplicado no NPS e, quando aplicado, se o gerenciamento de variabilidade foi utilizado. Após a análise dos dados extraídos do SMS, foi possível verificar e identificar as técnicas de teste de TBM utilizadas, que utilizam a notação UML, os trabalhos que realizam um controle de variabilidade, bem como a menção de rastreabilidade. Além disso, como os casos de teste são gerados, manualmente ou automatizados.

As contribuições para o TBM em LPS não são tão detalhadas quanto esperávamos, o que exigiu um trabalho extra na interpretação dos dados extraídos. Os principais dados que o SMS apresenta como contribuição são os artefatos utilizados, as técnicas relatadas, o uso de ferramentas e o uso do gerenciamento de variabilidade.

Com esse SMS, podemos ver que o domínio de maior desempenho é o software, a grande maioria considera o uso da variabilidade e a maior parte das validações estão sendo realizadas no setor, isso mostra que há um crescente interesse no assunto, embora não podemos identificar por que a rastreabilidade e os testes de requisitos não funcionais não são explicitamente apresentados, estudos adicionais sobre esses dois temas podem nos dizer mais sobre isso.

Um dos principais pontos que é a gestão da variabilidade atrelada ao modelo em LPS foi apresentado por uma maioria significativa, isso demonstra que há uma crescente conscientização da qualidade dos produtos variantes oriundos de um produto principal.

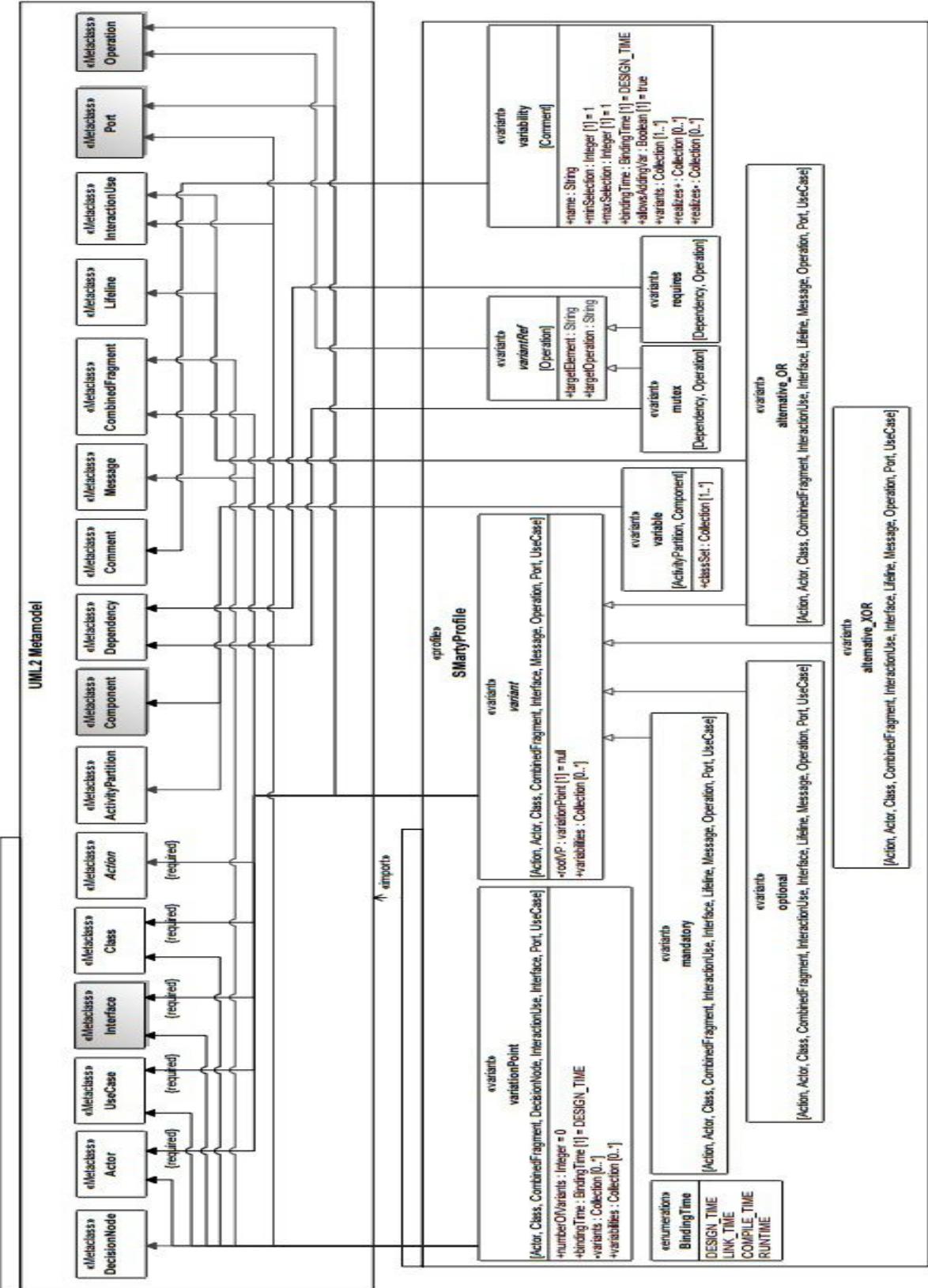
Devido a esse crescente interesse, incentivamos a comunidade a continuar com novas pesquisas para testes baseados em modelo, a fim de apresentar dados sobre rastreabilidade, automação de processos, cobertura de erros e gerenciamento de variabilidade. Para estes parecem precisar de mais compreensão e avaliação.

