UNIVERSIDADE FEDERAL DA BAHIA
INSTITUTO DE MATEMÁTICA

Programa de Pós-Graduação em Ciência da Computação

PLAR – UMA TÉCNICA PARA RECUPERAÇÃO DE ARQUITETURA DE LINHAS DE PRODUTO DE SOFTWARE

Mateus Passos Soares Cardoso

DISSERTAÇÃO DE MESTRADO

Salvador
14 de março de 2017

MATEUS PASSOS SOARES CARDOSO

PLAR – UMA TÉCNICA PARA RECUPERAÇÃO DE ARQUITETURA DE LINHAS DE PRODUTO DE SOFTWARE

Esta Dissertação de Mestrado foi apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal da Bahia, como requisito parcial para obtenção do grau de Mestre em Ciência da Computação.

Orientadora: Christina von Flach Garcia Chavez

Salvador
14 de março de 2017

Sistema de Bibliotecas - UFBA

Cardoso, Mateus Passos Soares.

PLAR – Uma Técnica para Recuperação de Arquitetura de Linhas de Produto de Software / Mateus Passos Soares Cardoso – Salvador, 2017.

111p.: il.

Orientadora: Prof. Dr. Christina von Flach Garcia Chavez.

Dissertação (Mestrado) – UNIVERSIDADE FEDERAL DA BAHIA, INSTITUTO DE MATEMÁTICA , 2017.

1. Engenharia de Software. 2. Linha de Produto de Software. 3. Recuperação de Arquitetura. 4.Arquitetura de Linha de Produto de Software. I. Chavez, Christina von Flach Garcia. II. UNIVERSIDADE FEDERAL DA BAHIA. INSTITUTO DE MATEMÁTICA . III Título.

CDD – XXX.XX

CDU – XXX.XX.XXX

Dedico este trabalho a minha família, sem eles nada do que fiz seria possível. A minha professora orientadora que teve paciência e foi crucial para conclusão deste trabalho. Aos demais professores do programa que auxiliaram o desenvolvimento deste trabalho.

AGRADECIMENTOS

Gostaria de agradecer a todas as pessoas que me apoiaram durante estes últimos anos para que fosse possível chegar onde estou. Foram muitos dias de batalha, pesquisa, coleta de métricas, análises e escrita. Durante os momentos de dificuldades pude contar sempre com minha família. Aos meus pais, Normando Sobral Pereira Cardoso e Solange Maria Passos Soares Cardoso, meus eternos agradecimentos por me apoiarem durante todo este tempo e sempre terem acreditado de que seria capaz de finalizar esta etapa em minha vida. A minha irmã Marina Passos Soares Cardoso, que esteve igualmente presente e sempre deu o seu apoio e compreensão. Agradeço em especial a Ana Marta Gomes de Aquino que nunca desistiu desta batalha comigo e sempre me apoiou todas as vezes que pensei desistir. Agradeço também a Crescencio Lima por ter literalmente me acompanhado durante toda esta trajetória, fornecendo orientação, conselhos e suporte durante toda a pesquisa, sem sua ajuda nada disso seria possível. Agradeço também a minha orientadora Christina von Flach G. Chavez, pelo suporte fornecido nos momentos cruciais desta pesquisa.

Mas não se trata de bater forte. Se trata de quanto você aguenta apanhar e seguir em frente, o quanto você é capaz de aguentar e continuar tentando.

—SYLVESTER STALLONE (Rocky Balboa)

RESUMO

Linhas de produtos de software promovem o reúso em larga escala, apoiando a criação, evolução e gerenciamento de portfólios de produtos que compartilham um núcleo comum de características e se diferenciam com base em características variáveis. Na Engenharia de Linha de Produtos de Software, a arquitetura da linha de produtos é um ativo importante, que descreve os pontos de variabilidade da linha de produtos. Se estiver desatualizada ou ausente, pode ser parcialmente recuperada a partir da arquitetura implementada nos produtos da linha. As abordagens para recuperação de arquitetura de software existentes podem ser utilizadas para recuperar a arquitetura de cada produto da linha. Entretanto, técnicas e ferramentas de recuperação, especialmente concebidas para identificar e descrever pontos de variabilidade e comunalidade na arquitetura da linha de produtos propriamente dita, ainda são incipientes. Este trabalho apresenta uma abordagem para recuperação de arquitetura de linhas de produtos de software. A abordagem proposta inclui a técnica PLAR (*Product Line Architecture Recovery*) e a ferramenta PLAR Tool, que implementa a técnica PLAR. A técnica PLAR permite a identificação de pontos de variabilidade e comunalidade da linha de produtos em nível arquitetural. A ferramenta PLAR Tool apóia a recuperação de arquitetura de linhas de produtos, bem como a avaliação de seu grau de reúso. A ferramenta PLAR Tool foi objeto de um estudo realizado com desenvolvedores de linhas de produtos de software, com o propósito de avaliar o resultado da recuperação, com base em visões arquiteturais geradas pela ferramenta, e coletar sugestões de melhoria. Os desenvolvedores não identificaram erros de classificação de elementos arquiteturais com pontos de variabilidade nas arquiteturas recuperadas. Por outro lado, alguns elementos que implementam a variabilidade não foram classificados, sugerindo a necessidade de melhorar a seleção de produtos da linha para uso no processo de recuperação de arquitetura. A ferramenta foi utilizada em dois estudos empíricos, um com o propósito de recuperar a arquitetura de linha de produtos de projetos de código aberto e avaliar sua qualidade, e outro com o propósito de comparar dois métodos de geração de produtos: Padrão x T-Wise. Os principais resultados destes estudos são: existe uma relação estatística entre o número de produtos da linha usados pela técnica PLAR e o grau de reúso da arquitetura recuperada e, o uso do método de geração de produtos T-Wise permitiu que um número reduzido de produtos da linha fosse usado na recuperação de arquitetura, sem comprometer a precisão da recuperação.

Palavras-chave: Linha de Produtos de Software. Arquitetura de Software. Recuperação de Arquitetura de Software. Arquitetura de Linha de Produtos. Recuperação de Arquitetura de Linha de Produtos. Engenharia de Software Experimental.

ABSTRACT

Software Product Lines promote large scale reuse, supporting creation, evolution and management of product portfolio that shares a common core of characteristics that differentiates from each other based on variable characteristics The product line architecture is an important active to the product line software engineering. If it is absent or outdated, it can be partially recovered from products architecture generated by the SPL. The existing techniques and tools can be used to recover the architecture from each SPL product. However, techniques and recovery tools, conceived to identify and document the variability and communality points on the product line architecture still incipient. This work presents an approach to product line architecture recovery that includes the PLAR technique (Product Line Architecture Recovery) and the PLAR Tool. The PLAR technique supports identification and documentation of SPL implemented variabilities and commonalities in architectural level. The PLAR Tool implements the technique and supports recovery and evaluation of the reuse rate of product line architectures. The PLAR Tool was the object of a study conducted with software product line developers, with the goal of evaluating the recovery results of architectural views generated by the tool and collect feedback to improve the tool. Product line developers did not identify errors on the classification of variability elements on the architectures of SPL projects. However, some architectural elements were not classified, suggesting the need of improvement on the selection of SPL products used on the recovery process.

Then, the tool was used in two empirical studies, one with the purpose of recovering the SPL architecture of open source projects and evaluate the architectural quality based on the reuse rate of its components. And another with the purpose of comparing to product generation methods: Standard x T-Wise. The main results of these two studies are: there is an statistical relation between the number of products used on the PLAR technique to recover the product line architecture and the reuse rate and, the T-Wise method for product generation allowed to reduce the number of individual products analyzed without compromising the precision of the recovery process.

Keywords: Software Product Line. Software Architecture. Software Architecture Recovery. Software Product Line Architecture. Software Product Line Architecture Recovery. Empirical Software Engineering.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

Capítulo 1

INTRODUÇÃO

Software é um dos fatores-chave que definem a vantagem competitiva de diversos segmentos da indústria. Nas indústrias automotiva, aeroespacial e eletrônica, por exemplo, há produtos intensivos em software (em inglês, *software-intensive products*), como controle residencial, piloto-automático, controle do processo fabril, entre outros.

Empresas desenvolvem, de forma *ad hoc*, um porfólio de produtos de software que compartilham características em comum, com adição ou remoção de funcionalidades (RUBIN; CHECHIK, 2012) de um produto para outro, para satisfazer necessidades semelhantes, mas não idênticas, de seus clientes. Com o crescimento da quantidade de produtos, o gerenciamento da variabilidade e do reúso começa a ficar prejudicado (SHATNAWI; SERIAI; SAHRAOUI, 2014).

Linhas de Produtos de Software (em inglês, *Software Product Line - SPL*)[1] têm sido usadas em diferentes domínios para explorar características em comum e gerenciar características variáveis que permeiam portfólios de produtos de software. Características em comum ou *comunalidades* (em inglês, *commonalities*) entre produtos da SPL e características variáveis ou *variabilidades* (em inglês, *variabilities*) são gerenciadas, promovendo reúso sistemático e customização de produtos mais confiáveis, com prazos e custos reduzidos (APEL et al., 2013).

A adoção de SPL como paradigma de reúso tem seguido diferentes estratégias (KRUEGER, 2002), por exemplo, desenvolver a SPL antes da geração de qualquer produto (pro-ativa), aplicar transformações em um único produto até derivar uma SPL (reativa), ou aplicar processo de reengenharia sobre vários produtos do domínio para derivar uma SPL (extrativa). Em geral, profissionais da indústria relatam o uso predominante de estratégia extrativa para adoção de SPL (BERGER et al., 2013), com diversos produtos e suas variantes desenvolvidos com técnicas de reúso *ad hoc*, usados para derivar a SPL.

Este trabalho se insere em um contexto geral de investigação de *métodos, técnicas e ferramentas de reengenharia de software* para apoiar a criação e evolução de linhas de produtos de software.

[1] Ao longo do texto da dissertação, a sigla SPL será utilizada.

1.1 PROBLEMA DE PESQUISA

A *Arquitetura da Linha de Produtos* (em inglês, *Product Line Architecture - PLA*)[2] é um dos principais ativos (em inglês, *assets*) de uma SPL. A PLA prescreve o núcleo comum e pontos de variação da SPL (LINDEN; SCHMID; ROMMES, 2007), e restrições relacionadas. Ao ser documentada, a PLA pode ser usada como guia para a geração de novos produtos, base para avaliação da SPL, entre outros usos (CLEMENTS et al., 2010).

A PLA pode ser obtida com apoio de uma abordagem de reconstrução de arquitetura de software (em inglês, *Software Architecture Reconstruction - SAR*)[3]. A reconstrução *bottom-up* de arquitetura, em geral, com base no código-fonte de um sistema de software, é também conhecida como *recuperação de arquitetura* (DUCASSE; POLLET, 2009).

Entretanto, abordagens de SAR para recuperação da arquitetura de sistemas tradicionais (em inglês, *single systems*)[4] não são adequadas para a recuperação de arquitetura de linhas de produtos.

A recuperação de PLA deve permitir a identificação e a representação, em nível arquitetural, do núcleo comum e dos pontos de variação da SPL. Em cenários de recuperação com base no código-fonte, a identificação de variabilidade pode requerer um número arbitrário de produtos da SPL.

Abordagens de SAR para recuperação de PLA existentes (EIXELSBERGER, 2000; SHATNAWI; SERIAI; SAHRAOUI, 2014; LINSBAUER; LOPEZ-HERREJON; EGYED, 2016) possuem algumas limitações, por exemplo, inexistência ou indisponibilidade de suporte ferramental, informações insuficientes sobre a técnica de reconstrução utilizada (como a reconstrução é realizada, como os produtos são escolhidos, como são identificados elementos comuns e variáveis) e inexistência de avaliação da PLA recuperada. Algumas contribuições e limitações de trabalhos relacionados serviram como ponto de partida para nossa pesquisa.

1.2 OBJETIVO GERAL

O objetivo geral deste trabalho é apoiar a recuperação de arquitetura de linha de produtos de software, e servir para a comunicação entre interessados e avaliação da SPL. A PLA recuperada deve conter elementos arquiteturais que representem a variabilidade da SPL e seus relacionamentos

Para alcançar esse objetivo, uma técnica para recuperação de arquitetura de linhas de produtos foi proposta, implementada e avaliada.

1.3 OBJETIVOS ESPECÍFICOS

O1 Propor uma técnica para recuperação de arquitetura de linhas de produtos.

[2]Ao longo do texto da dissertação, a sigla PLA será utilizada.

[3]Ao longo do texto da dissertação, a sigla SAR será utilizada.

[4]Ao longo do texto da dissertação, o termo "sistema tradicional" é utilizado como tradução para "single system".

A técnica de recuperação de arquitetura de linhas de produtos proposta, chamada de PLAR (*Product-Line Architecture Reconstruction*), recupera informações arquiteturais com base em extração de informações do código-fonte de produtos da SPL e sua posterior combinação em informações arquiteturais da PLA. O escopo da técnica PLAR está limitado à recuperação de informações arquiteturais (elementos, relacionamentos, pontos de variação) usados em visões de módulo (CLEMENTS et al., 2010).

O2 Desenvolver uma ferramenta que automatize a técnica de recuperação de PLA proposta.

A ferramenta PLAR Tool automatiza e serve como prova de conceito para a técnica PLAR. Sua principal funcionalidade é a recuperação da PLA, a partir da combinação de informações extraídas de um conjunto de produtos relacionados. A PLA recuperada contém informações sobre módulos e relacionamentos usados na descrição de arquitetura de sistemas tradicionais, e informações sobre comunalidades e variabilidades da SPL.

A ferramenta também apóia a avaliação da qualidade da PLA recuperada e a geração de diferentes visões arquiteturais para documentação da PLA.

O3 Avaliar a técnica de recuperação de PLA proposta.

A avaliação da técnica proposta deve ser realizada com base em sua implementação pela ferramenta PLAR Tool, e considerar ao menos a precisão e abrangência da recuperação com respeito à identificação de pontos de variabilidade da SPL.

O4 Avaliar a qualidade da PLA recuperada.

A avaliação da PLA recuperada deve considerar ao menos o grau de reúso. O grau de reúso é uma das formas de mensurar a qualidade da PLA (ZHANG et al., 2008), os critérios para essa avaliação são detalhados no capítulo 2.

Neste trabalho de mestrado realizamos três estudos para alcançar este objetivo. Um primeiro estudo foi realizado com desenvolvedores de projetos SPL a fim avaliar a qualidade da PLA recuperar pela ferramenta. Um segundo estudo foi realizado com projetos SPL open-source também com o propósito de avaliar a qualidade das PLA recuperadas de acordo com o reúso dos seus componentes. O terceiro estudo realizado buscou comparar dois métodos de geração de produtos da SPL com o objetivo de verificar a possibilidade de redução de produtos analisados e ainda assim manter o mesmo padrão de qualidade na PLA recuperada.

1.4 QUESTÕES DE PESQUISA

As questões de pesquisa que norteiam este trabalho, relacionadas aos objetivos O1-O4, são:

Q1.1 Quais *técnicas* para recuperação de PLA utilizam informações extraídas do código-fonte de produtos da SPL? Esta questão está relacionada ao objetivo **O1**.

Q2.1 Quais *ferramentas* para recuperação de PLA utilizam informações extraídas do código-fonte de produtos da SPL? Esta questão está relacionada ao objetivo **O2**.

As questões de pesquisa associadas ao objetivo **O3** são:

Q3.1 A abrangência e a precisão da recuperação da PLA é influenciada pelos produtos da SPL utilizados na recuperação?

Q3.2 Visões arquiteturais com representação de variabilidade ajudam a compreender a variabilidade da SPL?

As questões de pesquisa associadas ao objetivo **O4** são:

Q4.1 Quais métricas são utilizadas para avaliação da PLA?

Q4.2 O número de produtos da SPL utilizados na recuperação da PLA interfere nos resultados de métricas de reúso?

1.5 MÉTODOS DE PESQUISA

Os métodos utilizados para alcançar os objetivos e responder as questões de pesquisa associadas são apresentados na Figura 1.1 e explicados em seguida.

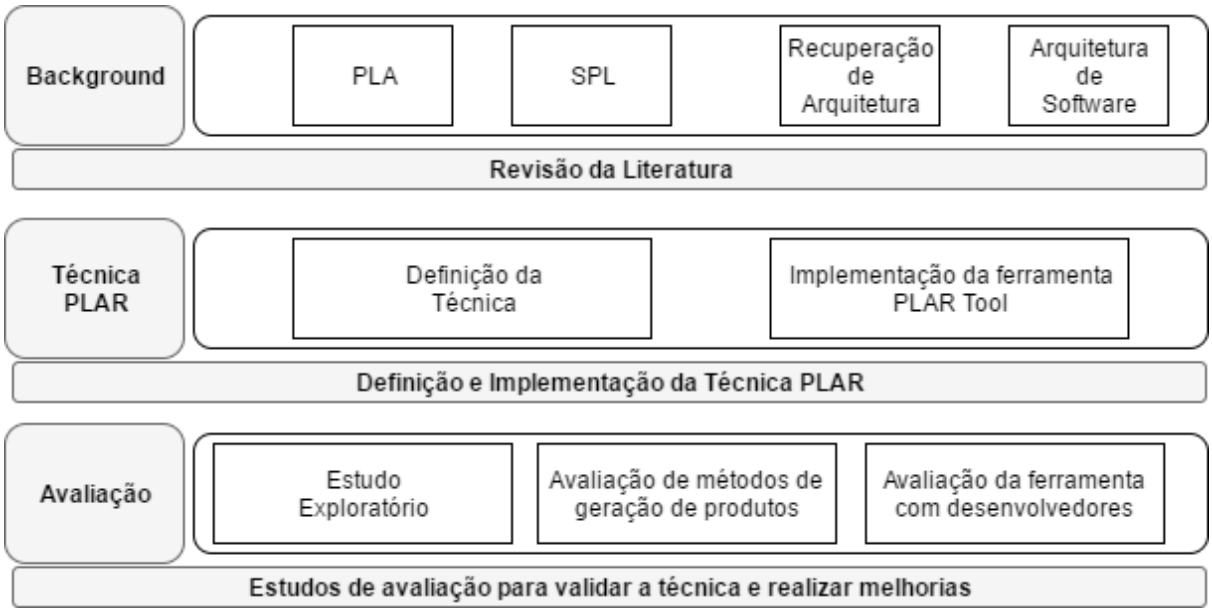

Figura 1.1 Metodologia de Pesquisa utilizada no desenvolvimento dessa dissertação

Revisão da Literatura. Os objetivos de pesquisa O1 e O2 são de natureza descritiva. Para responder às questões de pesquisa Q1.1, Q2.1 e Q4.1, foi realizada uma revisão da literatura de forma *ad-hoc*.

Livros e artigos científicos serviram para fundamentar os conceitos básicos sobre arquitetura de software e linhas de produto de software, e métricas para avaliação da qualidade das PLAs. Artigos científicos sobre técnicas e ferramentas de reconstrução de PLA foram revisados. Técnicas e ferramentas relacionadas foram selecionadas, caracterizadas e comparadas com abordagem proposta neste trabalho. Finalmente, métricas de avaliação de arquitetura de linhas de produtos de software foram investigadas para implementação e uso nos estudos empíricos realizados.

Definição da Técnica PLAR. A técnica PLAR recupera informações arquiteturais com base na extração de informações do código-fonte de produtos da SPL e sua posterior combinação em informações arquiteturais da PLA que contemplem aspectos de comunalidade e variabilidade da SPL. Seu escopo está limitado à recuperação de informações arquiteturais (elementos e relacionamentos) usados em visões de módulo (CLEMENTS et al., 2010).

Avaliação da Técnica PLAR. A PLAR Tool foi implementada para servir como prova de conceito para a técnica de recuperação proposta e apoiar atividades de recuperação da PLA.

Um *survey* com desenvolvedores de dois projetos SPL *open source* foi conduzido para avaliar a qualidade da *documentação de arquitetura para PLA* gerada pela ferramenta PLAR Tool e responder a questão Q3.2.

A técnica PLAR foi avaliada com base na quantificação e análise da precisão e abrangência da PLA recuperada com respeito à identificação de pontos de variabilidade da SPL.

Com base nos resultados da avaliação, ajustes e melhorias foram introduzidos na técnica proposta e implementados na ferramenta PLAR Tool.

Avaliação do Grau de Reuso. Um estudo empírico foi realizado para responder às questões de pesquisa Q3.1. Foram utilizados seis projetos SPL *open source* e alguns produtos disponíveis, para a reconstrução de PLA com a ferramenta PLAR Tool. A qualidade de cada PLA recuperada foi avaliada com base no reúso de componentes. O grau de reúso da PLA é analisado em duas dimensões: a quantidade de vezes que um determinado elemento da PLA aparece nos demais produtos e a razão entre elementos comuns e variáveis mapeados na PLA.

Avaliação dos métodos de Geração de Produtos. Um estudo empírico foi realizado para responder a questão de pesquisa Q4.2. Foram usados dez projetos SPL *open source* e os produtos foram gerados automaticamente por meio de dois métodos distintos de geração de produtos, o método padrão onde todos os produtos gerados a partir de uma configuração válida[5] são gerados e o método T-Wise (HENARD et al., 2014) que reduz o número de produtos gerados pela SPL ao escolher as configurações mais significativas. Os resultados das métricas foram obtidas para os dois métodos e comparados.

[5]Uma configuração é válida quando não viola as restrições do diagrama de *features*

1.6 PRINCIPAIS RESULTADOS

Os principais resultados deste trabalho de mestrado são:

- A comparação entre diferentes técnicas de recuperação de PLA mostrou que cada técnica utiliza sua própria visão arquitetural para representar a PLA, não há uma uniformidade.
- A implementação da técnica de recuperação proposta pela ferramenta PLAR Tool permitiu que pudéssemos fazer a prova de conceito da técnica e realizar aperfeiçoamentos.
- Na avaliação da ferramenta, desenvolvedores de projetos SPL não identificaram erros de classificação de elementos arquiteturais com pontos de variabilidade nas PLAs recuperadas.
- Alguns elementos arquiteturais variáveis obtidos a partir dos projetos SPL do *survey* não foram classificados, e melhorias foram introduzidas no processo de geração e seleção de produtos para a recuperação da PLA.
- Os estudos apresentaram indícios de correlação entre o número de produtos utilizados para recuperação da PLA e seu grau de reúso.
- O método T-Wise mostrou-se vantajoso para recuperação da PLA e seus elementos variáveis, pois permitiu a redução no número de produtos usados, sem comprometer a precisão e abrangência dos resultados.