Anexo: Abordagem SMarty

A abordagem *SMarty* (Bera et al., 2015); (Geraldi et al., 2015); (Marcolino et al., 2014a); (Marcolino et al., 2014b); (Marcolino et al., 2013); (OliveiraJr et al., 2010b) possibilita o gerenciamento de variabilidades em LPSs modeladas em *Unified Modeling Language* (UML), a partir de um conjunto de estereótipos e diretrizes, que aplicadas aos elementos dos modelos, permitem a representação das variabilidades.

Os estereótipos fornecidos por *SMarty* estão organizados em um perfil UML denominado *SMartyProfile* (apresentado abaixo) e em um conjunto de diretrizes para auxiliar na identificação, delimitação e representação das variabilidades, com base em tais estereótipos, denominado *SMartyProcess*.

Figura 2.1: Versão 5.2 do *SMartyProfile*, com suporte a Componentes, Portas, Interfaces e Operações (Bera, 2015).



O *SMartyProfile* é uma extensão dos metamodelos da UML, versão 2.5. É possível perceber todos os estereótipos suportados pelo perfil na região inferior da Figura - 2.1. Os estereótipos representam conceitos característicos de LPS, como variabilidades, pontos de variação e variantes nos modelos UML. A seguir, são apresentados os estereótipos do *SMartyProfile*:

<< variability >> estereótipo de variabilidade. Esse estereótipo é uma extensão da metaclasse UML *Comment*, para representar explicitamente as variabilidades nos modelos UML. Tal estereótipo apresenta os seguintes meta-atributos;

- ***name***: nome utilizado para referenciar uma variabilidade;
- ***minSelection***: corresponde ao número mínimo de variantes selecionadas para resolver um ponto de variação e/ou uma variabilidade;
- ***maxSelection***: corresponde ao número máximo de variantes selecionadas para resolver um ponto de variação e/ou uma variabilidade;
- ***bindingTime***: corresponde ao momento de resolução da variabilidade. Os possíveis momentos de resolução são representados pela classe de enumeração *BindingTime*;
- ***allowsAddingVar***: indica se novas variantes podem ser incluídas após a resolução de uma variabilidade;
- ***variants***: coleção de instâncias, associadas à variabilidade; e
- ***realizes***: coleção de variabilidades de modelos de menor nível que realiza a variabilidade.