1.7 ESTRUTURA DA DISSERTAÇÃO

O capítulo 2 apresenta conceitos sobre arquitetura de software, linhas de produtos de software, recuperação de arquitetura e métricas de reúso, que fundamentam esta dissertação de mestrado. O capítulo 3 apresenta e detalha a técnica de recuperação proposta neste trabalho, e a ferramenta PLAR Tool, desenvolvida como prova de conceito da técnica. Os três capítulos seguintes apresentam estudos realizados para avaliar a técnica PLAR e a ferramenta PLAR Tool. O capítulo 4 apresenta os resultados do estudo realizado com desenvolvedores de projetos SPL com a finalidade de avaliar as saídas geradas pela ferramenta. Os resultados obtidos foram utilizados para realizar melhorias na ferramenta e na técnica proposta. O capítulo 5 apresenta os resultados da avaliação realizada em 6 projetos SPL open-source, com o objetivo de avaliar a qualidade da PLA recuperada. O capítulo 6 apresenta um estudo comparativo entre dois métodos de geração de produtos: o convencional (onde todos os produtos que podem ser gerados a partir de todas as combinações possíveis da SPL podem ser gerados) e o método T-Wise que busca reduzir o conjunto de produtos gerados mantendo a representatividade arquitetural. Por último, as conclusões da dissertação são apresentadas no capítulo 7.

Capítulo 2

FUNDAMENTAÇÃO TEÓRICA

As três primeiras seções deste capítulo apresentam conceitos de arquitetura de software (seção 2.1), documentação de arquitetura de software (seção 2.2) e reconstrução de arquitetura de software (seção 2.3).

As últimas seções apresentam conceitos de linhas de produto de software (seção 2.4) e retomam questões de arquitetura de software, documentação de arquitetura e reconstrução de arquitetura no contexto de linhas de produto de software (seção 2.5).

2.1 ARQUITETURA DE SOFTWARE

A arquitetura de um sistema de software é a estrutura ou estruturas do sistema, composta por elementos de software, propriedades externamente visíveis destes elementos e relacionamentos entre eles (CLEMENTS, 2002). A arquitetura de software também diz respeito ao conjunto de decisões de *design* tomadas durante o seu desenvolvimento e evolução do sistema (TAYLOR; MEDVIDOVIC; DASHOFY, 2009).

As decisões de *design* arquitetural tomadas antes da construção de um sistema de software constituem sua arquitetura conceitual (em inglês, *as-intended architecture*) ou prescritiva. A arquitetura conceitual é obtida durante o projeto arquitetural do sistema. As decisões de *design* (prescritas ou não por uma arquitetura conceitual) implementadas por um ou mais artefatos, incluindo o código-fonte do sistema de software, constituem sua arquitetura implementada (em inglês, *as-realized architecture*) ou arquitetura descritiva. A arquitetura implementada pode ser obtida por meio de técnicas de recuperação de arquitetura (TAYLOR; MEDVIDOVIC; DASHOFY, 2009).

A seguir, apresentamos os conceitos que têm relação direta com o escopo desta dissertação.

2.1.1 Arquiteturas e descrições arquiteturais

Descrições arquiteturais são artefatos resultantes do *design* arquitetural, usadas para expressar arquiteturas de sistemas de software.

O padrão internacional ISO/IEC/IEEE 42010:2011 (ISO/IEC/IEEE. . . , 2011) especifica como organizar e expressar descrições arquiteturais de sistemas. O padrão provê uma ontologia para a descrição arquitetural, que inclui os elementos apresentados na Figura 2.1.

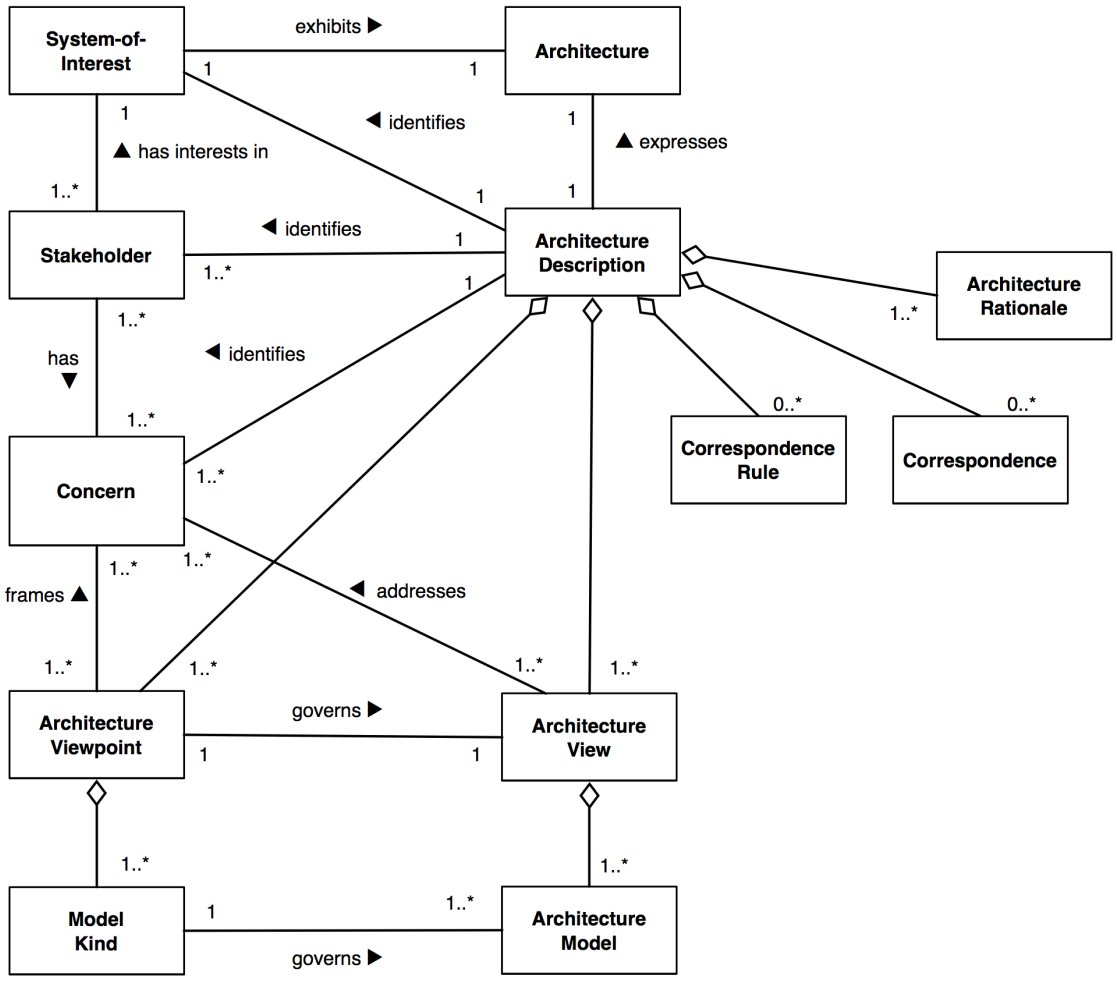

Figura 2.1 Modelo conceitual de uma descrição arquitetural (ISO/IEC/IEEE. . . , 2011)

2.1.2 Interessados e interesses

A arquitetura de software possui diversos interessados (em inglês, *stakeholders*). Os interessados no sistema têm diferentes preocupações e níveis de conhecimento. Assim, a arquitetura não deve ser apresentada da mesma maneira para interessados diferentes. Visões arquiteturais descrevem o sistema ou parte dele segundo a perspectiva de um conjunto de interesses (em inglês, *concerns*) relacionados (CLEMENTS et al., 2010).

2.1.3 Visões e pontos-de-vista arquiteturais

Uma descrição arquitetural inclui uma ou mais visões arquiteturais (em inglês, *architectural views*). Uma visão arquitetural expressa um ou mais interesses definidos pelos interessados no sistema, segundo um ponto-de-vista arquitetural. Um ponto-de-vista arquitetural (em inglês, *viewpoint* ou *viewtype* (CLEMENTS et al., 2010)) estabelece as convenções para a construção, interpretação, análise e uso de visões arquiteturais que aderem a tal ponto-de-vista.

2.1.4 Modelos arquiteturais

Uma visão arquitetural deve ser composta por um ou mais modelos arquiteturais, que devem identificar o tipo de modelo no qual se baseiam e aderir às convenções deste modelo. Um modelo arquitetural pode fazer parte de mais de uma visão arquitetural (ISO/IEC/IEEE..., 2011).

2.1.5 Decisões arquiteturais e rationale

As decisões de *design* da arquitetura fazem parte de sua concepção e evolução, podendo estar relacionadas ao domínio de aplicação, estilos e padrões arquiteturais utilizados, entre outros aspectos necessários para satisfazer os requisitos do sistema (JANSEN; BOSCH, 2005).

Decisões de *design* compreendem um conjunto de modificações arquiteturais (adições, remoções e alterações), justificativas, regras de *design*, restrições de *design* e requisitos adicionais associados (JANSEN; BOSCH, 2005). As justificativas devem descrever o porquê da alteração na arquitetura do sistema. As regras de *design* definem as prescrições para futuras decisões a serem tomadas. As restrições de *design* definem o que não pode ser alterado na arquitetura. Finalmente, uma decisão de *design* pode implicar na adição de novos requisitos a serem satisfeitos pela arquitetura. Estes novos requisitos devem ser tratados por outras decisões de *design*.

2.2 DOCUMENTAÇÃO DA ARQUITETURA DE SOFTWARE

A documentação da arquitetura de software (CLEMENTS et al., 2010) oferece informações para diferentes tipos de interessados. Por exemplo, documentos e modelos da arquitetura podem conter informações importantes para desenvolvedores, testadores, gerentes, dentre outros (CLEMENTS et al., 2010).

A documentação da arquitetura serve a diferentes propósitos. Por exemplo, quando suficientemente detalhada, pode servir como um guia para a construção e evolução do software; quando formalizada, pode servir de base para diversos tipos de análise (TAYLOR; MEDVIDOVIC; DASHOFY, 2009). Dentre os usos principais para a documentação de arquitetura de software (CLEMENTS et al., 2010), destacamos:

- **Educação.** A documentação é útil para introduzir o sistema para novas pessoas que irão trabalhar no sistema que podem ser novos membros, analistas ou até mesmo um novo arquiteto.

- **Comunicação entre interessados**. Diferentes interessados no sistema podem usar a documentação de arquitetura de software como base para comunicação.

- **Análise.** A documentação da arquitetura deve prover informação suficiente para fornecer suporte a sua análise como por exemplo, informações a respeito da segurança do sistema, desempenho, usabilidade entre outros aspectos que podem ser postos em análise.

Clements et. al. 2010 propõem que a documentação de arquitetura de software contemple três tipos de visão arquitetural (*architectural viewtype*): tipo de visão de módulo, tipo de visão de componente-conector e tipo de visão de alocação. Cada tipo de visão arquitetural inclui estilos arquiteturais. Nesta dissertação, utilizaremos apenas o tipo de visão arquitetural de módulo e seus estilos arquiteturais.

Módulo. O tipo de visão arquitetural de módulo (em inglês, *module viewtype*) mapeia os módulos do sistema, que são unidades de funcionalidade que implementam um conjunto de responsabilidades. Estas unidades de funcionalidade podem ser uma classe, um agrupamento de arquivos de código fonte, pacotes, entre outros. Além de mapear os módulos do sistema, este tipo de visão arquitetural também mapeia as relações entre estes módulos que podem ser: *é parte de*, *depende de*, *é um*. A Tabela 2.1 sumariza estas relações.

Tabela 2.1 Relações da *viewtype* de módulo (CLEMENTS et al., 2010)

Relação	Descrição
É parte de	Define que dois módulos possuem algum tipo de agregação.
Depende de	Define que um módulo depende de outro.
É um	Define uma generalização entre dois módulos.

Há quatro estilos arquiteturais associados ao tipo de visão arquitetural de módulo (Figura 2.2): decomposição, uso, generalização e camada. Cada estilo usa a representação básica da visão arquitetural de módulo (em geral, UML), podendo adicionar novos detalhes à representação dos módulos e suas relações.

A seguir, detalhamos apenas os estilos arquiteturais utilizados neste trabalho de mestrado.

- **Decomposição:** Este estilo é usado para apresentar as responsabilidades do sistema e como elas estão divididas entre os módulos. Responsabilidades são decompostas em módulos e submódulos, a fim de alcançar determinados critérios: critérios de qualidade, definir módulos a serem reutilizados, e fornecer suporte a implementação de linhas de produto de software (CLEMENTS et al., 2010).

- **Uso:** Este estilo introduz a relação *usa* (*uses*), que especializa a relação *depende de*, para definir quais módulos usam "serviços" de outros módulos.

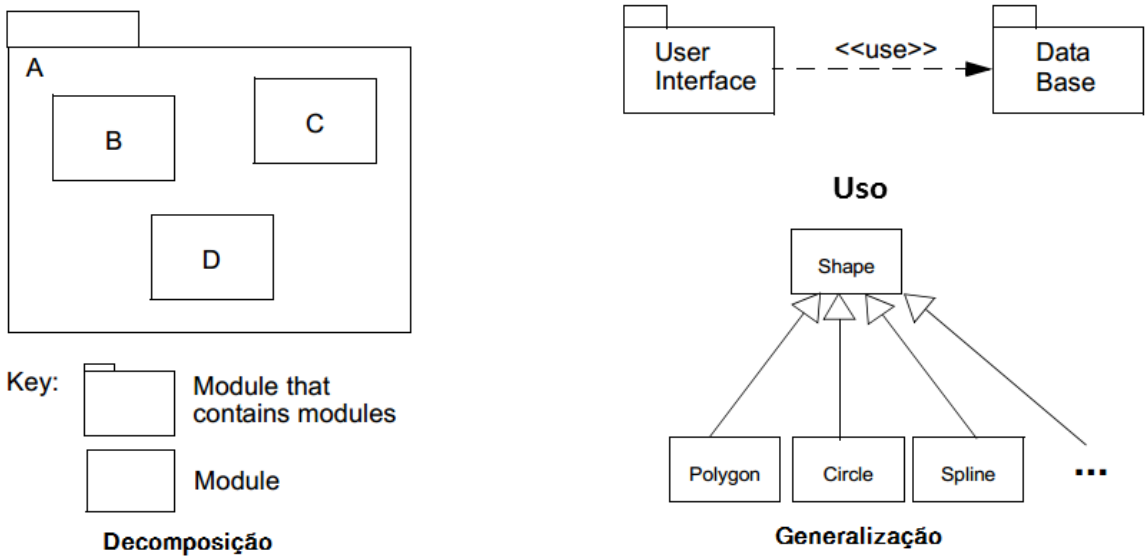

Figura 2.2 Estilos da visão de módulo (CLEMENTS et al., 2010)

- **Generalização:** Este estilo é usado quando a relação *é um* é especializada para uma relação do tipo *generalização*. Este estilo é utilizado quando o arquiteto deseja permitir a extensão e evolução da arquitetura e dos módulos, de forma que a arquitetura forneça suporte a mecanismos de generalização como a herança, por exemplo.

2.3 RECONSTRUÇÃO DE ARQUITETURA DE SOFTWARE

A reconstrução de arquitetura de software (SAR) busca resgatar arquitetura conceitual (*as-intended architecture*) ou a arquitetura implementada (*as-realized architecture*) de um sistema de software (DUCASSE; POLLET, 2009). Em geral, a arquitetura conceitual do software não existe ou está desatualizada. Por outro lado, a arquitetura implementada por um sistema de software existe e pode ser parcialmente recuperada a partir de seu código fonte e arquivos do tipo *makefile*.

Abordagens de SAR podem ser classificadas segundo diversas características. A Figura 2.3 apresenta uma taxonomia orientada a processos para SAR, organizada em cinco eixos: objetivos, processos, entradas, técnicas e saídas (DUCASSE; POLLET, 2009). Neste trabalho, essa taxonomia é usada para apresentar conceitos de SAR e classificar trabalhos relacionados.

O eixo de **objetivos** define seis tipos de objetivos para SAR: redocumentação, reúso, compreensão, evolução, análise e gerenciamento de arquitetura. No eixo de **processos**, as abordagens de SAR são classificadas como *bottom-up*, *top-down* ou híbridas. No eixo de **técnicas**, as abordagens de SAR são classificadas segundo seu nível de automação em *quasi*-manual, semiautomática ou *quasi*-automática. No eixo de **entradas**, as abordagens de SAR são classificadas segundo o tipo de entrada em: arquiteturais e não-arquiteturais

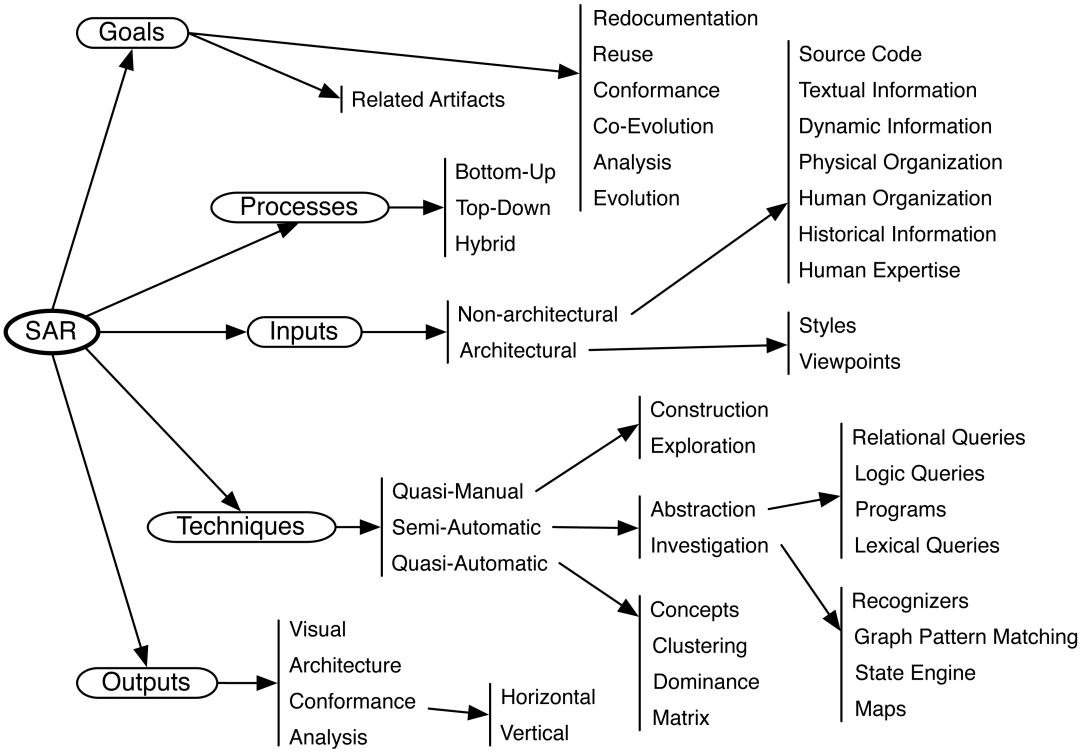

Figura 2.3 Uma Taxonomia Orientada a Processos para Reconstrução de Arquitetura de Software (DUCASSE; POLLET, 2009)

(código-fonte, informações textuais, etc.). No eixo de **saídas**, as abordagens de SAR são classificadas de acordo com o tipo de saída (*output*) do processo de reconstrução.

2.3.1 Objetivos

A reconstrução de arquitetura tem como principal objetivo resgatar a arquitetura de um software para que ela possa ser utilizada nos processos de manutenção e evolução do sistema. Recentemente, a recuperação de arquitetura tem sido utilizada também para identificar componentes reutilizáveis e realizar a migração de sistemas para o paradigma de linha de produto de software.

SAR pode ser usada para verificação de conformidade, de forma que os desenvolvedores possam comparar a arquitetura conceitual com a arquitetura realizada do sistema. Além disso, a recuperação da arquitetura auxilia os desenvolvedores no processo de manutenção e evolução do sistema, pois através das visões arquiteturais fornecidas é possível ter uma visão dos componentes do sistema e sua forma de interação, facilitando este processo.

Outro objetivo da SAR é a compreensão do sistema, pois em alguns casos, o software em análise não possui uma arquitetura conceitual definida. Dessa forma, a recuperação da mesma auxilia os desenvolvedores na compreensão do sistema.

2.3.2 Processos

O processo de reconstrução de arquitetura pode ser realizado com base em informações extraídas diretamente do código-fonte (abordagem *bottom-up*), com base em informações extraídas de documentos, decisões de *design* ou outros tipos de artefatos (abordagem *top-down*), ou combinando as duas estratégias anteriores para otimizar os resultados da reconstrução (abordagem híbrida).

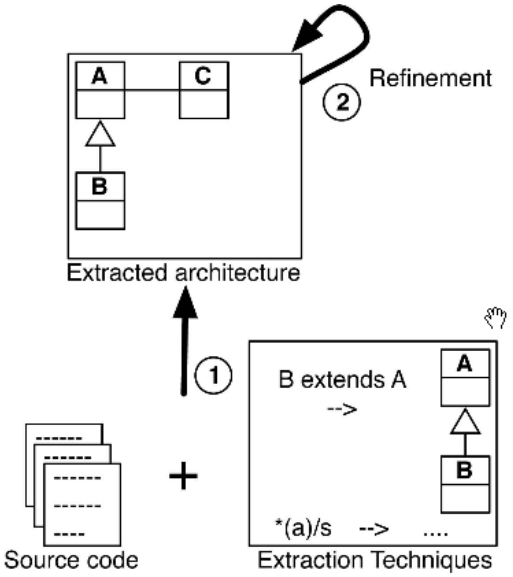

Figura 2.4 Processo *Bottom-Up* (DUCASSE; POLLET, 2009)

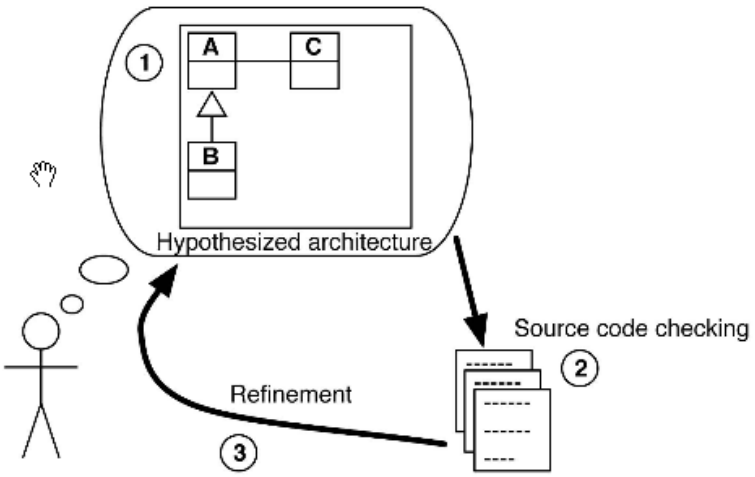

Figura 2.5 Processo *Top-Down* (DUCASSE; POLLET, 2009)

- ***Top-Down*** – Processos *top-down* partem de conhecimento de alto nível a respeito do software, por exemplo, requisitos ou alguns estilos arquiteturais. A partir destes artefatos, busca-se descobrir a arquitetura do sistema através da formulação de hipóteses e comparação com o código fonte, como pode ser visto na Figura 2.5. Processos *top-down* são também conhecidos como *processos de descoberta de arquitetura*.

- ***Bottom-up*** – Processos *bottom-up* normalmente começam com pouco conhecimento sobre a arquitetura de um sistema de software e, partindo do seu código fonte, identificam informações arquiteturais de nível de abstração mais alto. As informações arquiteturais podem ser armazenadas em um repositório e disponibilizadas para os interessados no sistema de software (TILLEY; SMITH; PAUL, 1996). Processos *bottom-up* também são conhecidos como *recuperação de arquitetura*. A Figura 2.4 mostra um processo *bottom-up* de SAR.

- **Híbrido** – Processos híbridos combinam informações acerca da arquitetura do sistema recuperadas do código-fonte com estratégias de processos *bottom-up*, com informações de alto nível do sistema, descobertas com estratégias de processos *top-down*.

Neste trabalho, apenas processos de recuperação de arquitetura (*bottom-up*) serão considerados.

2.3.3 Entradas

As abordagens de SAR podem demandar diferentes tipos de entrada. Os tipos de entrada são classificados em arquiteturais e não-arquiteturais (código-fonte, informações textuais, etc.). Normalmente, o código-fonte do projeto e a experiência de bons engenheiros de software são importantes fontes de informação para o processo de reconstrução.

No caso de processos de recuperação (*bottom-up*), usualmente contamos com três tipos de entrada: código fonte, perícia humana e organização de arquivos. Contudo, outros tipos de dados de entrada podem estar disponíveis para outros tipos de processo (DUCASSE; POLLET, 2009).

- **Código Fonte** – É a fonte de informação "onipresente"; a maioria das técnicas de recuperação de arquitetura usa o código fonte de alguma forma – seja na consulta direta ao código fonte para obtenção de informações ou na criação de metamodelos a partir da análise do código fonte.

- **Perícia Humana** – Em muitos casos, desenvolvedores com conhecimento sobre o sistema de software podem ajudar no processo *top-down* de recuperação de arquitetura. Informações de desenvolvedores ativos podem ser utilizadas para guiar o processo e validar resultados.

- **Organização dos Arquivos** – Algumas técnicas consideram a estrutura de pacotes e arquivos de um sistema de software e, a partir dela, são capazes de recuperar informações arquiteturais.

2.3.4 Técnicas

As técnicas para recuperação de arquitetura podem ser classificadas segundo os dados que manipulam e o grau de participação humana durante o processo de recuperação:

- ***Quasi*-manual** – Nas técnicas do tipo *quasi*-manual, o engenheiro de software identifica os elementos da arquitetura utilizando ferramentas que o auxiliam a entender o que está extraindo.

- **Semiautomática** - Nas técnicas semiautomáticas, o engenheiro de software instrui, de forma manual, como a ferramenta deve descobrir e recuperar as abstrações da arquitetura.

- ***Quasi*-automática** – Nas técnicas *quasi*-automáticas, a ferramenta tem total controle sobre o processo de SAR, cabendo ao engenheiro apenas conduzir o processo.

2.3.5 Saídas

A depender do objetivo da reconstrução, uma técnica de SAR pode produzir diferentes saídas:

- **Visões Arquiteturais -** É saída principal fornecida pela maioria das técnicas de recuperação de arquitetura. Uma técnica pode fornecer uma ou mais visões arquiteturais.

- **Arquitetura -** Algumas técnicas de recuperação de arquitetura fornecem determinadas visões arquiteturais a fim de prover uma representação da arquitetura recuperada.

- **Conformidade -** Algumas abordagens tem como objetivo verificar a conformidade entre a arquitetura recuperada (implementada) e a arquitetura conceitual do projeto analisado, fornecendo como saída o resultado desta verificação.

- **Análise -** Algumas técnicas de recuperação podem ainda realizar análises adicionais, além da verificação de conformidade, por exemplo, avaliação da qualidade, verificação de propriedades, análise evolucionária, busca de componentes reutilizáveis, entre outras (DUCASSE; POLLET, 2009).

2.4 LINHAS DE PRODUTOS DE SOFTWARE

Reuso de Software é o processo de criar novos produtos de software por meio da reutilização de partes existentes, ao invés de construí-las (KRUEGER, 2010). Alguns benefícios do reúso de software são: desenvolvimento mais rápido, custo reduzido, uma vez que partes de código são reutilizados, entre outros .

Linhas de produtos de software (SPL) promovem reúso sistemático de software. Uma SPL é um conjunto de produtos dentro do portfólio de uma empresa, que compartilham similaridades e que são gerados através de componentes reusáveis (APEL et al., 2013).

No modelo atual de produção, as empresas oferecem a customização em massa de seus produtos, ou seja, os clientes podem customizar o produto, escolhendo uma série de características pré-definidas de acordo com os componentes reusáveis disponibilizados pela empresa para aquele determinado produto.

O uso de linhas de produtos de software traz diversos benefícios (APEL et al., 2013):

- **Personalização dos produtos.** É possível customizar o produto de acordo com as necessidades do cliente.
- **Custos reduzidos.** Os componentes reusáveis serão desenvolvidos apenas uma vez; logo, os custos para montar novos produtos são reduzidos, já que isso é feito de forma automática por alguma ferramenta apropriada.
- **Melhoria da qualidade.** A qualidade dos produtos gerados é aumentada, já que os componentes reusáveis serão utilizados em diversos produtos e, consequentemente, amplamente testados.
- **Tempo de entrega.** Os produtos da linha serão construídos com base em componentes reutilizáveis, o que leva à redução do tempo para entrega de novos produtos, permitindo que mais produtos sejam entregues em menos tempo.

Cada produto da linha de produtos de software deve ser criado pela seleção de componentes disponíveis em uma *coleção de componentes reutilizáveis* e de mecanismos de variação, adicionando novos componentes, se necessário. O *produto da linha* deve estar em conformidade com regras de uma arquitetura comum a toda a linha de produtos de software[1].

A Figura 2.6 apresenta as etapas do processo de desenvolvimento de uma SPL. O desenvolvimento é dividido em duas etapas: Engenharia de Domínio e Engenharia de Aplicação. Na engenharia de domínio, o domínio de aplicação da SPL é analisado e a partir dele as *features* que irão compor a SPL são elicitadas. A etapa de engenharia da aplicação é responsável por codificar estas *features* e gerar os produtos da SPL.

2.4.1 Conceitos de SPLs

Features.

A principal característica de uma linha de produtos de software de terceira geração são suas *features*, pois capturam os desejos dos *stakeholders* em relação aos requisitos do sistema. Uma *feature* é uma característica ou um comportamento visível ao usuário de um sistema de software (KANG et al., 1990) São usadas nas linhas de produto para especificar e comunicar as comunalidades e diferenças dos produtos entre os *stakeholders* e para guiar a estruturação, o reúso e a variação entre todas as fases do ciclo de desenvolvimento de software (APEL et al., 2013). O conjunto

[1]Alguns autores usam termos como *plataforma* em lugar de coleção de componentes reutilizáveis, *customização* em lugar de produto da linha, e *família de produtos* ao invés de a linha de produtos de software.

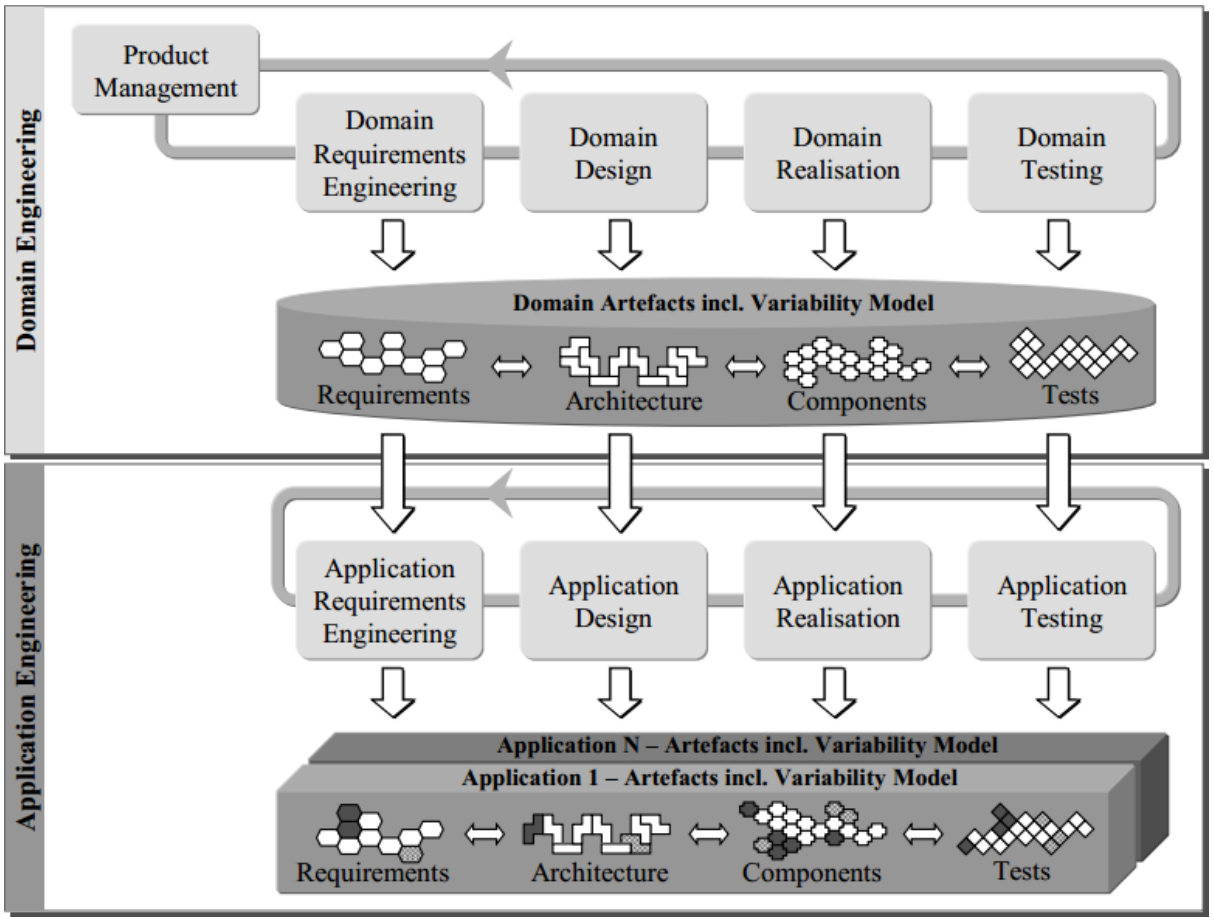

Figura 2.6 Processo de desenvolvimento de uma SPL (LINDEN; SCHMID; ROMMES, 2007)

de *features* dá a SPL a flexibilidade necessária para gerar a variabilidade na linha, sendo isso a base para a customização em massa dos produtos de uma SPL(POHL; BöCKLE; LINDEN, 2005).

O produto de uma SPL é definido pelo conjunto de *features* que compõe o produto e suas relações (APEL et al., 2013). Nem todas as combinações são válidas; existem determinadas *features* que não podem se relacionar com outras. Por exemplo, considerando uma linha de produtos que tenha como *feature* o suporte a diversos sistemas operacionais (SO), não é possível instanciar um produto que ofereça suporte a mais de um SO simultaneamente.

Engenharia de Domínio.

Para o desenvolvimento de uma SPL, ao invés de se focar em um único produto, a equipe de desenvolvimento deve considerar uma área do conhecimento e um domínio específico, que permitam conceber uma variedade de possíveis produtos para aquele domínio em particular.

O domínio de aplicação de uma SPL é um conjunto de aplicações atuais e futuras

que compartilham dados e capacidades em comum (KANG et al., 1990), complementando esta definição, Apel (APEL et al., 2013) define que um domínio é uma área de conhecimento que:

- Busca maximizar o cumprimento dos anseios dos *stakeholders*;
- Deve incluir um conjunto de termos e conceitos identificados pelos especialistas da área;
- Inclui conhecimento de como construir sistemas de software nesta área do domínio.

Quanto mais amplo for o domínio para uma SPL maior será a quantidade de requisitos dos *stakeholders* que deverão ser satisfeitos. Além disso, a comunalidade entre os produtos tende a diminuir (APEL et al., 2013). Para evitar este tipo de problema, a engenharia de domínio (POHL; BöCKLE; LINDEN, 2005) prescreve a realização de uma análise criteriosa do domínio antes do desenvolvimento de artefatos reusáveis. A engenharia de domínio é o processo pelo qual a comunalidade e a variabilidade de uma SPL são definidos (POHL; BöCKLE; LINDEN, 2005). A engenharia de domínio não irá gerar um único produto específico, mas sim preparar artefatos para serem utilizados em múltiplos, se não todos os produtos de uma SPL (APEL et al., 2013).

A engenharia de domínio é dividida em quatro etapas (APEL et al., 2013; POHL; BöCKLE; LINDEN, 2005):

- **Análise de Domínio** - Nesta etapa o escopo do domínio é definido, isto é, quais produtos serão feitos por esta SPL e quais *features* devem ser implementadas e reutilizadas. Os resultados da análise de domínio normalmente são documentadas em um modelo de *features*.
- **Análise de Requisitos** - Nesta etapa as necessidades de um cliente em específico são levantadas. Na maioria dos casos, as *features* já implementadas que satisfazem estas necessidades são escolhidas para a montagem do produto. Caso surja um novo requisito, volta-se a etapa de análise de domínio, o que pode resultar em uma alteração do modelo de *features*.
- **Implementação do domínio** - Nesta etapa os artefatos reusáveis serão desenvolvidos com base nas *features* identificadas durante a análise de domínio.
- **Geração do produto** - Nesta etapa o produto será gerado de acordo os resultados obtidos pela etapa de análise de requisitos, utilizando as *features* necessárias.

Modelo de *Features*.

O modelo de *features* documenta as *features* da linha de produto de software e seu relacionamento (APEL et al., 2013). Nem sempre essas *features* são compatíveis entre si, e algumas delas podem requisitar a presença de outras, portanto o modelo de *features* além de representar, também define quais escolhas de *features* são

válidas. A fim de facilitar a representação do modelo de *features*, são utilizados os diagramas de *features* que são uma notação gráfica para especificar o modelo. É uma árvore cujo os nós são rotulados com os nomes das *features*, estabelecendo uma hierarquia entre elas, sendo que esta árvore pode ser também formalizada utilizando lógica proposicional (Figura 2.7) (APEL et al., 2013).

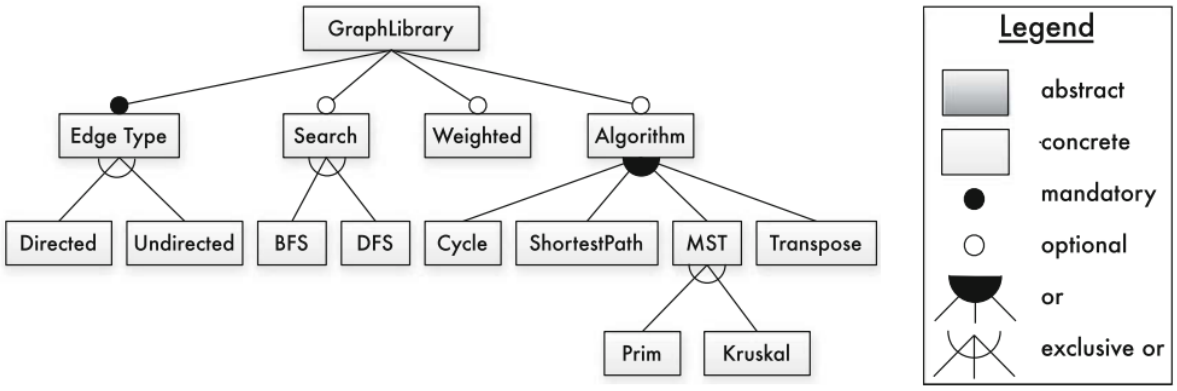

Figura 2.7 Diagrama de *Features* (APEL et al., 2013)

Normalmente representa-se no modelo de *features* o relacionamento entre as *features* de 4 maneiras distintas, porém existem autores que usam outras formas de representação, entretanto esses 4 tipos de representação são os mais comuns dentro da literatura (APEL et al., 2013) :

- **Mandatória** - Define a *feature* como sendo obrigatória, é representada por um círculo preenchido.
- **Opcional** - Define a *feature* como sendo opcional, é representada por um círculo vazado.
- **Alternativa** - Define um conjunto de *features* onde apenas uma delas deve ser escolhida excluindo as demais, é representada por um arco vazado.
- **XOR** - Define um conjunto de *features* onde todas podem ser escolhidas, apenas uma, algumas ou nenhuma *feature*, é representada por um arco preenchido.

2.4.2 Gerações de Linhas de Produto de Software

A pesquisa na área de linhas de produto de software data do início da década de 90. Desde então, houve avanços em seu desenvolvimento (KRUEGER, 2007) que permitem classificar o desenvolvimento de projetos SPL em três gerações.

- **Primeira Geração.**

  Na primeira geração, o desenvolvimento das linhas de produto de software se concentrava na Engenharia de Aplicação (KRUEGER, 2010) e a equipe de desenvolvedores concentrava-se no desenvolvimento de componentes reutilizáveis. Para criar

um novo produto, as *features* eram selecionadas e a partir desta escolha, os componentes reutilizáveis desenvolvidos eram escolhidos e então se montava um produto para aquela linha.

Entretanto, essa abordagem trazia alguns problemas. Em algumas situações, era necessário criar funções muito específicas para um produto em particular de modo que este componente acabaria sendo utilizado apenas no produto para o qual foi desenvolvido. Isso dificultava sua manutenção e evolução, uma vez que o processo de instanciação de produtos era controlado pelo desenvolvedor. Este problema se espalhava na medida em que novos produtos com funcionalidades únicas eram adicionados ao portfólio do projeto SPL, o que trazia dificuldades também para a manutenção dos componentes reusáveis.

- **Segunda Geração.**

  Os problemas encontrados durante a primeira geração levaram os desenvolvedores de projetos SPL a trabalhar na criação de gerenciadores de configurações de produtos, responsáveis pelo gerenciamento dos componentes reutilizáveis e pela instanciação de produtos da linha.

  Nesta geração, o desenvolvimento de uma SPL assemelhava-se ao desenvolvimento de sistemas legados (KRUEGER, 2007). Os desenvolvedores concentravam-se no desenvolvimento do código-fonte para todas as funcionalidades da linha, levando em consideração aspectos comuns e reusáveis detectados na Engenharia de Domínio e os associavam às *features* correspondentes para a sua devida implementação.

  Para gerar um produto da linha o desenvolvedor precisaria apenas informar quais *features* iriam compor o produto e então o gerenciador de configuração construiria automaticamente o produto, escolhendo os trechos de código-fonte específicos paras as *features* selecionadas. No entanto, o uso de gerenciadores de configurações enfrenta o problema da análise combinatorial, pois, em muitos casos, devido a disposição dos relacionamentos entre *features*, a quantidade de produtos que podem ser gerados por um projeto SPL pode ser enorme (KRUEGER, 2010), sendo que alguns desses produtos não possuem representatividade para a linha, ou seja, não seriam produtos comercializáveis dentro do portfólio.

- **Terceira Geração (Atual).**

  A terceira geração decidiu atacar este problema da análise combinatorial, uma vez que na maioria dos projetos SPL, a quantidade de *features* faz com que a quantidade de produtos ultrapasse os limites do que pode ser testado e mantido pelos desenvolvedores (KRUEGER, 2007). Neste sentido, foram desenvolvidos novos métodos de instanciação de produtos como o T-Wise (HENARD et al., 2014) para minimizar a quantidade de produtos que podem ser gerados por um projeto SPL.

  Dessa forma, a partir da terceira geração, os gerenciadores de configurações passaram a utilizar algoritmos para analisar as possíveis combinações no relacionamento entre as *features*, a fim de identificar o conjunto de configurações mais significativo, reduzindo drasticamente o número de produtos que podem ser gerados.

2.5 ARQUITETURA DE LINHA DE PRODUTOS

Arquitetura de Linha de Produtos (em inglês, *Product Line Architecture - PLA* pode ser definida como a coleção de elementos, relacionamentos e decisões arquiteturais que considera as similaridades e possíveis variabilidades definidas para uma SPL. A PLA é uma arquitetura genérica que deve tornar explícitos elementos arquiteturais para descrição de similaridades e opções de variabilidade da SPL, que serão implementados por componentes reusáveis (LINDEN; SCHMID; ROMMES, 2007). Tais elementos são necessários para instanciação de produtos da linha (LINDEN; SCHMID; ROMMES, 2007).

A PLA deve servir como guia para a geração de todos os produtos da SPL. Para cada produto gerado, a PLA é estendida ou instanciada, de modo que a arquitetura do produto esteja em conformidade com a PLA (LINDEN; SCHMID; ROMMES, 2007; NAKAGAWA; ANTONINO; BECKER, 2011). A arquitetura de software de cada produto gerado incluirá elementos arquiteturais para a descrição de todas similaridades e de um subconjunto das variabilidades da SPL, trazendo benefícios análogos aos de arquitetura de software para sistemas tradicionais.

A arquitetura da SPL deve atender a quatro interesses (LINDEN; SCHMID; ROMMES, 2007):

- **Prover uma descrição abstrata da linha de produtos**. A arquitetura conceitual da SPL deve descrever os seus conceitos-chave, abstraindo detalhes da implementação.
- **Descrever a estrutura da família de produtos**. A arquitetura da SPL deve capturar a decomposição dos sistemas em componentes e seus relacionamentos.
- **Textura -** A arquitetura da SPL deve fornecer um conjunto de regras para que a mesma possa evoluir no decorrer do tempo. Essas regras podem ser convenções de código, padrões de *design* e estilos arquiteturais.
- **Variabilidade e Comunalidade -** A arquitetura de uma linha de produto deve conter também os elementos comuns que estarão presentes na arquitetura de todos os produtos, bem como os elementos variáveis que estarão presentes apenas em alguns produtos.