<< variationPoint >> estereótipo de ponto de variação. Esse estereótipo estende as metaclasses *Actor*, *Class*, *CombinedFragment*, *DecisionNode*, *InteractionUse*, *Interface*, *Port*, *UseCase* e possui os seguintes meta-atributos:

- ***numberofVariants***: número de variantes associadas ao ponto de variação;
- ***bindingTime***: estabelece o tempo de resolução do ponto de variação. Os possíveis tempos de resolução são determinados pela classe de enumeração *BindingTime* do Figura - 2.1;
- ***variants***: coleção de instâncias de variantes associadas ao ponto de variação; e
- ***variabilities***: coleção de variabilidades associadas com este ponto de variação.

`<< variant >>` estereótipo de variante. Tal estereótipo é especializado em outros quatro estereótipos (`<< mandatory >>`, `<< optional >>`, `<< alternative_XOR >>` e `<< alternative_OR >>`), estende as metaclasses *Action*, *Actor*, *Class*, *CombinedFragment*, *Interface*, *Message*, *Operation*, *Port*, *UseCase* e possui os seguintes meta-atributos:

- ***rootVP***: representa o ponto de variação ao qual a variante está associada; e
- ***variabilities***: coleção de variabilidades ao qual a variante está associada.

`<< mandatory >>` estereótipo de variante obrigatória. Isso indica que essa variante sempre deve estar na resolução de um ponto de variação e/ou variabilidade associado(a). As seguintes metaclasses são estendidas por esse estereótipo: *Action*, *Actor*, *Class*, *CombinedFragment*, *InteractionUse*, *Interface*, *Lifeline*, *Message*, *Operation*, *Port*, *UseCase*;

`<< optional >>` estereótipo opcional, na resolução de um ponto de variação e/ou variabilidade associado(a). Tal estereótipo é uma extensão das seguintes metaclasses: *Action*, *Actor*, *Class*, *CombinedFragment*, *InteractionUse*, *Interface*, *Lifeline*, *Message*, *Operation*, *Port* e *UseCase*;

`<< alternative_XOR >>` estereótipo que indica a existência de um grupo de variantes exclusivas, do qual a variante marcada com tal estereótipo faz parte. Isso significa que apenas uma variante desse grupo pode ser selecionada para a resolução de um ponto de variação e/ou variabilidade associado(a). Esse estereótipo é uma extensão das metaclasses *Action*, *Actor*, *Class*, *CombinedFragment*, *InteractionUse*, *Interface*, *Lifeline*, *Message*, *Operation*, *Port* e *UseCase*;

`<< alternative_OR >>` estereótipo que indica que a variante pertence a um grupo de variantes inclusivas. Isso significa que diferentes combinações de variantes inclusivas podem ser selecionadas para a resolução de um ponto de variação e/ou variabilidade associado(a). Esse estereótipo é uma extensão das metaclasses *Action*, *Actor*, *Class*, *CombinedFragment*, *InteractionUse*, *Interface*, *Lifeline*, *Message*, *Operation*, *Port* e *UseCase*;

`<< mutex >>` estereótipo que representa o relacionamento mutuamente exclusivo entre variantes. Isso significa que a escolha de uma variante desse relacionamento exige a não-escolha da outra variante do relacionamento. Tal estereótipo é uma extensão das metaclasses *Dependency* e *Operation*;