2.5.1 Documentação de PLA

A documentação de PLA segue os mesmos princípios definidos para a documentação de arquitetura de sistemas tradicionais (CLEMENTS et al., 2010). Adicionalmente, a variabilidade da SPL, considerada parte integrante de qualquer arquitetura de linha de produto de software, deve ser documentada (POHL; BöCKLE; LINDEN, 2005; LINDEN; SCHMID; ROMMES, 2007). Alguns trabalhos propõem diretrizes para a documentação de características de variabilidade e comunalidade presentes na PLA (CLEMENTS et al., 2010).

Em abordagens pro-ativas, as características de variabilidade e comunalidade são identificadas durante a análise de domínio da SPL (Seção 2.4.1) a partir da variabilidade e comunalidade especificadas nos requisitos para os produtos da linha.

Estes novos elementos presentes na documentação da PLA permitem que os arquitetos priorizem novos aspectos no projeto da SPL, por exemplo, flexibilidade (a PLA é uma representação arquitetural que atende a diversos produtos da linha), facilidade de evolução e manutenibilidade (a PLA suporta a novas configurações de produto; a PLA suporta análise para verificar se uma mudança arquitetural causa impactos negativos nos produtos gerados a partir da PLA) (POHL; BöCKLE; LINDEN, 2005).

2.5.2 Avaliação de PLA

A qualidade de uma PLA pode ser avaliada com base no grau de reúso de seus elementos, considerando que, quanto maior for o número de elementos reutilizáveis da PLA, maior será sua qualidade. Nesse contexto, métricas foram propostas para mensurar o grau de reúso dos elementos de uma PLA (ZHANG et al., 2008). Este grau de reúso é calculado em duas dimensões: Através da métrica CRR, é possível calcular o grau de reúso de cada elemento da PLA, sendo que o valor indica a porcentagem de produtos da PLA que contém o elemento analisado. A outra dimensão é calculada através das métricas SSC e SVC que calculam o nível de reúso geral da PLA através da relação entre o elementos comuns e variáveis da PLA como um todo.

Oliveira Junior *et. al.* (2008) propõem um conjunto de métricas para mapear a variabilidade de uma PLA. O foco de tais métricas é o cálculo da quantidade de elementos variáveis presentes nas classes que compõem a PLA e tipo de variabilidade associada.

Neste trabalho, selecionamos um subconjunto das métricas propostas por estes autores para avaliação da PLA recuperada. Estas métricas foram selecionadas com base nas informações coletadas e analisadas pela técnica desenvolvida nessa pesquisa de mestrado, a PLAR. Os dados obtidos pela técnica permitem os cálculos das métricas apresentadas na Tabela 2.2. Para cada métrica, são apresentados uma descrição do cálculo realizado, sua fórmula e o autor da métrica.

2.5.3 Reconstrução da Arquitetura de Linha de Produtos

A principal motivação para a reconstrução da arquitetura de uma linha de produtos é identificar e documentar os elementos comuns e variáveis que serão reutilizados pelos produtos da SPL (EIXELSBERGER, 2000). Além disso, é possível obter os seguintes benefícios com a PLA recuperada (EIXELSBERGER, 2000):

- Suporte a introdução de novos produtos na SPL;
- Redução na complexidade associada ao desenvolvimento e manutenção de código-fonte;
- Aumento na qualidade dos produtos lançados; e
- Suporte ao compartilhamento de componentes e conhecimento acerca da SPL.

Ducasse e Pollet (DUCASSE; POLLET, 2009) sugerem que a recuperação da arquitetura dos produtos de uma família permite a identificação de seus elementos comuns

Tabela 2.2 Métricas de Avaliação de PLA.

Métrica	Descrição	Fórmula	Referência
SSC	SSC calcula a similaridade geral entre os elementos da PLA.	$\frac{\lvert C_C \rvert}{\lvert C_C \rvert + \lvert C_V \rvert}$	[1]
SVC	Calcula a variabilidade geral entre os elementos da PLA	$\frac{\lvert C_V \rvert}{\lvert C_C \rvert + \lvert C_V \rvert}$	[1]
CRR	Calcula o coeficiente de reúso de cada elemento da PLA	$\frac{\sum_i Ex(M_i)}{\lvert M \rvert} \times 100\%$	[1]
ClassVP	Indica se determinada classe é um ponto de variabilidade		[2]
ClassOptional	Calcula o número de classes opcionais	$\sum C_V$	[2]
ClassMandatory	Calcula o número de classes mandatórias	$\sum C_C$	[2]
PLTotalVariability	Estima o número total de variabilidade encontrada na PLA	$\sum RC_V + \sum C_V$	[2]
ComponentVariable	Indica se determinado componente apresenta algum tipo de variabilidade		[2]
Legenda Cc **-** Total de componentes comuns Cv **-** Total de componentes variáveis M **-** Total de produtos da SPL RCv - Total de relacionamentos variáveis $Ex(M_i)$ **-** Determina se o elemento está presente ou não na Arquitetura de um produto, assume o valor 1 para presença e 0 para ausência			
Referências - [1] (ZHANG et al., 2008) [2] (JUNIOR; GIMENES; MALDONADO, 2008)			

e variáveis, alcançando um resultado arquitetural próximo ao de uma PLA, através da identificação de potenciais componentes de reúso, além de facilitar a migração da família de produtos para o paradigma SPL.

Nesse contexto, algumas abordagens para recuperação de PLA foram propostas (EIXELSBERGER et al., 1996, 1998a; SHATNAWI; SERIAI; SAHRAOUI, 2016).

Eixelsberger propôs uma técnica híbrida para recuperação de famílias de programas de software (EIXELSBERGER et al., 1996, 1998a; EIXELSBERGER, 2000). A abordagem proposta requer o código fonte e a documentação dos programas, e um engenheiro de software responsável pelo sistema para reconstruir a arquitetura dos programas e assim, recuperar a arquitetura da linha de produtos (Figura 2.8).

Shatnawi (SHATNAWI; SERIAI; SAHRAOUI, 2016, 2014) apresenta ROMANTIC, uma abordagem de recuperação de PLA que considera diferentes versões de um mesmo projeto SPL. Nesta abordagem a recuperação da PLA é feita em duas etapas: recuperação da arquitetura de produtos e identificação de elementos em comum e variabilidade. Cada versão da SPL tem sua arquitetura recuperada. Posteriormente, estas arquiteturas são

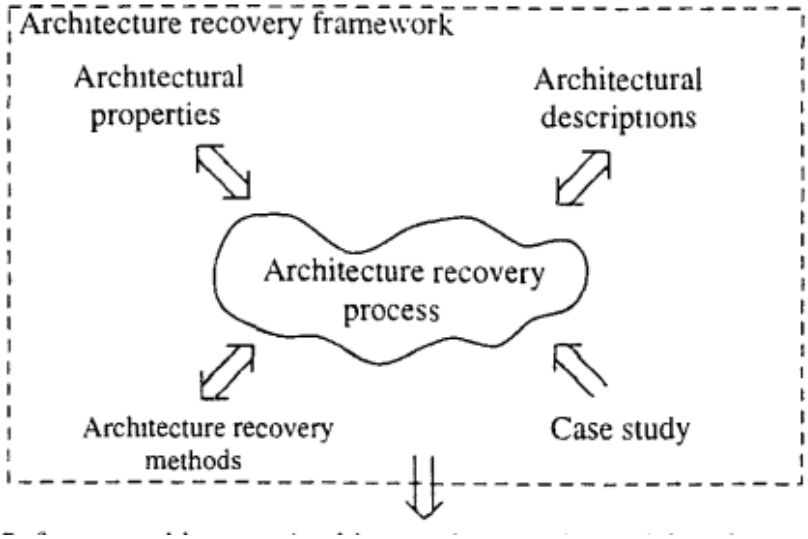

Figura 2.8 Processo de recuperação proposto por Eixelsberger(EIXELSBERGER et al., 1996)

analisadas em conjunto, para que sejam identificados elementos em comum e pontos de variação entre produtos. Por fim uma visão componente-conector da PLA é produzida com base nas informações obtidas.

Em resumo, com base na revisão da literatura realizada, as abordagens para reconstrução de PLA encontradas dependem da recuperação de arquitetura dos produtos da SPL ou de diferentes versões da mesma. Não foram encontradas técnicas ou ferramentas específicas para recuperação de PLA diretamente do código fonte da SPL, sendo necessário instanciar produtos e/ou utilizar diferentes versões para realizar a recuperação da PLA a partir destas fontes de informação.

2.6 FERRAMENTAS DE RECUPERAÇÃO DE ARQUITETURA DE LINHAS DE PRODUTO

Esta seção apresenta ferramentas de reconstrução de PLA, selecionadas com base nos resultados de um *survey* conduzido para identificar abordagens de reconstrução de arquitetura de software no contexto de linhas de produto de software (LIMA-NETO et al., 2015). A Tabela 2.3 apresenta detalhes das ferramentas selecionadas e da ferramenta PLAR Tool proposta neste trabalho, nesta tabela estão considerados os objetivos de cada ferramenta, quais visões arquiteturais são produzidas, qual linguagem de programação é suportada na recuperação e a disponibilidade da ferramenta.

A principal dificuldade foi encontrar uma versão das ferramentas utilizadas nas abordagens. Entramos em contato com os desenvolvedores porém não obtivemos resposta.

Outro ponto importante é que não há um consenso sobre qual visualização da PLA deve ser utilizada; cada ferramenta/abordagem adotou uma visualização diferente para representação da PLA recuperada. Tais informações foram consideradas durante o desenvolvimento da ferramenta PLAR Tool.

Tabela 2.3 Ferramentas de Recuperação de PLA

Ferramenta/Abordagem	Objetivo	Visão Gerada	LP	Disponibilidade
ROMANTIC	Recuperar a PLA	Componente-Conector	Java	Sob Consulta
Linsbauer	Recuperar o modelo de features	Diagrama de Feature	Java/C	Sob Consulta
Eixelsberger	Adaptar família de software para SPL	ND	ND	Não há apoio ferramental
RECoVar	Melhorar atividades relacionadas a variabilidade em SPL evolutiva	Árvore de Variação	C/C++	Sob Consulta
Legenda	[ND] - Não Disponível	[LP] - Linguagem de Programação		

2.6.1 ROMANTIC

O objetivo da ferramenta ROMANTIC é a recuperação da PLA de uma SPL a partir do código-fonte de um conjunto de produtos da SPL. A arquitetura de cada produto é extraída e a variabilidade entre estas arquiteturas é identificada e mapeada. A ferramenta recupera a visão arquitetural componente-conector a partir do código fonte dos produtos, se limitando apenas a analisar projetos SPL escritos em linguagens de programação que oferecem suporte ao paradigma de orientação a objetos (SHATNAWI; SERIAI; SAHRAOUI, 2016).

O processo de recuperação da PLA está divido em quatro etapas (Figura 2.9): (i) Extração da arquitetura de cada produto, (ii) Identificação das variantes arquiteturais, (iii) Identificação da variabilidade e comunalidade, e (iv) Identificação da variabilidade das dependências arquiteturais.

A extração da arquitetura dos produtos é feita utilizando uma ferramenta proprietária de extração de dependências, com o resultado desta extração a ferramenta busca encontrar entre os diversos produtos, componentes arquiteturais que fornecem serviços semelhantes. Em seguida a variabilidade interna de cada componente é analisada. Após fazer este levantamento, a ferramenta ROMANTIC usa algoritmos de FCA (*Formal Concept Analysis*) para identificar o que é comum e variável dentro da PLA a ser construída.

A técnica FCA[2] (GANTER; GODIN, 2005) permite a análise de um conjunto de relacionamentos entre objetos, descritos por um conjunto de atributos. Nesta técnica, os objetos que compartilham os mesmos atributos são chamados de *conceitos formais*. Estes conceitos formais são extraídos e organizados de forma hierárquica em um grafo chamado de *concept lattice*. Por fim os relacionamentos entre estes componentes são classificados de acordo com os relacionamentos das *features* que implementam os componentes arquiteturais identificados.

2.6.2 Abordagem proposta por Linsbauer

Linsbauer et. al. (2016) propõe uma abordagem incremental, *bottom-up*, para recuperação automática do modelo de *features* a partir do código-fonte dos produtos de uma SPL. Sua técnica de recuperação é dividida em duas etapas:

[2] *Formal Concept Analysis*

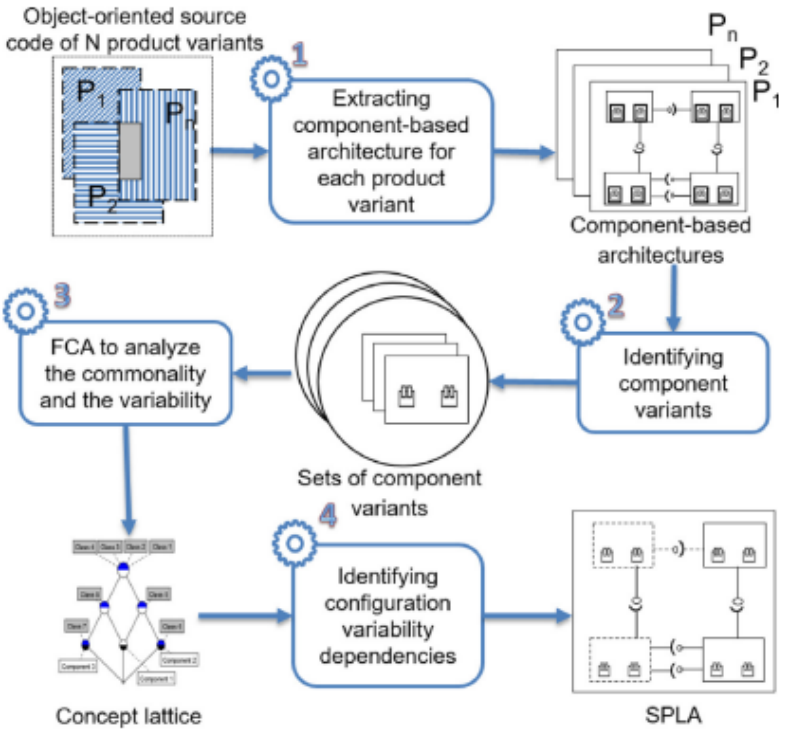

Figura 2.9 Processo de recuperação feito pelo ROMANTIC (SHATNAWI; SERIAI; SAHRAOUI, 2016)

- **Extração de "rastros":** Nesta primeira etapa o código fonte de cada produto é analisado em busca de traços de implementação das *features*, que são os trechos do código fonte responsáveis por sua implementação. Ao término da análise, os traços encontrados servem de entrada para próxima etapa da técnica.

- **Extração de dependências:** Com os traços coletados, o próximo passo da técnica é analisar as dependências entre estes traços e organizá-los de forma que, ao final deste processo, se obtenha o diagrama de *features* da SPL analisada.

Os produtos são analisados por uma ferramenta proprietária de nome não divulgado que analisa o código fonte, em busca da implementação das *features* do projeto SPL em análise. Os produtos são analisados 1 a 1; no primeiro produto, todos os rastros (*trace*), que são os trechos de código fonte responsáveis pela implementação das *features*, são armazenados.

Do segundo produto em diante, os rastros já armazenados são comparados com os novos rastros encontrados, a fim de encontrar os comuns e variáveis entre os diversos produtos, até que todos os produtos fornecidos para análise sejam analisados (LINSBAUER; LOPEZ-HERREJON; EGYED, 2016). Ao final deste processo o diagrama de *features* é reconstruído, a Figura 2.10 mostra um exemplo de um diagrama de *features* reconstruído.

Neste diagrama simplificado não há informações sobre o tipo de dependências entre as *features*. Todas as *features* mandatórias (as que foram identificadas como comuns a todos

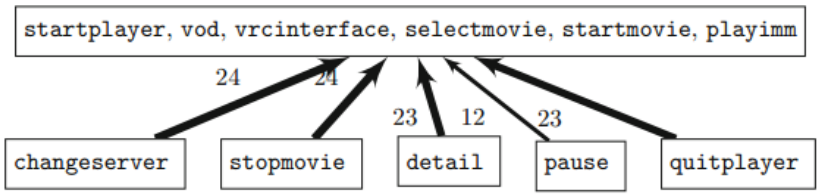

Figura 2.10 Diagrama de *features* recuperado (LINSBAUER; LOPEZ-HERREJON; EGYED, 2016)

os produtos) ficam condensadas em um único retângulo, enquanto que as opcionais ficam isoladas. Além disso, o diagrama recuperado também mapeia o nível da dependência entre estes elementos por meio das setas de diferentes densidades e da numeração que indica o nível de acoplamento entre os grupos de *features*.

2.6.3 Abordagem proposta por Eixelsberger

A abordagem proposta por Eixelsberger et. al. (1998) utiliza o código fonte de uma família de produtos, constituída de sistemas legados, aliado a documentação disponível para construir uma arquitetura comum (EIXELSBERGER et al., 1998b) aos produtos desta família. Posteriormente esta abordagem foi aplicada ao domínio de SPL (EIXELSBERGER, 2000).

A abordagem se apoia no uso de diversas visões arquiteturais dos sistemas, recuperadas a partir de ferramentas de recuperação e da documentação disponível. Com estas visões arquiteturais, o desenvolvedor deverá realizar uma análise em busca dos componentes variáveis e comuns a todos os sistemas da família, a fim de construir uma arquitetura comum a todos os produtos.

Nesta abordagem a arquitetura de cada produto é recuperada por diversas ferramentas e/ou abordagens de extração a fim de garantir uma ampla variedade de visões arquiteturais. A(s) arquitetura(s) recuperada(s) são então comparadas com a documentação disponível a fim de analisar a conformidade, neste processo os elementos não mapeados pela documentação são adicionados no documento final que contém a arquitetura da família de produtos de acordo com o que foi implementado em código fonte. Na extensão desta abordagem para o contexto de SPL, foi adicionado a este processo de verificação, mais uma atividade que é a de identificar os elementos que são comuns a todos os produtos e os que são variáveis para construir a PLA a partir dessas informações. Esta abordagem entretanto não se apoia em nenhuma ferramenta neste processo de comparação da arquitetura recuperada com a documentação, sendo este processo feito pelo responsável pela recuperação, tornando assim esta abordagem semi-automática.

2.6.4 RECoVar

Assim como a abordagem proposta por Linsbauer *et. al.* (2016) o *framework* RECoVar analisa o código fonte do projeto SPL e a partir dele, encontrar os pontos de variação. A diferença entre as duas abordagens está no fato de que enquanto a RECoVar busca

os pontos de variação implementados em código, a abordagem de linsbauer tenta extrair destes pontos de variação o modelo de *features* da linha.

O *framework* possui uma limitação, que é lidar apenar com projetos cuja a variabilidade foi implementada a partir de compilação condicional (CC). O principal objetivo deste *framework* é melhorar as atividades relacionadas com variabilidade no contexto de SPL evolutivas (ou SPLs em evolução). As atividades incluem, configuração, manutenção e garantia da qualidade da SPL (ZHANG; BECKER, 2012) através da recuperação da árvore de variação do projeto.

A técnica proposta pela RECoVAR é baseada na análise do código fonte dos produtos gerados por uma SPL, em específico na identificação das diretivas de compilação condicional (*#ifdefs*) encontradas no código. Com a análise destas diretivas, a técnica consegue recuperar uma árvore de variabilidade que representa toda a variabilidade implementada no código fonte.

Para realizar a recuperação dos pontos de variação da linha, é utilizado um *parser* cuja função é identificar as marcações de compilação condicional no código (neste caso específico as marcações : *#ifdef*) e a partir destas marcações identificar a variabilidade implementada em código-fonte.

Após a execução do *parser*, uma árvore de variabilidade contendo todas as marcações encontradas é gerada (Fig 2.6.4), como passos de verificação da validade dessa árvore, são analisados o código dos produtos gerados bem como a opinião de desenvolvedores (ZHANG; BECKER, 2012, 2013)

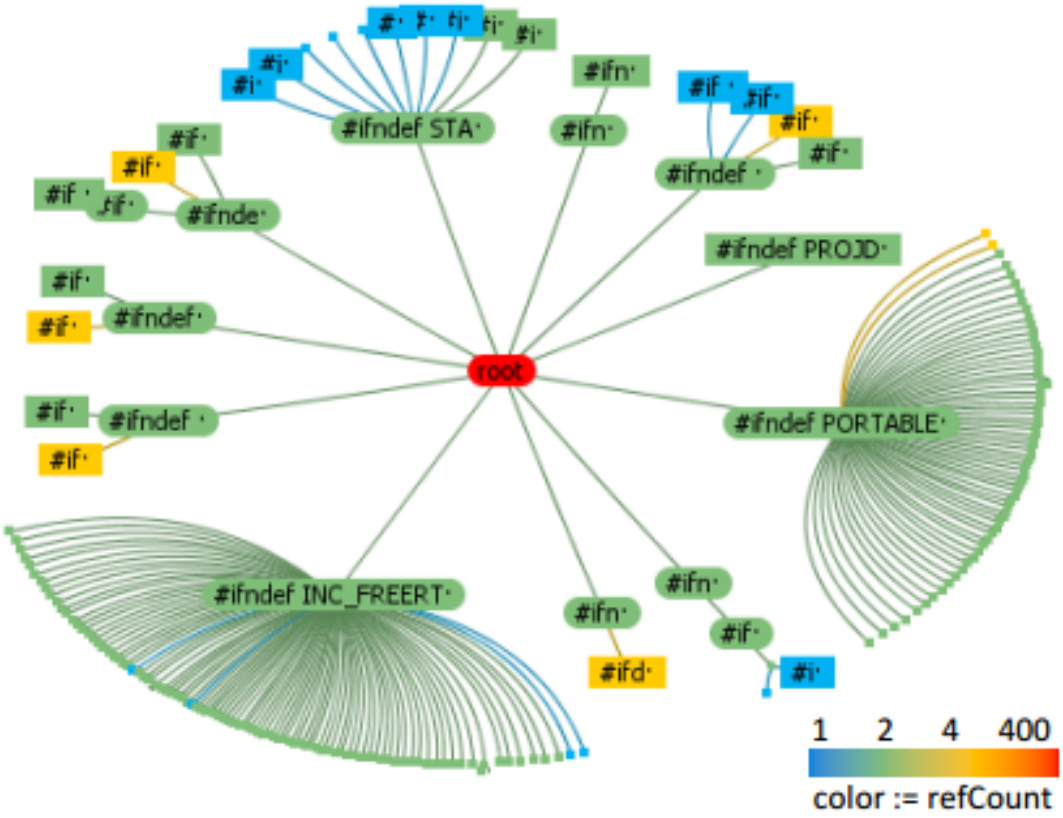

Figura 2.11 Árvore de Variabilidade gerada pelo *framework* RECoVar (ZHANG; BECKER, 2012)

Capítulo 3

A TÉCNICA PLAR

Este capítulo apresenta a ferramenta PLAR Tool, uma implementação e prova de conceito da técnica PLAR para recuperação de arquitetura de linha de produtos proposta neste trabalho.

A seção 3.1 apresenta a técnica PLAR e comparação com trabalhos relacionados. A seção 3.2 apresenta a ferramenta PLAR Tool, uma implementação e prova de conceito para a técnica PLAR. A seção 3.2.3 apresenta a arquitetura da ferramenta e seus principais módulos. A seção 3.2.4 apresenta detalhes sobre a implementação da recuperação de PLA. As seções seguintes apresentam detalhes sobre o suporte dado pela ferramenta para avaliação da qualidade de PLA (seção 3.2.5), e geração de visões arquiteturais (seção 3.2.6). A seção 3.2.7 apresenta uma síntese das características técnicas da PLAR Tool. A seção 3.2.8 discute as as limitações da ferramenta quanto a recuperação da PLA de projetos SPL. Finalmente, uma comparação da ferramenta a PLAR Tool com ferramentas relacionadas é apresentada (seção 3.2.10).

3.1 A TÉCNICA PLAR PARA RECUPERAÇÃO DE PLA

PLAR é uma técnica *quasi*-automática para apoiar o processo de reconstrução *bottom-up* de arquitetura de linhas de produtos (PLA) a partir de um conjunto de produtos de software[1]. A arquitetura da linha de produtos recuperada, com elementos e relações arquiteturais comuns e variáveis, é a principal saída da aplicação da técnica de recuperação PLAR.

São decisões associadas à técnica proposta:

- Independência de linguagem de programação - A recuperação da PLA é baseada nas dependências encontradas no código fonte dos produtos gerados pela SPL. Existem diversas ferramentas que extraem dependências de código fonte em diversas linguagens de programação. A técnica PLAR se apoia no uso de arquivos no formato MDG[2]. Isto dá a técnica uma independência de linguagem de programação uma vez

[1] A reconstrução *bottom-up* de PLA será tratada no restante do texto da dissertação como *recuperação de PLA*.

[2] *Module Dependency Graph*

que a extração é baseada neste arquivo de dependência extraído e não no código fonte do projeto.

- Seleção de subconjunto de produtos da SPL - A fim de recuperar uma PLA completa, isto é, que contém todos os elementos comuns e variáveis da mesma, um subconjunto de produtos representativos deve ser selecionado previamente a fim de evitar a omissão de elementos da PLA, este subconjunto de produtos deve obedecer dois critérios:
  - Precisão - A seleção dos produtos deve permitir a classificação de todos os elementos comuns e variáveis devem ser mapeados corretamente.
  - Abrangência - A seleção de produtos deve permitir que a PLA recuperada pela técnica PLAR identifique todos os elementos que podem estar presentes em todos os produtos.

A técnica PLAR está organizada em duas etapas: Seleção de Produtos da SPL e Recuperação da PLA.

3.1.1 Seleção de produtos da SPL

A fim de garantir que todos os elementos (módulos / relações) estarão presentes na PLA é necessário selecionar um subconjunto de produtos que seja representativo a fim de evitar que elementos não sejam mapeados. Nesta dissertação utilizados dois métodos para geração de produtos, um método que chamamos de convencional e o método T-Wise (HENARD et al., 2014).

O método convencional consiste em selecionar todos os produtos que podem ser gerados a partir de uma configuração válida. Uma configuração é dita válida quando a seleção das *features* que irão compor o produto obedecem as restrições de relacionamento estabelecidas no diagrama de *features* (APEL et al., 2013). Ao selecionar o método convencional garantimos a precisão e abrangência da recuperação, uma vez que ao utilizar todos os produtos na recuperação, não haverá perda de informação.

O método T-Wise (HENARD et al., 2014) foi concebido com o objetivo de reduzir o impacto da combinação das *features* na geração de produtos. A depender da quantidade de *features* de um projeto SPL, o número de produtos que pode ser gerado a partir de uma configuração válida torna-se grande o suficiente para prejudicar e inviabilizar diversos tipos de análise que podem ser feitas nos projetos SPL. O método T-Wise consiste em selecionar as configurações mais significativas entre as possíveis para construção dos produtos, dessa forma minimizando drasticamente o número de produtos que serão utilizados na recuperação. O estudo apresentado no capítulo 6 aborda sobre o impacto da seleção de produtos na recuperação da PLA e compara as recuperações realizadas pelos dois métodos de geração de produtos.

Após a seleção dos produtos, cada produto deve ser instanciado individualmente. O código-fonte de cada produto será então analisado e as dependências encontradas no código fonte devem ser extraídas. Estas dependências compreendem as classes (módulos) e suas relações. O arquivo que contém as informações extraídas é chamado de MDG,

para cada produto teremos um arquivo MDG que representa esta arquitetura extraída. Estes arquivos MDG são utilizados como entrada para a próxima etapa:Recuperação da PLA.

3.1.2 Recuperação da PLA

A recuperação da PLA contempla a análise dos produtos selecionados e a síntese da arquitetura de linha de produtos. O Algoritmo 1 apresenta os passos usados pela técnica de recuperação de PLA proposta.

```
   Entrada: P, uma coleção de N produtos, N > 1, onde cada produto é
            representado por um MDG.
   Saída: PLA, uma coleção de elementos e relações arquiteturais comuns e variáveis.
 1 início
 2   para cada produto P_i em P faça
 3     para cada elemento e ou a relação r encontrado no produto P_i faça
 4       se o elemento e ou a relação r já tiver sido identificado em outro
           produto P_j, i ≠ j então
 5         Adicione +1 ao número de ocorrências do elemento e ou relação r;
 6         Atualize a informação sobre os produtos que contêm o elemento e ou
             relação r;
 7       fim
 8       se o elemento e ou a relação r não estiver presente então
 9         Insira o elemento e ou a relação r na coleção PLA;
10         Marque o elemento e ou relação r como identificado;
11       fim
12     fim
13   fim
14   para cada elemento e ou relação r catalogado em PLA faça
15     se o elemento e ou a relação r estiver em todos os produtos então
16       Marque o elemento e ou a relação r como "comum";
17     senão
18       Marque o elemento e ou a relação r como "variável";
19     fim
20   fim
21 fim
```

Algoritmo 1: Algoritmo de recuperação da PLA

O algoritmo recebe como entrada o conjunto de produtos selecionados na etapa anterior. Inicialmente, os produtos gerados pela SPL são analisados, individualmente, para identificação de elementos arquiteturais. Esses elementos podem ser classes, interfaces, classes abstratas, entre outros. Durante a análise de cada produto, os elementos (módulos e relações) são identificados um a um. Para cada produto analisado. Se um elemento ainda não tiver sido catalogado, ele será mapeado na coleção de elementos da PLA. Caso

contrário, a ferramenta contabiliza o número de ocorrências do elementos nos produtos a fim de verificar posteriormente se este elemento é comum ou variável a todos os produtos inseridos para realização do processo de recuperação. (algoritmo 1, linhas 2–13).

Com estas informações catalogadas, a técnica inicia o processo de identificação de elementos e relações comuns e variáveis (algoritmo 1, linhas 14–20). Cada elemento será analisado de acordo com o número de ocorrências detectadas na etapa anterior. Se este número for igual ao número de produtos inseridos para análise este elemento é classificado como "comum"caso contrário é classificado como "variável".

Uma das vantagens da técnica PLAR é que a mesma depende apenas dos produtos implementados e de seu código fonte, sendo independente em relação ao tipo de gerador de produtos usado e à maneira como cada gerador gerencia a variabilidade dentro do código fonte da SPL. Isto permite que seja possível a análise de um maior número de projetos SPL além de contemplar todas as gerações.

Algumas técnicas de reconstrução de PLA foram selecionadas como trabalhos relacionados, com base nos resultados de um *survey* conduzido para identificar abordagens de reconstrução de arquitetura de software no contexto de linhas de produto de software (LIMA-NETO et al., 2015). Elas se diferenciam da técnica proposta em diversos aspectos, estas técnicas estão detalhadas no seção 2.5.3. A seção 3.2.10 apresenta um comparativo da Técnica PLAR com as demais técnicas relacionadas.

3.2 VISÃO GERAL DA PLAR TOOL

PLAR Tool é uma ferramenta para recuperação de PLA que automatiza e serve como prova de conceito para a técnica PLAR (Seção 3.1). A PLAR Tool recebe como entrada um conjunto de arquivos MDG que representam os produtos de uma SPL e gera como saída a PLA recuperada para a SPL, em diferentes formatos (Figura 3.1).

A PLA recuperada contém informações sobre módulos e relacionamentos usados na descrição de arquitetura de sistemas tradicionais, e informações sobre comunalidade e variabilidades específicas para a descrição de arquiteturas de linhas de produtos.

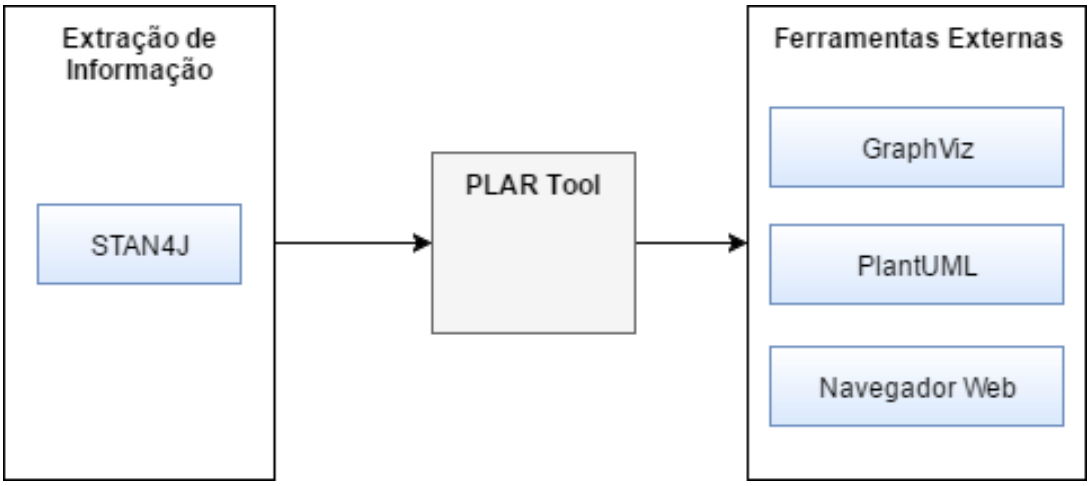

Figura 3.1 A ferramenta PLAR Tool

3.2.1 Contexto de Uso

Duas etapas precedem a recuperação de PLA com apoio da ferramenta PLAR Tool: a geração de produtos de uma SPL e a extração do grafo de dependências entre módulos para cada produto gerado. Os produtos da SPL, representados em formato MDG, servirão de entrada para a PLAR Tool.

1. **Geração de Produtos**

   A geração de produtos é feita com apoio de um plugin da plataforma Eclipse que suporta a criação de projetos SPL*"feature-oriented"*, o FeatureIDE[3].

   No FeatureIDE, cada projeto SPL possui um arquivo de configuração, que contém a relação de todas as *features* que compõe a SPL. Há três opções para geração de produtos: criar uma configuração e gerar um produto a partir desta configuração; gerar todos os produtos possíveis baseados em configurações válidas; ou gerar todos os produtos com as configurações mais significativas com base no método de T-Wise.

   Neste trabalho, utilizamos os métodos de gerar todas as configurações válidas e o método de geração T-Wise nos estudos realizados.

2. **Extração de MDG**

   Para cada produto gerado na etapa de Geração de Produtos, um novo projeto do Eclipse é criado. Dessa forma, os produtos de uma SPL são tratados como *single-systems*.

   A extração do MDG de um produto pode ser realizada a partir da análise seu código fonte. Diferentes ferramentas de análise e extração podem ser usadas, tais como, STAN4J, Struct101[4] e Doxygen[5].

   Para cada produto da SPL, um arquivo MDG que representa a sua arquitetura é extraído. Em nosso trabalho, utilizamos a ferramenta Macro Creator[6] para automatizar o processo de extração de MDGs para todos os produtos da SPL.

3.2.2 Funcionalidades

São três as principais funcionalidades[7] da ferramenta PLAR Tool:

1. **Recuperação da PLA**

   A principal funcionalidade da ferramenta é a recuperação de PLA a partir de um conjunto de produtos de uma SPL, com identificação de elementos comuns e variáveis da PLA. A técnica de recuperação implementada pela ferramenta foi apresentada na seção 3.1.

[3]http://v.ht/FeatureIDE
[4]http://structure101.com/
[5]http://www.stack.nl/ dimitri/doxygen/
[6]http://www.macrocreator.com/
[7]As funcionalidades da ferramenta são detalhadas na Seção 3.2.3.

2. **Apoio à Avaliação da Qualidade da PLA**

   Métricas podem ser calculadas tendo como base a PLA recuperada. No momento, a ferramenta PLAR Tool calcula o grau de reúso dos componentes da PLA. A Tabela 2.2 apresentou um sumário sobre as métricas implementadas pela ferramenta que podem ser usadas na avaliação da qualidade da PLA.

3. **Apoio à Visualização da PLA recuperada**

   A ferramenta PLAR Tool pode gerar representações da PLA recuperada em diferentes formatos, que servem como entrada para ferramentas de visualização. As representações geradas pela ferramenta permitem a criação de visões de módulo da PLA, destacando a comunalidade e a variabilidade entre dois produtos, de matrizes de estrutura de design (DSM) informações sobre elementos comuns e variáveis, e diagramas de classes, decorados com informações sobre a variabilidade.

3.2.3 Arquitetura e Módulos

A Figura 3.2 apresenta uma visão de módulos que descreve a arquitetura da PLAR Tool. A visão de módulos apresenta quatro módulos e seus relacionamentos responsáveis pela implementação de suas funcionalidades.

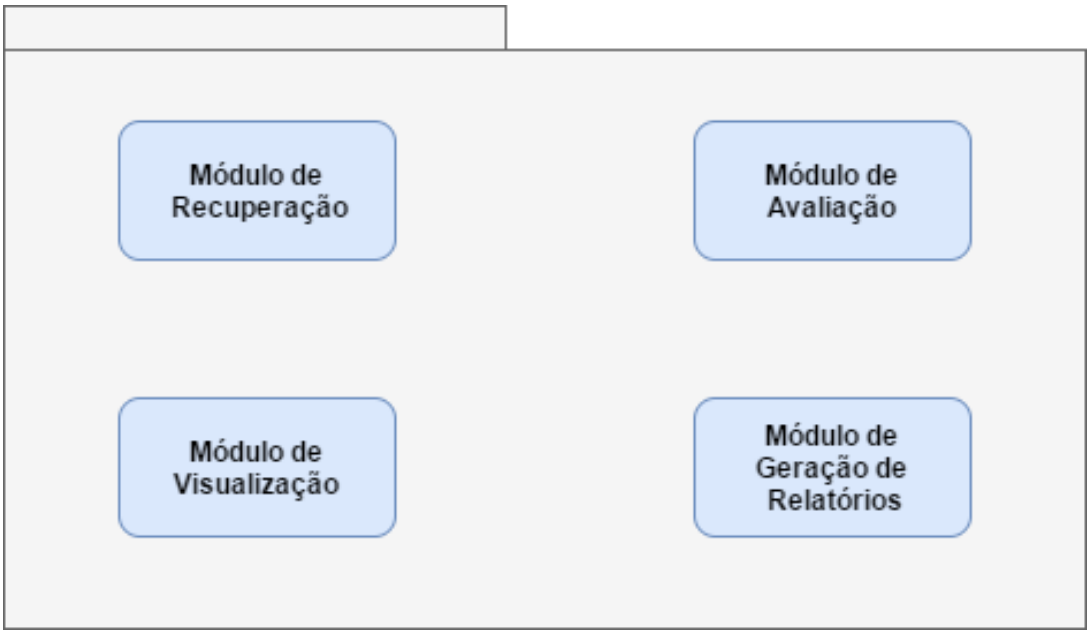

Figura 3.2 Arquitetura da PLAR Tool

Módulo de Recuperação. O módulo de Recuperação é responsável pela implementação da técnica de recuperação PLAR. Os algoritmos contidos neste módulo são responsáveis por analisar diversos produtos da SPL (representados por grafos de dependências), identificar os elementos comuns e variáveis de cada produto e gerar a PLA recuperada.

Módulo de Suporte à Avaliação. Responsável pelo cálculo das métricas suportadas pela ferramenta, utiliza os resultados obtidos pelo módulo de recuperação para realizar sua função.

Módulo de Suporte à Visualização. Responsável por gerar as diferentes visualizações da PLA - Visão de Módulo, diagrama de classes e a DSM.

Módulo de Geração de Relatórios. Responsável por gerar o relatório com o resultado das métricas calculadas pelo módulo de avaliação e um relatório que informa em quais produtos é possível encontrar cada elemento mapeado na PLA.

Nas seções seguintes, apresentamos o detalhamento dos módulos.

3.2.4 Recuperação da PLA

A Figura 3.3 apresenta mais detalhes sobre o módulo de recuperação.

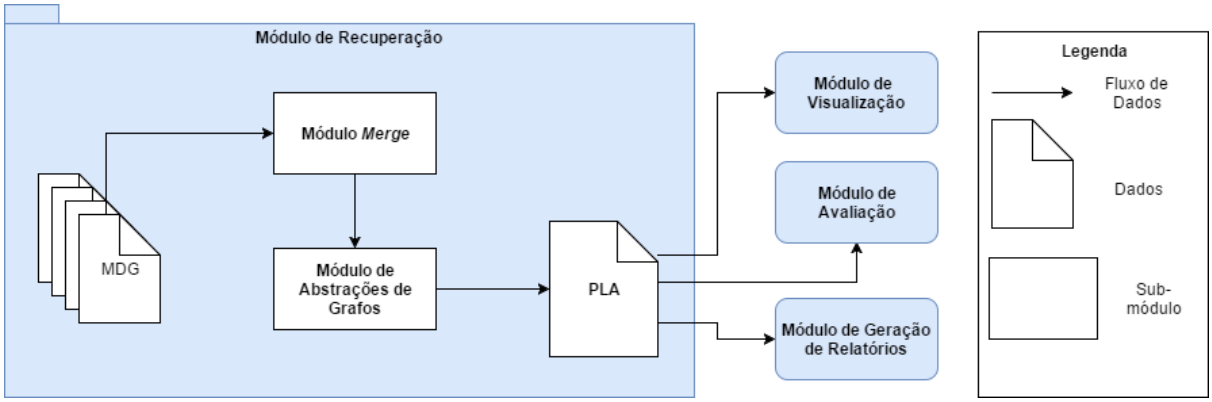

Figura 3.3 Módulo de Recuperação PLAR Tool

O módulo de recuperação da PLA recebe como entrada um conjunto de arquivos MDG, sendo um por produto. O módulo de recuperação possui dois submódulos sendo que o principal módulo é o *Merge*, responsável por interpretar o MDG (*Module Dependency Graph*) dos produtos de uma SPL e armazenar a informação obtida para o módulo de abstrações de grafos possa utilizá-lo para gerar a base de dados da PLA que será utilizada pelos demais módulos da ferramenta.

Atualmente são aceitos como entrada válida apenas arquivos MDG gerados pela ferramenta STAN4J[8]. O arquivo MDG gerado pelo STAN4J é um arquivo de texto que apresenta as classes de um sistema de software e suas relações de forma ordenada.

O módulo *Merge* lê e analisa cada arquivo MDG e, a partir das informações coletadas, gera a PLA do projeto SPL, evidenciando os elementos e relações comuns e variáveis.

Ao término do processo, o módulo de *Merge* fornece a informação compilada para o módulo `Abstrações de Grafos`, o qual transforma a informação obtida pelo módulo

[8]http://stan4j.com/

de *Merge* em uma representação de grafos que será utilizada pelos demais módulos da ferramenta.

O módulo de Geração de Relatórios é responsável pela geração automática de relatórios com informações detalhadas sobre o processo de recuperação da PLA seus resultados. No momento este módulo produz apenas o log com os resultados do processo de recuperação e o relatório com o resultado das métricas coletadas pelo módulo de avaliação como demonstrado na Figura 3.4

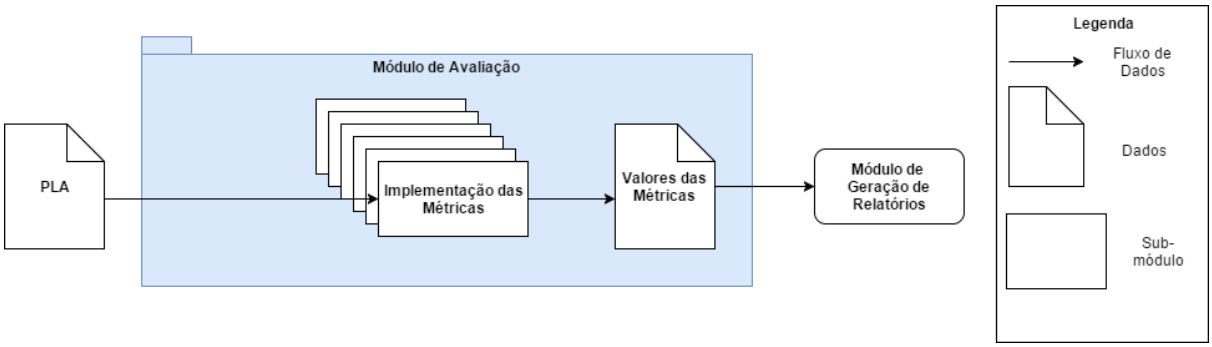

Figura 3.4 Módulo de Avaliação

O Relatório de Inspeção tem como objetivo apresentar os elementos e relações encontrados durante a recuperação da PLA, e os produtos da SPL que os contêm. Este relatório é apresentado ao final do processo de recuperação.

A Figura 3.5 apresenta um exemplo de relatório de inspeção gerado pela ferramenta, que mostra que os três elementos `Client`, `Email` e `Util` estão presentes nos produtos `P1`, `P10`, `P11` e `P12`.