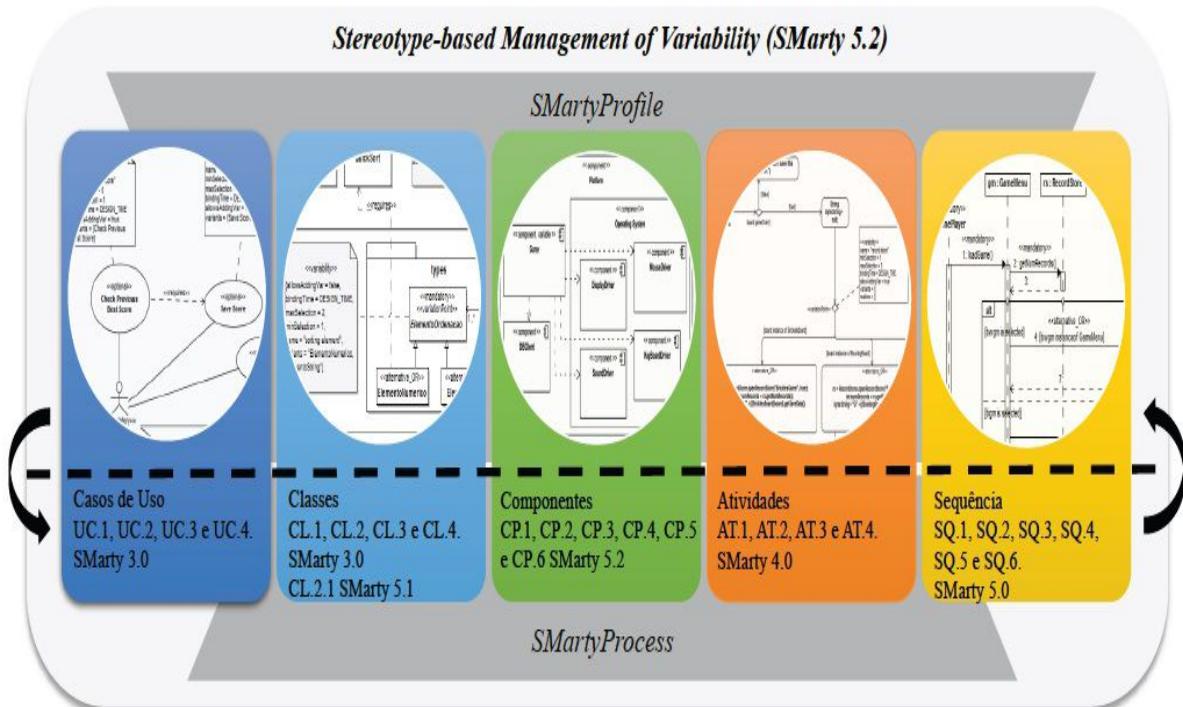
`<< requires >>` estereótipo que representa um relacionamento de complemento entre duas variantes. Isso significa que a escolha de uma variante desse relacionamento requer a seleção da outra variante relacionada. As metaclasses *Dependency* e *Operation* são estendidas por `<< requires >>`;

<< **variable** >> estereótipo extensão das metaclasses *ActivityPartition* e *Component*, que indica a existência de classes com variabilidades explícitas em um componente. O atributo *classSet* representa a coleção de instâncias das classes variáveis existentes no componente.

Os estereótipos do *SMartyProfile* permitem a representação explícita dos conceitos de LPS e possibilitam que o processamento automatizado de modelos UML, realizado por ferramentas de modelagem UML, também considere as características de LPS (?).

A Figura - 2.2 exibe uma visão geral da abordagem SMarty.