Figura 3.5 PLAR Tool: Relatório de Inspeção

3.2.5 Avaliação de PLA

O módulo de Suporte à Avaliação de PLA é responsável por calcular as métricas relacionadas a qualidade da PLA recuperada. Uma PLA pode ser avaliada qualitativamente

de acordo com o grau de reúso de seus componentes (ZHANG et al., 2008), de forma que quanto maior for o grau de reúso dos seus componentes, melhor será a qualidade da PLA.

Esta característica reflete diretamente em aspectos importantes do ciclo de vida de um projeto SPL como a manutenção e evolução dos projetos. Componentes com alto grau de reúso estão presentes em grande parte dos produtos da linha, estes componentes, caso alterados, geram impactos em boa parte do portfólio da SPL. Da mesma forma, melhorias em um componente da PLA com um alto nível de reúso são refletidas em uma grande quantidade de produtos.

Desta forma, componentes com baixo grau de reúso geram problemas para a manutenção da SPL, pelo fato de serem utilizados em poucos produtos. Nem sempre sua manutenção é priorizada e ter uma equipe de desenvolvimento dedicada para tratar este componente com baixo reúso pode não ser viável (JUNIOR; GIMENES; MALDONADO, 2008).

Neste trabalho, implementamos as métricas propostas por Zhang *et. al.* (ZHANG et al., 2008) e Oliveira Junior *et. al* (JUNIOR; GIMENES; MALDONADO, 2008) para mensurar o grau de reúso dos componentes de uma PLA. As métricas foram apresentadas no capítulo 2 (Tabela 2.2).

3.2.6 Visualização da PLA

O módulo de Suporte à Visualização da PLA é responsável pela geração de arquivos de saída com a representação da PLA recuperada em diferentes formatos. Os arquivos de saída poderão ser utilizados por ferramentas externas para visualização da PLA recuperada.

A versão atual da ferramenta suporta os formatos dot (serve como entrada para GraphViz[9], html (serve como entrada para browsers) e arquivos de texto que obedecem a sintaxe proposta pela ferramenta PlantUML.

3.2.6.1 Visão da diferença entre dois produtos A técnica de recuperação da PLA, quando usada com apenas dois produtos, permite que o resultado seja usado para comparação e visualização de suas semelhanças e diferenças. A Figura 3.6 apresenta um exemplo da nova visão de módulo gerada, que coloca em perspectiva apenas dois produtos para análise.

Os elementos em comum entre os dois produtos estão destacados em azul e linhas cheias. Os elementos variáveis são apresentados apresentados na cor vermelha separados em dois grupos, um para os elementos exclusivos do primeiro produto inserido para análise e outro para os elementos exclusivos do segundo produto inserido para análise.

Esta visualização tem como propósito comparar a arquitetura de dois produtos e identificar as particularidades de cada um, uma vez que este nível de detalhe não está disponível na recuperação da PLA utilizando uma quantidade de produtos maior do que dois.

[9] www.graphviz.org/

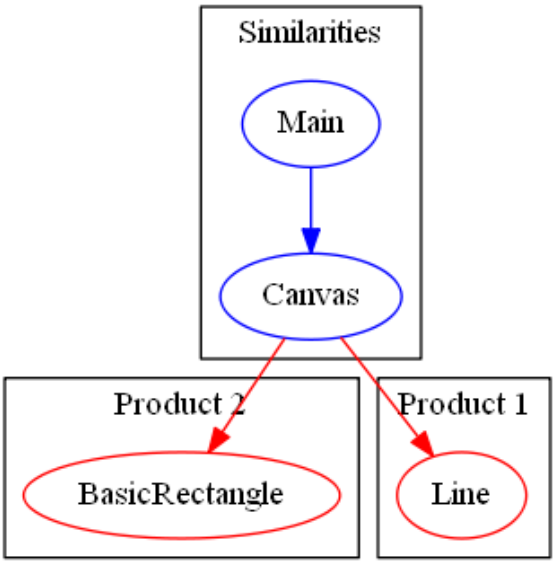

Figura 3.6 PLAR Tool: Diferença entre dois produtos.

3.2.6.2 Visão de Módulos da PLA recuperada A ferramenta PLAR Tool gera três tipos de representação para a PLA recuperada a partir de um número arbitrário de produtos. Tais representações são consideradas visões de módulos complementares, que evidenciam os módulos e relacionamentos da PLA decorados com diferentes cores para destacar a variabilidade.

O primeiro tipo de representação é um grafo de dependências entre módulos, decorado com informações sobre elementos e relações arquiteturais comuns e variáveis, no formato DOT. O segundo tipo de representação é uma matriz de estrutura de design (DSM), decorada com as mesmas informações sobre variabilidade, no formato HTML. O terceiro tipo de representação é um diagrama de classes, decorado com as mesmas informações sobre variabilidade.

Grafo de Dependências entre Módulos. A Figura 3.7 apresenta um exemplo de PLA recuperada, representada em um grafo de dependências decorado. Os elementos e relações da PLA em azul são comuns a todos os produtos e elementos e relações em vermelho implementam a variabilidade. Elementos azuis são ligados a elementos vermelhos por relações em vermelho. Este tipo de relação só pode ser recuperado a partir do código fonte dos produtos.

Design Structure Matrix. A Figura 3.8 apresenta um exemplo de DSM gerada pela ferramenta.

No decorrer do desenvolvimento da ferramenta percebemos que a depender do tamanho da PLA recuperada a visualização em forma de grafo dificulta a identificação dos elementos. Devido a este problema, além de fornecer a visualização em forma de grafo, a ferramenta também gera uma DSM (SULLIVAN et al., 2001) que apresenta a mesma informação contida no grafo na forma de uma matriz. O mesmo padrão de cores utilizado no grafo também é utilizado na DSM, com elementos em comum em azul e elementos

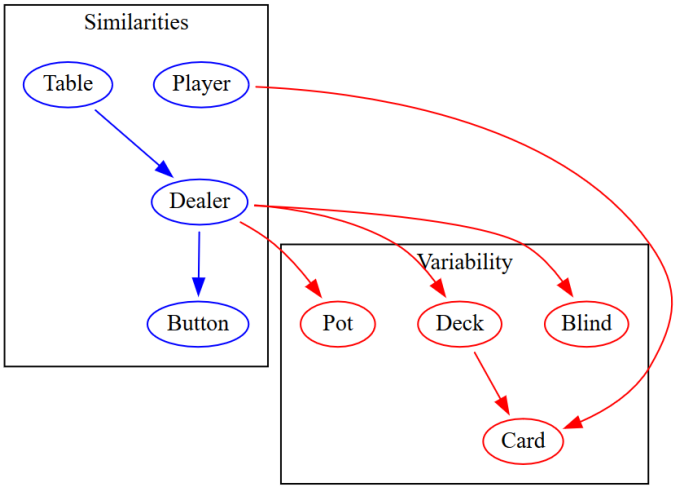

Figura 3.7 PLAR Tool: Grafo de Dependências decorado.

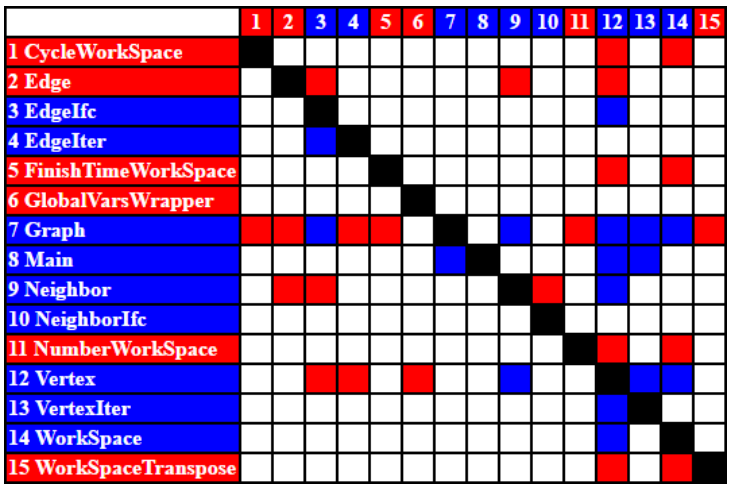

Figura 3.8 PLAR Tool: DSM decorada.

variáveis em vermelho.

Diagrama de Classes. A PLAR Tool gera um arquivo de texto utilizado pela ferramenta PlantUML[10]. Este arquivo de texto contém as informações sobre as classes que compõem a PLA e suas relações sendo utilizado pelo PlantUML para gerar um diagrama de classes simplificado da PLA. Neste digrama temos apenas as classes que compõem a PLA e suas relações. Assim como nas duas outras visualizações, o padrão de cores também é mantido para o diagrama de classes. Porém, nesta visualização não é possível mapear o esquema de cores para as relações. A Figura 3.9 apresenta um exemplo de diagrama de classes gerado pela ferramenta.

[10]http://plantuml.com/

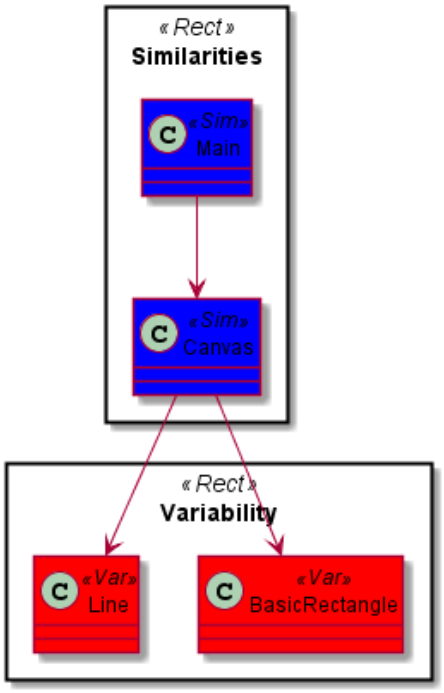

Figura 3.9 PLAR Tool: Diagrama de Classes para a PLA.

3.2.7 Características Técnicas

Linguagem: A ferramenta PLAR Tool está implementada em Java.

Entrada: Arquivos MDG de produtos de uma SPL.

Saídas:

- PLA recuperada, com informações em um arquivo de texto a respeito dos elementos comuns e variáveis bem como uma relação contendo os produtos que possuem cada elemento e relação encontrado.
- Relatório de métricas
- Arquivos em diferentes formatos para visualização da PLA (dot, html, txt)

Código fonte da ferramenta, binários e testes: https://github.com/MPassos/PLAR-Tool

Ferramenta de geração de produtos: FeatureIDE[11].

O FeatureIDE é um plugin para o Eclipse[12] cujo objetivo é fornecer suporte ao desenvolvimento de linhas de produto de software, incluindo geração de modelos de *features*, geração de produtos, entre outros.

[11] v.ht/FeatureIDE
[12] https://eclipse.org/

Ferramenta de extração: STAN4J[13].

O STAN4J é um plugin para o Eclipse cujo o objetivo é realizar a análise estrutural de projetos de software feitos em código Java. Com ele é possível verificar as dependências das classes e suas relações, permitindo a exportação dessa informação para um arquivo MDG.

Ferramenta de apoio: GraphViz[14].

A ferramenta Graphviz é utilizada para geração da visualização de módulos. A visão de módulo interativa é feita utilizando a biblioteca jquery.graphviz.svg[15] feita em JavaScript.

Licença:

A PLAR Tool é uma ferramenta gratuita sob a licença de uso GNU GPL v3[16] e licença de conteúdo Creative Commons[17].

3.2.8 Limitações da Ferramenta

- **Faltam informações detalhadas sobre classes, métodos e interfaces.**

  A PLAR Tool recebe informações extraídas pela ferramenta STAN4J, a qual extrai apenas o nome das classes e as relações entre essas classes. Informações sobre os métodos e atributos de cada classe, se a classe é abstrata ou é uma interface, se a relação entre as classes estabelece uso, herança ou implementação de interface não são mapeadas.

  Apesar disso, o MDG utilizado como entrada para a ferramenta PLAR Tool fornece informações suficientes para recuperação de PLA com informações úteis para desenvolvedores da SPL e arquitetos, a saber, classes e relações utilizadas para construir a SPL, explicitando a variabilidade.

- **Dependência sobre formato MDG de STAN4J.**

  A ferramenta PLAR Tool dá suporte à recuperação com base em arquivos MDG gerados pela ferramenta STAN4J. O STAN4J recuperar as dependências apenas de código-fonte de projetos Java o que acaba limitando a implementação da técnica pela ferramenta.

  Como trabalho futuro, pretende-se desenvolver adaptadores para diferentes tipos de arquivo MDG gerados por outras ferramentas de extração.

- **Recuperação baseada em nomes.**

[13] http://stan4j.com/
[14] http://www.graphviz.org/
[15] https://github.com/mountainstorm/jquery.graphviz.svg
[16] http://www.gnu.org/licenses/gpl.html
[17] https://creativecommons.org/licenses/by-sa/4.0/

Como estamos utilizando produtos gerados por uma mesma SPL, a análise dos elementos é feita baseada no nome das classes, caso o nome de alguma classe esteja diferente entre os produtos, esta classe será considerada com variável, mesmo que ela possua o mesmo papel. Dessa forma a versão atual da PLAR Tool não é indicada para recuperação da arquitetura de referência da SPL (NAKAGAWA; BECKER; MALDONADO, 2013) e para migração de família de produtos para o paradigma SPL.

3.2.9 Exemplo de Recuperação realizada pela ferramenta

A linha de produtos de software PayCard[18] será utilizada para ilustrar o funcionamento da PLAR Tool. Esta SPL implementa produtos relativos ao controle de operações de um cartão de crédito. A Tabela 3.1 apresenta algumas informações deste projeto SPL.

Tabela 3.1 Informações sobre a SPL Paycard.

SPL	**[F]**	**[FM]**	**[FO]**	**[P]**	**[C]**	**[LOC]**
PayCard	3	0	3	6	7	441

Legenda: [F] Features [FM] Features Mandatórias [FO] Features Opcionais [P] Produtos [C] Classes [LOC] Linhas de Código

O projeto PayCard está implementado em Java e utiliza o gerador de produtos FeatureHouse[19]. Possui um total de três *features*, sendo todas opcionais, resultando em seis produtos distintos. A Figura 3.10 apresenta a visão de módulo para a PLA recuperada com a PLAR Tool a partir dos seis produtos da linha PayCard.

Na Figura 3.10, as classes e relações variáveis e comuns são apresentados na cor vermelha, enquanto que as classes e relações comuns são apresentadas na cor azul. Além disso ao visualizar a PLA em um navegador web é possível dar foco a um elemento específico da visão de módulo, fazendo com que os demais elementos fiquem desfocados e seja possível analisar melhor as relações de determinado elemento selecionado.

A Tabela 3.2 apresenta algumas das métricas calculadas a partir da PLA recuperada. O foco destas métricas é medir o grau de reúso dos componentes encontrados na PLA. Estes resultados podem ser usados como parâmetro de qualidade da PLA recuperada (ZHANG et al., 2008).

3.2.10 Ferramentas Relacionadas

A revisão de outras ferramentas de recuperação de arquitetura de projetos SPL foi de fundamental importância para o desenvolvimento da PLAR Tool. Durante a investigação, constatamos que todas utilizam ferramentas auxiliares para recuperar a arquitetura dos

[18] http://spl2go.cs.ovgu.de/projects/57
[19] http://v.ht/FeatureHouse

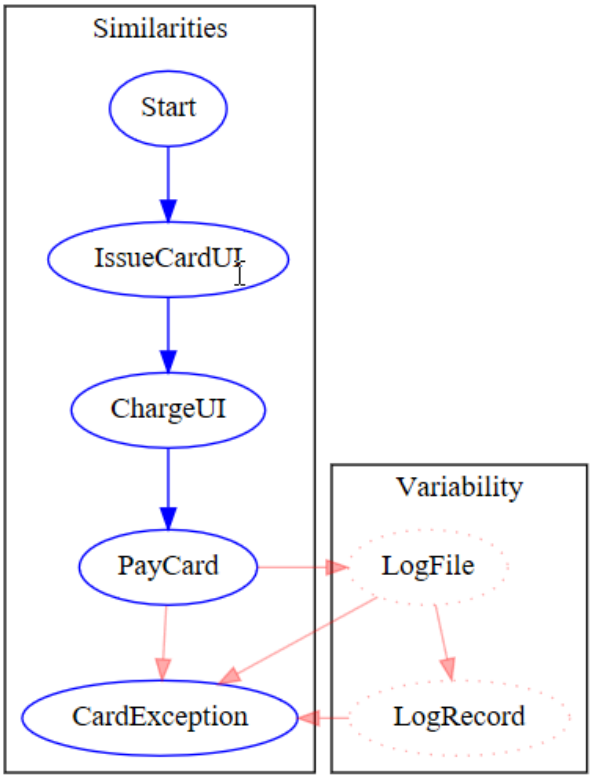

Figura 3.10 Visão de módulo recuperada a partir da PLAR Tool

Tabela 3.2 Métricas coletadas durante a análise da PLA da SPL PayCard.

SPL	SSC	SVC	RSSC	RSVC	CO	OR	CM	MR	PLTV
PayCard	0.71	0.29	0.37	0.63	2	5	5	3	7

Legenda: [SSC] - Structure Similiraty Coefficient, [SVC] - Structure Variability Coefficient, [RSSC] - Relation Structure Similiraty Coefficient, [RSVC] - Relation Structure Variability Coefficient, [CO] - Class Optional, [CM] - Class Mandatory, [OR] - Optional Relation, [MR] - Mandatory Relation, [PLTV] - Product Line Total Variability

produtos da linha, a fim de identificar os seus elementos e relações e, a partir dessa arquitetura inicial, realizar o processo de recuperação da PLA.

A tabela 3.3 apresenta uma comparação entre a PLAR tool e as ferramentas relacionadas mencionadas na Seção 2.6.

A seguir detalhamos alguns pontos em comum e alguns diferenciais da ferramenta PLAR Tool em relação as demais ferramentas:

- **Recuperação da arquitetura de cada produto de software.** A arquitetura de cada produto gerado pela linha é extraída por uma ferramenta específica para este propósito.

Tabela 3.3 Comparativo entre Ferramentas de Recuperação de PLA com a PLAR Tool

Ferramenta/Abordagem	Objetivo	Visão Gerada	LP
ROMANTIC	Recuperar a PLA	Componente-Conector	Java
Linsbauer	Recuperar o modelo de features	Diagrama de Feature	Java/C
Eixelsberger	Adaptar família de software para SPL	ND	ND
RECoVar	Melhorar atividades relacionadas a variabilidade em SPL evolutiva	Árvore de Variação	C/C++
PLAR Tool	Recuperar a PLA	Módulo, DSM, Diagrama de Classes	Independente
Legenda	[ND] - Não Disponível	[LP] - Linguagem de Programação	

- **Análise incremental de produtos.** Cada produto é analisado de forma individual e incremental e, ao final da análise, os elementos comuns e variáveis dos produtos são identificados.

A ferramenta PLAR Tool possui alguns diferenciais em relação às outras ferramentas:

- **Geração de diferentes visões de módulo da PLA**. As ferramentas oferecem apenas um tipo de visão arquitetural da PLA. A ferramenta PLAR Tool permite a geração de mais três visões arquiteturais distintas da PLA.

- **Cálculo de Métricas**. As ferramentas analisadas se limitam a recuperar a PLA. A ferramenta desenvolvida nesta pesquisa de mestrado, além de recuperar a PLA, realiza o cálculo de um conjunto de métricas que permitem ao usuário da ferramenta mensurar a qualidade da PLA recuperada.

- **Disponibilidade**. A maior dificuldade durante o trabalho foi encontrar ferramentas de recuperação de arquitetura disponíveis para uso. As abordagens de recuperação de PLA estudadas não disponibilizam a(s) ferramenta(s) utilizada(s) em seus estudos. A ferramenta PLAR Tool está disponível como software livre, para incentivar seu uso e a colaboração com outros pesquisadores e desenvolvedores interessados.

- **Independência de Linguagem de Programação**. As ferramentas estudadas possuem algum tipo de limitação no que diz respeito a linguagem de programação do projeto SPL e/ou o método de implementação da variabilidade. A PLAR Tool foi projetada para não impor restrições sobre a linguagem de programação do projeto SPL, pois a ferramenta realiza sua recuperação com base em arquivos obtidos a partir de ferramentas de extração de dependência do código fonte, as quais podem ser implementadas em qualquer linguagem de programação, ampliando as possibilidades de recuperação da PLAR Tool. Entretanto no momento da escrita dessa dissertação, a PLAR Tool apenas oferece suporte aos arquivos de dependência gerados pela ferramenta STAN4J, que recupera as dependências apenas de projetos

Java. Como trabalho futuro planejamos estender o suporte da ferramenta para inclusão de outras ferramentas de recuperação de dependências para projetos feitos em diferentes linguagens de programação para alcançar este objetivo.

Capítulo

4

AVALIAÇÃO DA TÉCNICA PLAR

Este capítulo apresenta um estudo realizado com alunos de pós-graduação e graduação com o objetivo de avaliar a técnica PLAR. Os projetos SPL estudados foram desenvolvidos pelos próprios alunos em uma disciplina de pós-graduação. A avaliação da técnica PLAR foi realizada com base em sua implementação na ferramenta PLAR Tool, considerando a precisão e abrangência da recuperação das PLAs estudadas. A saída gerada pela ferramenta PLAR Tool, após a recuperação da PLA de projetos SPL, também foi avaliada.

A seção 4.1 apresenta o objetivo, questões de pesquisa e contexto do estudo e a seção 4.2 apresenta o planejamento para sua condução. A seção 4.3 apresenta detalhes sobre a execução do estudo. A seção 4.4 apresenta a análise dos dados obtidos a partir dos questionários respondidos pelos participantes do estudo. A seção 4.5 apresenta a interpretação de resultados e os riscos à validade do estudo. Por fim, a seção 4.6 apresenta considerações finais e sugestão de trabalhos futuros.

4.1 ESCOPO DO ESTUDO

Aplicamos o método GQM (BASILI; CALDIERA; ROMBACH, 1994) para definir o processo que foi utilizado como guia durante a condução deste estudo, de natureza avaliativa.

4.1.1 Objetivo

O objetivo deste estudo foi analisar a ferramenta PLAR Tool com o propósito de avaliação no que diz respeito à precisão e à abrangência da recuperação, e da utilidade das saídas geradas após a recuperação da PLA, do ponto de vista de arquitetos e desenvolvedores, no contexto de projetos SPL desenvolvidos no ambiente acadêmico.

4.1.2 Questões de Pesquisa

Para alcançar este objetivo as seguintes questões de pesquisa foram levantadas:

- **RQ1:** A representação da PLA gerada pela PLAR Tool atende às necessidades dos desenvolvedores?
- **RQ2:** A representação da PLA gerada pela ferramenta apresenta todos os elementos variáveis e os comuns da SPL?
- **RQ3:** A representação da PLA gerada pela ferramenta pode ser útil em atividades de manutenção da SPL?

4.1.3 Contexto

Os participantes do estudo são estudantes de graduação e pós-graduação da Universidade Federal da Bahia (UFBA) que, ao longo de uma disciplina de pós-graduação, desenvolveram um projeto SPL.

A fim de mitigar problemas relacionados a familiaridade dos estudantes com o projeto por eles desenvolvido, solicitamos o código fonte dos projeto SPL e realizamos a recuperação da PLA correspondente com a ferramenta PLAR Tool. A PLA recuperada a partir de cada projeto SPL foi utilizada como material de avaliação em um *survey* aplicado aos participantes do estudo. O pesquisador esteve presente para apresentar as instruções e tirar dúvidas sobre o *survey*.

Tabela 4.1 Dados sobre os projetos SPL analisados.

	[#F]	*[#C]*	*[#LOC]*	*[#D]*	*[#G]*
WebStore	27	68	5017	9	ANT
SPLMessage	19	60	3085	1	ANT

Legenda: [**#F**] Features [**#C**]Classes [**#LOC**] Linhas de Código [**#D**] Desenvolvedores [**#G**]Gerador de Produtos

Os projetos SPL estudados foram `WebStore`, uma SPL que gera produtos com funcionalidades de uma loja online, e `SPLMessage`, uma SPL que gera produtos com funcionalidades de um chat de troca de mensagens instantâneas. A Tabela 4.1 sumariza os dados dos projetos analisados.

Na Tabela 4.1 não constam informações sobre a quantidade de produtos que podem ser gerados por cada projeto SPL, pois o processo de configuração e geração dos produtos das duas SPLs estudadas é feito por meio de compilação condicional, e controlado pelo desenvolvedor. Para geração de um produto, é necessário criar um arquivo de configuração contendo as *features* que irão compor o produto e então solicitar ao compilador que realize a construção do mesmo.

Para cada projeto SPL, dez arquivos de configuração foram criados da seguinte forma:

- Dois arquivos de configuração contendo um produto com todas as *features* disponíves e um produto apenas com as *features* mandatórias.

- Oito arquivos de configuração com *features* selecionadas de forma aleatória.

Após a criação de arquivos de configuração, os produtos foram gerados, servindo como entrada para o processo de recuperação da PLA, a partir da etapa de extração do MDG (Seção 3.2.4).

4.2 PLANEJAMENTO DO ESTUDO

4.2.1 Participantes

Os participantes deste estudo são discentes do Programa de Pós-graduação em Ciência da Computação da Universidade Federal da Bahia (PGCOMP-UFBA). Todos os participantes cursavam a mesma disciplina de pós-graduação, e estavam envolvidos no desenvolvimento de projetos SPL (Tabela 4.1).

4.2.2 Instrumentação

4.2.2.1 Questionário O questionário utilizado neste estudo está dividido em cinco partes:

- **Dados sobre os desenvolvedores -** Perguntamos aos desenvolvedores informações sobre sua experiência com programação, linha de produto de software e arquitetura de software.
- **Avaliação das Configurações -** Solicitamos aos desenvolvedores que os mesmos avaliassem as configurações escolhidas para geração os produtos, caso os desenvolvedores achassem necessário, poderiam sugerir novas configurações para serem incluídas.
- **Identificação de erros -** Nesta parte do questionário, os desenvolvedores poderiam informação os elementos da PLA que foram identificados de forma errada, ou que estavam ausentes.
- **Avaliação da Extração -** Neste ponto do questionário, fizemos diversas perguntas aos desenvolvedores a fim de que os mesmos avaliassem os resultados gerados pela PLAR Tool utilizando a escala de Likert.
- **Feedback** - Neste ponto os desenvolvedores foram incentivados a sugerir melhorias e dar uma feedback para o uso da ferramenta.

O modelo do questionário utilizado neste estudo encontra-se no Apêndice A.

4.2.3 Procedimento de coleta de dados

Durante este estudo os participantes tiveram que responder a um questionário. Os dados analisados neste estudo foram coletados a partir das respostas dos questionários obtidas dos participantes.

4.2.4 Procedimento de análise de dados

Para responder as questões de pesquisa levantadas, os participantes deste estudo responderam a um questionário. As perguntas do questionário utilizaram a Escala Likert (ALBAUM, 1997), sendo esta a métrica utilizada para realizar a avaliação das respostas fornecidas. As respostas de cada questionário foram compiladas numa planilha. Esta informação foi utilizada no processo de análise e interpretação dos dados coletados.

4.2.5 Avaliação da Validade do Estudo

Este estudo foi avaliado em relação a aspectos de validade externa (representatividade dos participantes) e de confiabilidade dos dados coletados e analisados.

4.3 EXECUÇÃO

4.3.1 Participantes

A Tabela 4.2 apresenta as informações relativas a experiência dos participantes do estudo. Os desenvolvedores possuíam diferentes perfis, tivemos desenvolvedores com experiência na indústria variando de 1 mês a 30 anos.

Tabela 4.2 Perfil dos Participantes do Estudo

Participante	Idade	Projeto	Sexo	Exp. Des. Soft.	Exp. Prog. Ind	Exp. Prod. Ac.	LP	Exp. SPL	Exp. Arq.
1	23	WebStore	M	Sozinho, Equipe, Empresa	2 anos	5 anos	PHP, Java, JavaScript, Ruby, C	3 Meses	Estudou
2	47	WebStore	M	Sozinho, Empresa	30 anos	-	Basic, Cobol, C, Assembly, PHP, Pascal, ASP	4 meses	Estudou
3	23	WebStore	M	Sozinho,Equipe,Empresa	1 mês	4 meses	Pascal, PHP, HTML, JavaScript, C++	Estudou	Estudou
4	27	WebStore	M	Sozinho,Equipe,Empresa	3 anos	2 anos	C, C++, C#, Java, JavaScript, Python, Basic	Estudou	2 anos
5	24	WebStore	F	Empresa	2 anos	3 anos	Java, PHP e C	6 meses	Estudou
6	40	WebStore	F	Empresa	6 anos	8 anos	Pascal, Cobol, Clipper, C, Visual Basic, Java	Nenhuma	Estudou
7	33	WebStore	F	Empresa	15 anos	5 anos	Delphi, Visual Basic, PHP, Java, Unify Vision, C#, Lua, ASP	1 ano	Estudou
8	31	WebStore	M	Equipe	3 anos	-	Object Pascal, C, C++, PHP, HTML, Java	6 meses	Estudou
9	25	SPLMessage	M	Sozinho,Equipe,Empresa	2,5 anos	2,5 anos	Java, C, C++, JavaScript	2 meses	2,5 anos

Oito dos nove desenvolvedores já haviam trabalhado empresas desenvolvendo software, entretanto, essa experiência nas empresas não foi refletida em conhecimento das áreas de SPL e Arquitetura de software (Colunas Exp. SPL e Exp Arq.).Todos os desenvolvedores programavam em mais de uma linguagem de programação (Coluna LP). Neste estudo, tivemos uma amostra de participantes com uma média de 7 anos de experiência de trabalho na indústria e uma média de 2,9 anos de experiência acadêmica.

4.3.2 Preparação

O processo de avaliação da ferramenta foi dividido em dois encontros: um encontro de explicação da ferramenta e extração de informações e um encontro para apresentação da recuperação e avaliação da ferramenta. Fizemos esta divisão a fim de minimizar possíveis dúvidas em relações as questões presentes no questionário avaliativo.

No primeiro encontro a ferramenta PLAR Tool foi apresentada aos desenvolvedores, o processo de recuperação utilizado e os arquivos de saída gerados pela ferramenta. Após

esta etapa de explicação da ferramenta uma reunião foi conduzida com a equipe de desenvolvedores de cada projeto analisado a fim de extrair informações sobre o projeto como: geração dos produtos, configurações válidas, conformidade do modelo de *features* com implementação, o que não foi implementado e conhecimento da equipe sobre o projeto. Estas informações foram utilizadas para refinar as perguntas utilizadas no questionários e facilitar o processo de recuperação da PLA de cada projeto.

Com as informações obtidas na primeira reunião, o processo de recuperação da PLA foi conduzido seguindo o esquema apresentado na Seção 3.2.4 com exceção do passo de geração de produtos que foi feita de forma manual, conforme explicado na seção 4.1.3. O resultado da recuperação foi utilizado para guiar os desenvolvedores no processo de responder as perguntas do questionário.

No segundo encontro, os resultados da recuperação foram apresentados aos desenvolvedores. O resultado das métricas obtidas e as visualizações da PLA foram exibidas, abrimos espaço para responder as dúvidas que surgiram durante a apresentação da PLA recuperada de cada projeto, em seguida, cada desenvolvedor recebeu uma cópia do questionário e foi orientado a responder. Para evitar que houvessem respostas em branco e sanar dúvidas em relação as perguntas do questionário, dois supervisores estavam disponíveis para este propósito.

4.3.3 Coleta de Dados Realizada

No segundo encontro com os participantes, foi entregue um questionário contendo algumas perguntas a respeito das saídas geradas pela ferramenta PLAR Tool para que pudessem emitir uma opinião sobre a informação fornecida.

Ao final do processo obtivemos 9 questionários respondidos, um para cada participante, cujas informações foram extraídas para realização da análise desta informação.

4.4 ANÁLISE

Esta seção apresenta os resultados das respostas dos questionários respondidos pelos desenvolvedores de acordo com os quatro critérios do questionário: Avaliação das configurações, Identificação de Erros e Avaliação da Extração.

4.4.1 Avaliação das Configurações

Todos os desenvolvedores consideraram as configurações escolhidas para representar os produtos como suficientes. A Tabela 4.3 mostra que apenas o participante 2 forneceu uma sugestão de configuração para ser analisada. Ao questionar o participante o porque desta configuração, o mesmo informou que havia sentido falta de alguns elementos na PLA e acreditava que esta configuração resolveria o problema. Utilizamos a configuração sugerida e realizamos uma nova recuperação e o problema foi resolvido.

Tabela 4.3 Configurações Sugeridas

Participante	Configuração Sugerida
1	-
2	Comunicação / BugTrack /Faq /Fale Conosco /Carrinho / Enviar Presente /Restante
3	-
4	-
5	-
6	-
7	-
8	-
9	-

4.4.2 Identificação de Erros

A Tabela 4.4 mostra que os desenvolvedores não encontraram erros quanto a classificação dos elementos na PLA. Entretanto houveram elementos que não foram mapeados, porém, o questionário não contemplou estes elementos. Sobre estes elementos não mapeados, verificamos que este fato ocorreu pois usamos uma quantidade de produtos insuficiente para mapear todos os elementos da PLA. Ao adicionar mais configurações de produtos, este elementos foram mapeados e os desenvolvedores constataram que não houve erros quanto a classificação destes elementos como comuns e variáveis.

4.4.3 Avaliação da Extração

As Figuras 4.1 e 4.2 apresentam o resultado da avaliação feita pelos desenvolvedores dos resultados gerados pela PLAR Tool respondendo a RQ1, RQ2 e RQ3.

A Figura 4.1 apresenta o resultado das respostas dos questionários para as perguntas relacionadas a qualidade de arquitetura recuperada pela ferramenta PLAR-Tool. As duas primeiras questões do questionário perguntaram aos desenvolvedores se as configurações escolhidas para construir os produtos representam bem o portfólio esperado e se a representação da PLA fornecida é satisfatória. Com a resposta destas duas perguntas perguntas pudemos responder a RQ1. Tivemos um alto nível de aceitação da representação da PLA, porém apesar deste alto grau de aceitação. Algumas ressalvas feitas pelos participantes do estudo. As questões de *feedback* tiveram o objetivo de mapear estas ressalvas.

As questões 3 e 5 buscaram responder a RQ2, pois nesta questão estávamos investigando se a representação da PLA fornecida estava mapeando os elementos corretamente. De acordo com o gráfico apresentado na Figura 4.1 pudemos constatar que não houve

Tabela 4.4 Avaliação das Configurações

Participante	Erros
1	-
2	-
3	-
4	-
5	-
6	-
7	-
8	-
9	-

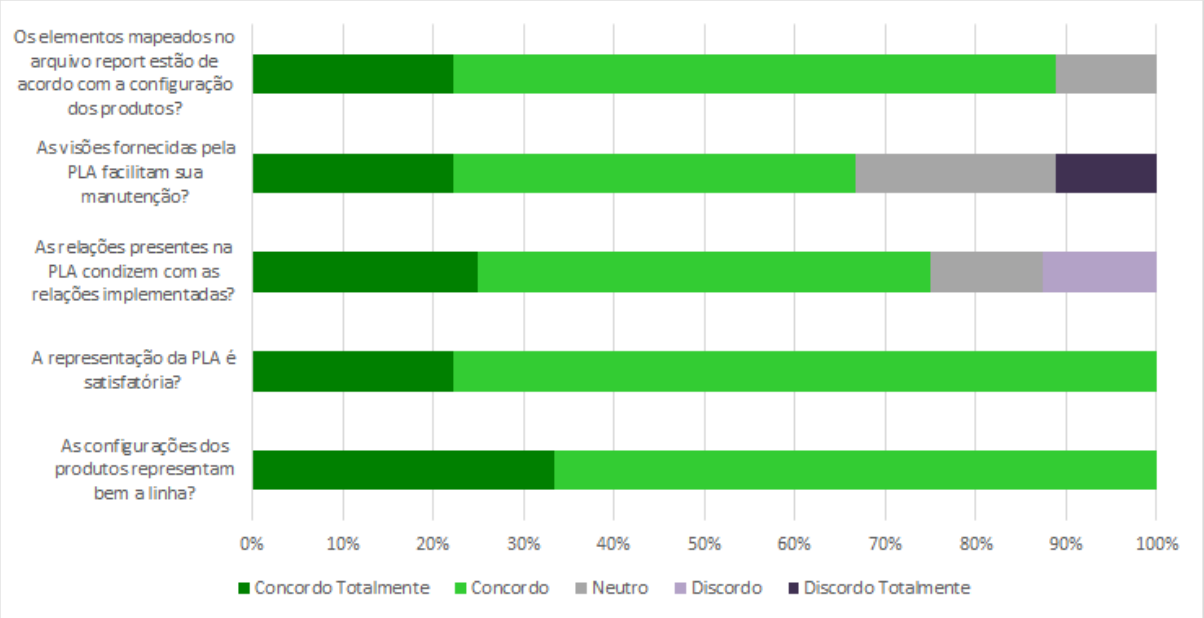

Figura 4.1 Avaliação da Extração Realizada

um resultado unânime entre os desenvolvedores, alguns desenvolvedores na questão 3 discordaram do mapeamento das relações na PLA e nos comunicaram isso. Verificamos posteriormente que a adição de mais configurações para os produtos resolve este problema

do falso mapeamento. A questão 5 também era relacionada ao mapeamento dos elementos da PLA, nesta questão alguns desenvolvedores não conseguiram analisar a tempo todos os relatório gerados pela ferramenta e como consequência, não emitiram opinião sobre o assunto.

A questão 4 foi utilizada para responder a RQ3. A visualização da PLA foi o ponto onde houve maior discussão entre os desenvolvedores, pois muitos consideraram a representação da PLA na visão de módulo confusa, o que fez alguns desenvolvedores não estarem de acordo com este tipo de representação da PLA, entretanto outros tipos de visualização foram fornecidos para este estudo o que contornou os problemas encontrados com a visão de módulo. Esta questão foi o ponto onde tivemos a maior quantidade de sugestões, sendo de fundamental importância para o posterior refinamento da ferramenta PLAR-Tool.

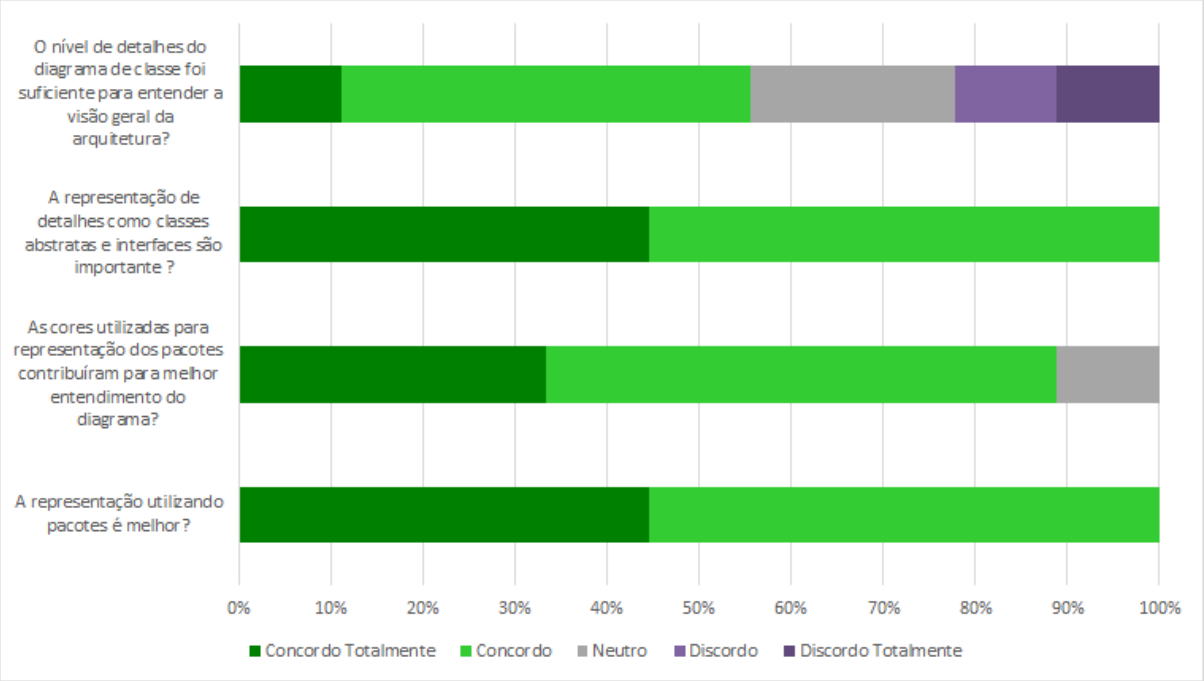

Figura 4.2 Avaliação Diagrama de Classes

O segundo bloco de perguntas do questionário teve o objetivo específico de avaliar o diagrama de classes gerado pela ferramenta PLAR-Tool. Na época do estudo, a visão de classe tinha acabado de ser implementada e não possuíamos nenhum tipo de *feedback* de especialistas e de experts, portanto decidimos incluir esta visualização no estudo a fim de avaliar a aceitação dos participantes e verificar os possíveis pontos de melhoria que poderiam ser incluídos para esta visualização.

No geral, os desenvolvedores aprovaram os resultados gerados pela ferramenta. A principal crítica dos desenvolvedores foi em relação a visão de módulo gerada pela ferramenta.

4.4.4 Feedback

Solicitamos no questionário que os participantes do *survey* informassem sugestões de melhoria e fizessem críticas aos resultados gerados pela ferramenta. Destacamos algumas das principais críticas e sugestões coletadas do questionário.

"Situar as linhas correlatoras do diagrama com maior clareza, utilizar uma linguagem menos implícita para o report e métricas."

"Mudar as cores, colocar cores mais neutras. Buscar uma outra maneira de tornar mais claros os relacionamentos, diminuir as linhas, o problema que provavelmente implica em perda de informação."

"Na representação matricial poderia usar alguma abordagem para representar o direcionamento das dependências. Na representação de grafo, buscar alguma forma de destacar relações desejadas dinamicamente."

"Acredito que seria interessante ter um projeto web que permitisse o usuário interagir com o diagrama de arquitetura gerado a partir da SPL. Uma outra coisa que poderia ser feita é a remoção de classes que não possuem relacionamentos."

Estas críticas foram utilizadas para realizar melhorias nas representações e relatórios gerados pela ferramenta. Dentre algumas mudanças realizadas após a aplicação do *survey*, podemos destacar:

- **Visualização de Módulo Dinâmica -** A visualização de módulo gerada pela ferramenta permite que o usuário selecione um módulo para deixar em evidência em relação aos demais, o que facilita a visualização de suas relações.

- **Geração de Páginas Web para apresentação de relatórios e visualizações -** Ao final do processo de geração da PLA, a ferramenta agora abre uma página web estática, onde o usuário pode navegar facilmente pelas diversas visualizações e relatórios gerados.

4.5 INTERPRETAÇÃO

4.5.1 Avaliação dos resultados

A avaliação dos desenvolvedores foi de fundamental importância para o refinamento da ferramenta PLAR-Tool. Neste estudo pudemos verificar a relevância dos arquivos de saída gerados pela ferramenta junto aos desenvolvedores dos projetos SPL cuja PLA foi recuperada, algo que não foi possível nos outros estudos realizados nesta pesquisa de mestrado.

No geral, tivemos boas avaliações dos desenvolvedores em relação a identificação dos elementos comuns e variáveis. Não houveram erros em relação aos elementos arquiteturais destacados como variáveis, porém, alguns desenvolvedores mencionaram que alguns elementos não haviam sido classificados, este problema é resolvido inserido novos produtos para realização da recuperação da PLA.

Houve muitas críticas em relação a visão de módulo da PLA, pois por se tratar de projetos com muitas classes e relacionamentos, a visão de módulo acabava muita vezes se

tornando confusa. Este retorno fez com que nos dedicássemos a refinar essa funcionalidade para a versão final da ferramenta apresentada nesta dissertação.

4.5.2 Ameaças a Validade

- **Ameaças a validade interna**
  - **Experiência dos Desenvolvedores** - A maioria dos desenvolvedores não tinha experiência com arquitetura de software, tampouco SPL, o que pode afetar na avaliação realizada devido a falta de familiaridade com estas duas áreas.
  - **Apreensão devido a avaliação -** Como esta avaliação foi feita durante o curso de uma disciplina, os estudantes poderiam ficar receosos sobre sua nota na atividade, o que poderia atrapalhar seu desempenho no estudo. Para evitar esse risco, informamos que o estudo não iria interferir nas notas da disciplina.
- **Ameaças a validade externa**
  - **Generalização dos desenvolvedores -** Os desenvolvedores que participaram deste estudo são alunos de pós-graduação e graduação, não sendo considerados como amostra representativa de potenciais usuários da ferramenta, o que pode impactar nos resultados de avaliação. Para mitigar esta ameaça, o processo de avaliação foi dividido em dois encontros, onde o primeiro foi inteiramente dedicado ao treinamento dos participantes envolvidos.