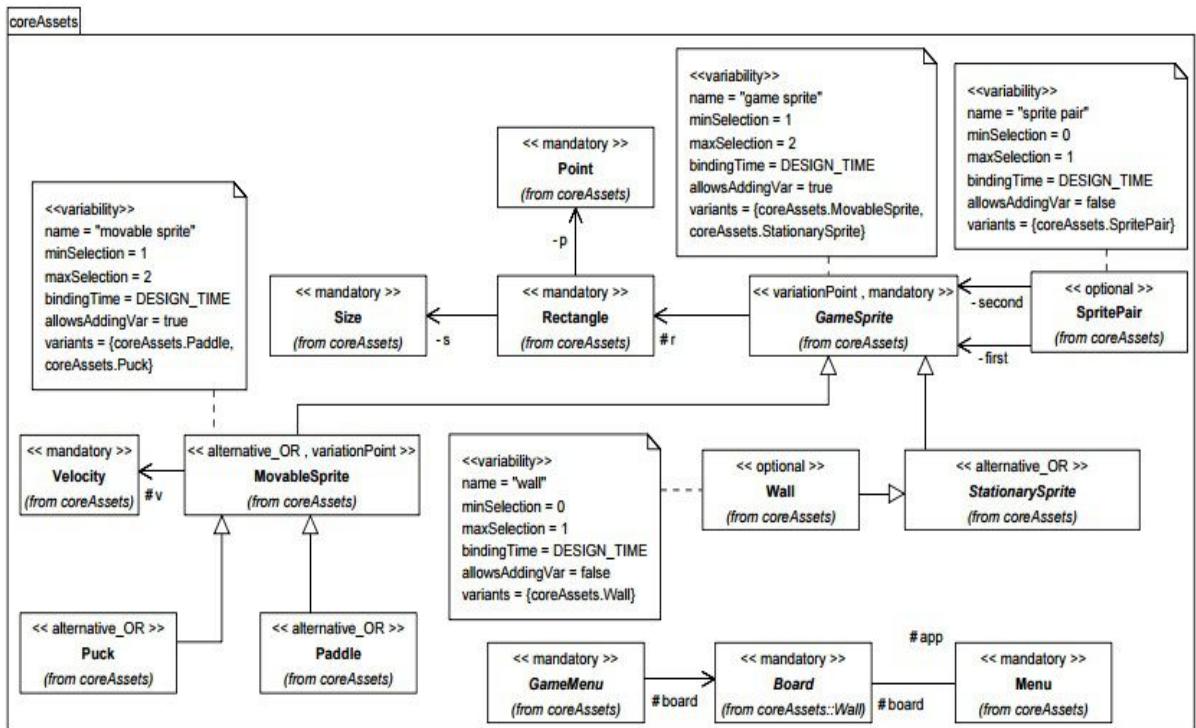
Figura 2.2: Visão Geral SMarty 5.2 (Bera, 2015).



A Figura - 2.2 possibilita visualizar os modelos UML para o qual a abordagem *SMarty* oferece suporte. Para cada modelo suportado, tem-se os estereótipos, definidos pelo *SMartyProfile* e um conjunto de diretrizes específicas para cada diagrama. As diretrizes são apresentadas por diferentes siglas, representando seus respectivos diagramas. UC representa as diretrizes para o diagrama de casos de uso, CL representa as diretrizes para o diagrama de classes, CP representa as diretrizes para o diagrama de componentes, AT representa as diretrizes para o diagrama de atividades e SQ representa as diretrizes para o diagrama de sequência. Cada conjunto de diretrizes foi especificado em uma versão do *SMarty*. Atualmente, o *SMarty* está na versão 5.2.

A Figura - 2.3 ilustra a aplicação dos estereótipos do *SMartyProfile* em um diagrama de classes da LPS *Arcade Game Maker* (AGM) .

Figura 2.3: Diagrama de Classes da LPS AGM (OliveiraJr et al., 2010b).



A relação entre variabilidades, pontos de variação e variantes pode ser percebida claramente na Figura - 2.3. Considerando a classe *GameSprite* presente na figura, percebe-se os estereótipos *<< variationPoint >>* e *<< mandatory >>*, indicando que a classe é um ponto de variação e simultaneamente uma variante obrigatória. Por ser um ponto de variação, tal classe está associada com uma variabilidade, no caso o comentário *game sprite*, estereotipado com *<< variability >>*. Duas variantes estão associadas ao ponto de variação, *StationarySprite* e *MovableSprite*, ambas estereotipadas com *<< alternative_OR >>*. Outros estereótipos também são observados, tais como o *<< optional >>*, na classe *SpritePair* e o *<< alternative_OR >>*, nas classes *Puck* e *Paddle*. Na geração de um produto por exemplo, o arquiteto de LPS pode escolher se o produto conterá elementos estáticos (*StationarySprite*), móveis (*MovableSprite*) ou mesmo ambos no jogo gerado.