4.6 CONSIDERAÇÕES

Realizamos a avaliação da PLAR Tool por meio de um estudo conduzido com desenvolvedores de projetos SPL dentro do contexto acadêmico. Para tal, realizamos a recuperação de projetos desenvolvidos por eles e solicitamos que respondessem a um questionário de forma a avaliar os resultados gerados pela ferramenta.

Os resultados obtidos com as respostas do questionário foram de fundamental importância para realização de melhorias e inclusão de novas funcionalidades na versão atual da ferramenta PLAR Tool, utilizada nos estudos apresentados nos capítulos 5 e 6.

Planejamos replicar este estudo com as melhorias implementadas em outros projetos e com outros desenvolvedores para identificar mais pontos de melhoria que a ferramenta pode receber bem como novas funcionalidades a serem implementadas.

O questionário, o material utilizado para recuperar a PLA dos projetos para fins de replicação, o código fonte dos projetos e a planilha contendo as repostas de cada participante do estudo estão disponíveis no endereço: http://v.ht/EYfM.

Capítulo

5

ESTUDO EXPLORATÓRIO: RECUPERAÇÃO DA ARQUITETURA DE SEIS PROJETOS SPL

Este capítulo apresenta um estudo exploratório para investigar a qualidade da arquitetura de linha de produtos recuperada com a PLAR Tool.

A estrutura do capítulo adota, em sua maior parte, a recomendação para documentação de experimentos controlados de Wohlin et. al. (2012). A seção 5.1 apresenta o escopo do estudo, a definição do problema, os objetivos de pesquisa e o contexto; a seção 5.2 apresenta o planejamento do estudo exploratório; a seção 5.3 apresenta os processos de preparação dos projetos SPL estudados e coleta de dados; a seção 5.4 apresenta a análise de dados obtidos e responde às questões de pesquisa; a seção 5.5 discute as descobertas do estudo e suas ameaças à validade; a seção 5.6 apresenta os trabalhos relacionados e, por fim, a seção 5.7 apresenta a conclusão e os trabalhos futuros.

5.1 MOTIVAÇÃO

A PLA recuperada a partir de um conjunto de produtos de um projeto SPL, contém elementos comuns, presentes em todos os produtos, e elementos variáveis, presentes em alguns produtos. Neste contexto, a qualidade da arquitetura de linha de produtos recuperada deve ser avaliada. A qualidade da PLA recuperada pela ferramenta PLAR-Tool pode ser avaliada com respeito ao grau de reúso de seus componentes, de forma que, quanto maior o grau de reúso, melhor a qualidade (ZHANG et al., 2008).

A PLAR Tool permite a avaliação do grau de reúso da PLA recuperada e suporta um subconjunto das métricas propostas por Zhang et. al. (2008) (Tabela 2.2), bem como outras propostas que buscam mensurar e identificar os componentes comuns e variáveis dentro da PLA (JUNIOR; GIMENES; MALDONADO, 2008).

Seis projetos SPL *open source* foram estudados para avaliar a qualidade da arquitetura recuperada segundo métricas suportadas pela ferramenta PLAR Tool. A relação entre a variabilidade presente nos produtos de cada SPL e os valores das métricas de reúso também foi investigada.

5.1.1 Objetivos de Pesquisa

O método GQM (BASILI; CALDIERA; ROMBACH, 1994) foi utilizado para definir o processo que guiou a condução do estudo, de natureza exploratória.

Objetivo (*Goal*) - O objetivo deste estudo foi avaliar a qualidade da PLA recuperada com o propósito de avaliação no que diz respeito ao reúso de componentes do ponto de vista do pesquisador dentro do contexto de projetos SPL *open source*.

Para alcançar o objetivo geral do estudo, definimos as seguintes questões de pesquisa:

- **RQ1:** A variabilidade presente na PLA é influenciada pela quantidade de produtos usados na recuperação da PLA? Com esta RQ investigamos como a quantidade de produtos inseridas para análise impacta a variabilidade encontrada na PLA.
- **RQ2:** O grau de reúso dos componentes da PLA é afetado pelo número de *features* opcionais da SPL? Com esta RQ o nosso objetivo foi investigar como a variabilidade se propaga entre as classes e suas relações e como isso afeta o nível de reúso dos componentes da PLA.
- **RQ3:** Qual a métrica que melhor identifica o reúso de componentes da PLA? Neste estudo utilizamos duas métricas para calcular o grau de reúso de componentes da PLA: SSC e CRR. Com esta questão, buscamos responder qual destas duas métricas melhor representa o reúso dos componentes da PLA.

5.1.2 Contexto

Foram escolhidos seis projetos SPL para o estudo. Quatro destes projetos (DPL, VOD, GOL e Zip Me) foram fornecidos pelo material utilizado no estudo conduzido por Linsbauer et. al. (LINSBAUER; LOPEZ-HERREJON; EGYED, 2016) onde estes projetos foram utilizados para recuperação do modelo de *features* a partir do código fonte. Os demais projetos analisados (GPL e Prop4J) foram escolhidos aleatoriamente do site SPL2GO[1]. Todos estes projetos foram feitos em Java. Com exceção dos projetos utilizados no estudo do Linsbauer et. al., os demais utilizaram o processo de recuperação proposto no Capítulo 3. Nas SPLs utilizadas pelo Lisnbauer et. al. apenas o código fonte dos produtos gerados por cada SPL estava disponível, e, portanto o processo de geração de produtos para estes casos não foi realizado.

Os seis projetos SPL utilizados neste estudo foram:

- *Draw Product Line* (DPL) - Uma SPL utilizada para aplicações de desenho.
- *Video on Demand* (VOD) - Uma SPL para desenvolvimentos de aplicações *video-on-demand*.
- *Game of Life* (GOL) - Uma SPL que simula o jogo de tabuleiro - jogo da vida.
- *Graph Product Line* (GPL) - Uma SPL que implementa bibliotecas de manipulação de grafos.

[1] http://spl2go.cs.ovgu.de/

Tabela 5.1 Projetos SPL Analisados

SPL	#F	#FM	#FO	#P	#C	LOC	Gener.
DPL	5	3	2	12	4	473	ND
VOD	11	6	5	32	42	5.2K	ND
Zip Me	7	2	5	32	31	6.2K	ND
GOL	21	12	9	65	21	1.0K	ND
GPL	38	18	20	155	15	1.4K	CIDE
Prop4J	13	0	13	5029	14	2.1K	Featurehouse

Legenda: [**#F**] Features [**#FM**] Features Mandatória [**#FO**] Features Opcional [**#P**] Produtos [**#C**] Classes [**LOC**] Linhas de Código [**Gener.**] Gerador de Produtos [**ND**] Não disponível

- *Prop4J* - Uma SPL que fornece bibliotecas para soluções de fórmulas de lógica proposicional.

A Tabela 5.1 apresenta informações sobre os projetos SPL analisados como: número de *features* de cada projeto (opcionais e mandatórias), quantidade de classes, produtos gerados, linhas de código e gerador de produtos utilizado nos projetos SPL. Os projetos SPL utilizados e o material de replicação do estudo podem ser encontrados no endereço: http://bit.ly/1UfiI3U

5.2 PLANEJAMENTO DO ESTUDO

5.2.1 Formulação de Hipóteses

Para responder as questões de pesquisa foram formuladas as seguintes hipóteses para cada RQ:

- **RQ1 - A variabilidade é influenciada pela quantidade de produtos fornecidos para a recuperação da PLA?**
  - H_{0a}: A variabilidade não é influenciada pela quantidade de produtos inseridas para análise.
  - H_{1a}: A variabilidade é influenciada pela quantidade de produtos inseridos para análise.
- **RQ2** O grau de reúso dos componentes da PLA é afetado pelo número de *features* opcionais?
  - H_{0b}: Todas as PLAs possuem o mesmo grau de reúso, e não há diferença estatística entre os valores.

  - H_{1b}: Existe pelos menos um valor CRR diferente.

- **RQ3 - Qual métrica melhor identifica o reúso dos componentes da PLA?**
  - H_{0c}: Não existe diferença estatística significante entre as duas métricas.
  - H_{1c}: A métrica CRR apresenta um melhor grau de reúso do que a métrica SSC.
  - H_{2c}: A métrica SSC apresenta um melhor grau de reúso do que a métrica CRR.

5.2.2 Instrumentação

As etapas apresentadas no Capítulo 3 para recuperação da PLA foram seguidas neste estudo.

5.2.3 Procedimento de Coleta de Dados

Para cada projeto SPL analisado neste estudo, todas as métricas utilizadas para realização das análises foram coletadas durante o procedimento de recuperação da PLA feito pela ferramenta PLAR Tool.

5.2.4 Procedimentos de Análise

As métricas coletadas durante o estudo foram avaliadas quantitativamente (WOHLIN et al., 2012) de forma a responder as questões de pesquisa levantada, para isso utilizamos os métodos estatísticos ANOVA e e teste de Tukey. A análise estatística da informação coletada pode ser encontrada na seção 5.4.7 .

5.2.5 Avaliação da Validade do Estudo

Este estudo foi avaliado quanto aos aspectos de validade externos (representatividade dos projetos analisados) e sobre os aspectos de confiabilidade dos dados coletados e analisados. A discussão sobre estas ameaças a validade encontra-se na Seção 5.5.2.

5.3 EXECUÇÃO

5.3.1 Preparação

Para cada projeto SPL estudado, foram gerados primeiramente, todos os produtos a partir de configurações válidas para os projetos nos quais isto foi possível (GPL); nos demais projetos, os produtos já estavam disponíveis ou outro método de geração de produtos havia sido empregado.

Com base no código fonte dos produtos de cada SPL, foram extraídos os arquivos MDG que servem de entrada para a ferramenta PLAR Tool prosseguir com a recuperação da PLA de cada projeto.

5.3.2 Coleta de Dados Realizada

Os dados utilizados para avaliar cada PLA recuperada foram coletados a partir das métricas geradas pela ferramenta PLAR-Tool. Ao final da recuperação de cada projeto SPL, um relatório com as métricas calculadas foi gerado. Os dados contidos neste relatório foram utilizados para realizar a análise da qualidade da arquitetura recuperada.

De acordo com Zhang et. al. (2008), a qualidade de uma PLA pode ser medida baseada no grau de reúso dos seus componentes (ZHANG et al., 2008). As métricas responsáveis por esta avaliação qualitativa são (i) SSC (*Structure Similatiry Coefficient*), (ii) SVC (*Structure Variability Coefficient*) e (iii) CRR (*Component Reuse Rate*). As métricas propostas por Oliveira Junior et. al. (2008) medem a quantidade elementos comuns e variáveis da PLA. Estes valores servem como base para o cálculo das métricas propostas por Zhang et. al. (2008).

Para investigar a qualidade da PLA recuperada, investigamos os valores das métricas SSC, SVC e CRR. Projetos SPL que obtiveram um alto valor de SCC e um baixo valor de SVC indicam que a PLA é composta em sua maioria por componentes comuns a todos os produtos. Além disso a PLAR Tool calcula a métrica CRR de forma individual para cada elemento. Valores acima de 50% significam que o elementos analisado foi encontrado em mais da metade dos produtos gerados pela SPL o que indica um bom grau de reúso. A Tabela 5.2 apresenta um resumo das métricas utilizadas.

Tabela 5.2 Métricas de Zhang et. al. e Oliveira Junior et. al. para PLA

Métrica	Descrição	Referência
SSC	Calcula a similaridade geral dos elementos da PLA.	(ZHANG et al., 2008)
SVC	Calcula a variabilidade geral dos elementos da PLA.	(ZHANG et al., 2008)
CRR	Calcula a taxa de reúso de cada elemento da PLA.	(ZHANG et al., 2008)
ClassVP	Indica se determinada classe é um ponto de variabilidade.	(JUNIOR; GIMENES; MALDONADO, 2008)
ClassOptional	Calcula o número de classes opcionais.	(JUNIOR; GIMENES; MALDONADO, 2008)
ClassMandatory	Calcula o número de classes mandatórias.	(JUNIOR; GIMENES; MALDONADO, 2008)
PLTotalVariability	Estima a variabilidade encontrada na PLA	(JUNIOR; GIMENES; MALDONADO, 2008)
ComponentVariable	Indica se determinado componente da PLA possui algum tipo de variabilidade	(JUNIOR; GIMENES; MALDONADO, 2008)

5.4 ANÁLISE

A Tabela 5.3 apresenta os valores das métricas coletadas para cada PLA recuperada.

A métrica `SSC` calcula o nível de similaridade dos elementos da PLA. De acordo com Zhang et. al. (2008), quanto maior for o valor desta métrica, melhor o reúso dos elementos da PLA (o valor máximo para esta métrica é 1).

A métrica `SVC` calcula o nível de variabilidade dos elementos da PLA. Quando maior for o valor desta métrica, pior é o reúso dos elementos da PLA (o valor máximo é 1). As métricas `RSSC` e `RSVC` são análogas às métricas SSC e SVC, respectivamente, mas consideram as relações entre elementos arquiteturais.

A métrica `CRR` calcula a porcentagem de produtos que possuem determinado elemento ou relação. Um valor de 100% indica que o elemento/relação é encontrado em todos os produtos que são gerados por aquela PLA; valores próximos de 100% indicam que o

Tabela 5.3 Métricas Coletadas durante o estudo

SPL	SSC	SVC	RSSC	RSVC	[#CO]	[#OP]	[#CM]	[#MR]	[#PV]
DPL	0.5	0.5	0.34	0.66	2	2	2	1	4
VOD	0.77	0.23	0.7	0.3	10	23	32	55	33
Zip Me	0.8	0.2	0.7	0.3	6	14	25	32	20
GOL	0.62	0.38	0.68	0.32	8	11	13	24	19
GPL	0.6	0.4	0.42	0.58	6	23	9	16	29
Prop4J	0.07	0.93	0.0	1.0	13	50	1	0	63

Legenda: [**#CO**] ClassOptional [**#OP**] OptionalRelation [**#CM**] ClassMandatory [**#MR**] MandatoryRelation [**#PV**]PLTotalVariability

elemento/relação possui um bom grau de reúso. Entretanto, a métrica `CRR` não está presente na Tabela 5.3, visto que calcula o grau de reúso de cada elemento da PLA individualmente. Discutiremos os resultados desta métrica em detalhe para alguns dos projetos analisados.

As métricas `ClassMandatory` e `ClassOptional` calculam o número de classes que são mandatórias e opcionais respectivamente. De acordo com Oliveira Junior et. al. (2008), o número de classes mandatória deve ser superior ao número de classes opcionais, o que determina o grau de reúso dos elementos da PLA. As métricas `OptionalRelation` e `MandatoryRelation` usam o mesmo princípio para calcular o número de relações mandatórias e de relações opcionais.

A métrica `PLTotalVariability` é apresentada na Tabela 5.3 como uma métrica que estima a variabilidade total encontrada na PLA; isso se deve ao fato de que os arquivos MDG utilizados pela PLAR Tool não capturam informações detalhadas sobre os métodos contidos nas classes e seus respectivos atributos. A Tabela 5.3 mostra o valor de `PLTotalVariability` como a soma dos valores de `ClassOptional` e `OptionalRelation` uma vez que foram os únicos tipos de variabilidade que poderiam ser medidos pela métricas coletadas pela PLAR Tool.

As métricas `ClassVP` e `ComponentVariable` não estão presentes na Tabela 5.3 porque apenas especificam se um determinado elemento da PLA é variável ou não. Essa informação pode ser visualizada nos arquivos de representação gerados pela PLAR Tool.

5.4.1 Resultados DPL

Este foi o menor projeto SPL analisado neste estudo. A Tabela 5.3 mostra a média de reúso dos componentes desta PLA - o valor da métrica SSC foi de 0.5. Entretanto a métrica RSVC apresentam um alto valor o que indica que a maior parte das relações desta PLA são variáveis. A Figura 5.1 apresenta a PLA desta SPL.

A Tabela 5.4 os valores da métrica CRR para os elementos da PLA. Esta métrica foi coletada em três estágios de comparação, o primeiro comparava dois produtos (CRR_{par}), o segundo comparou 8 produtos (CRR_8) e o terceiro comparou todos os produtos (CRR_{todos}).

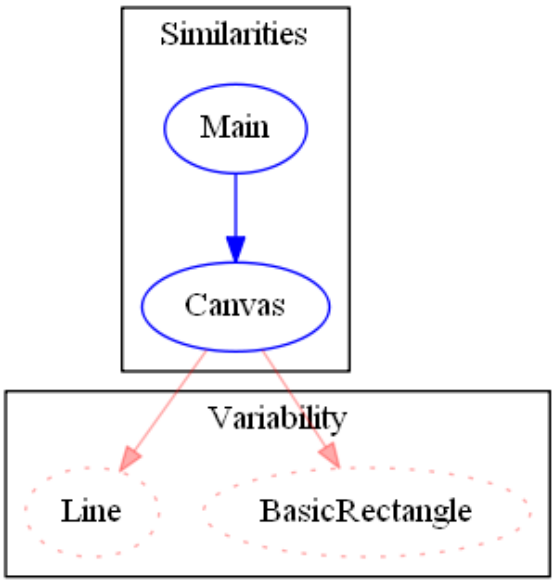

Figura 5.1 DPL PLA

Investigando os valores de CRR percebemos que alguns elementos da PLA possuem um grau de reúso superior a 50% indicando que qualquer mudanças feita pelos desenvolvedores nestas classes podem impactar muitos produtos da SPL (LINSBAUER; LOPEZ-HERREJON; EGYED, 2016).

Tabela 5.4 Valores da métrica CRR para a SPL DPL

Element	**CRR_{par}**	**CRR_{8}**	**CRR_{todos}**
BasicRectangle	100.0	62.5	66.66667
Canvas	100.0	100.0	100.0
Line	50.0	50.0	66.66667
Main	100.0	100.0	100.0

O número de produtos utilizados nesta comparação influencia na precisão dos valores das métricas. Por exemplo, a classe *BasicRetangle*, teve o valor de sua métrica alterado de 100% (comparando apenas dois produtos) para 62.5% (comparando oito produtos) e 66.7% (comparando todos). Um cenário diferente pode ser observado na classe *Line*, ao comparar apenas dois produtos a variabilidade deste elemento foi identificada, mas desta vez, ao aumentar a quantidade de produtos em análise, houve um aumento da precisão da métrica que variou de 50% para 66.7%.

5.4.2 Resultados para VOD

A SPL *Video on Demand* é uma SPL de tamanho mediano comparada a outras SPL utilizadas neste estudo. O valor da métrica SSC foi de 0.7 o que indica que os 32 produtos desta SPL fazem reúso de boa parte de seus componentes. Valores similares foram obtidos analisando as relações.

Como esta SPL possui 42 classes decidimos não colocar a tabela contendo os valores obtidos da métrica CRR para não tomar espaço. Todo este material pode ser encontrado no material de replicação fornecido anteriormente.

A SPL VOD possui um total de 32 classes mandatórias e 10 classes opcionais. Porém, identificamos que as classes opcionais tem um CRR de 50%, o que indica que estes elementos foram utilizados em pelo menos metade dos produtos da linha, indicando um alto nível de reúso dentro da SPL. Entretanto, o mesmo não pode ser dito das relações, algumas delas apresentaram valores da métrica CRR como 25% e 3.25% o que indica que poucos produtos utilizaram estas relações. Por esta razão uma refatoração dos elementos envolvidos nestas relações deve ser considerada (ZHANG et al., 2008).

A Figura 5.2 apresenta a PLA recuperada deste projeto SPL.

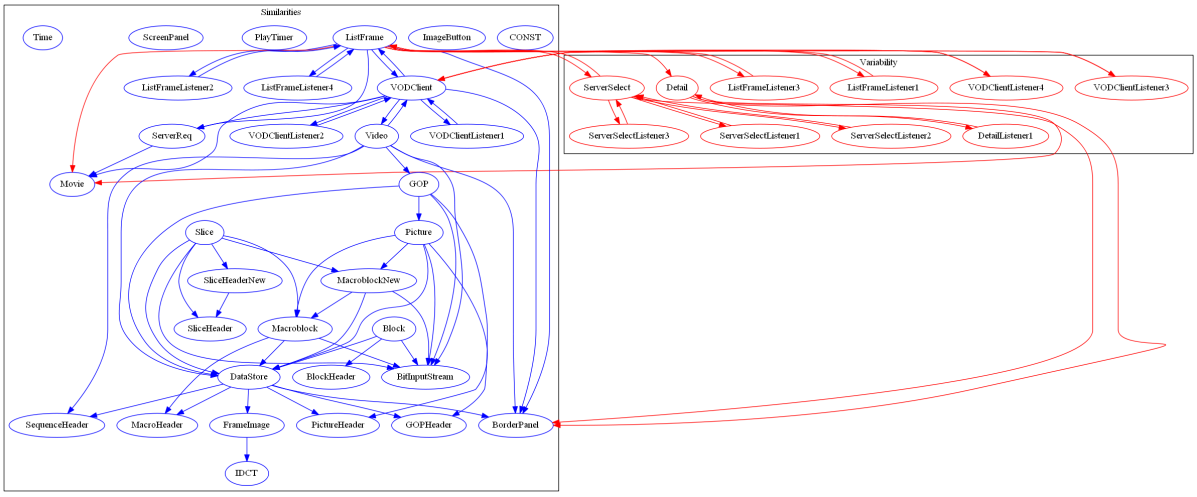

Figura 5.2 VOD PLA

5.4.3 Resultados para ZipMe

Semelhante a SPL VOD, Zip Me também é um projeto de tamanho mediano com 32 produtos compondo a linha. O valor da métrica SSC foi de 0.8 indicando que praticamente todos os elementos foram reutilizados entre os produtos. As relações também apresentarão um alto valor da métrica RSSC (0.7).

Pelos mesmos motivos da SPL VOD, não iremos apresentar a tabela com os valores da métrica CRR. Analisando os valores da métrica CRR verificamos que os 6 elementos identificados como variáveis possuem um nível de reúso superior a 50%. O mesmo cenário foi observado com as relações. Das 14 relações identificadas como variáveis, 11

apresentaram um valor de CRR superior 50% enquanto que as demais tiveram o valor de 25%.

A Figura 5.3 apresenta a PLA recuperada da SPL ZipMe.

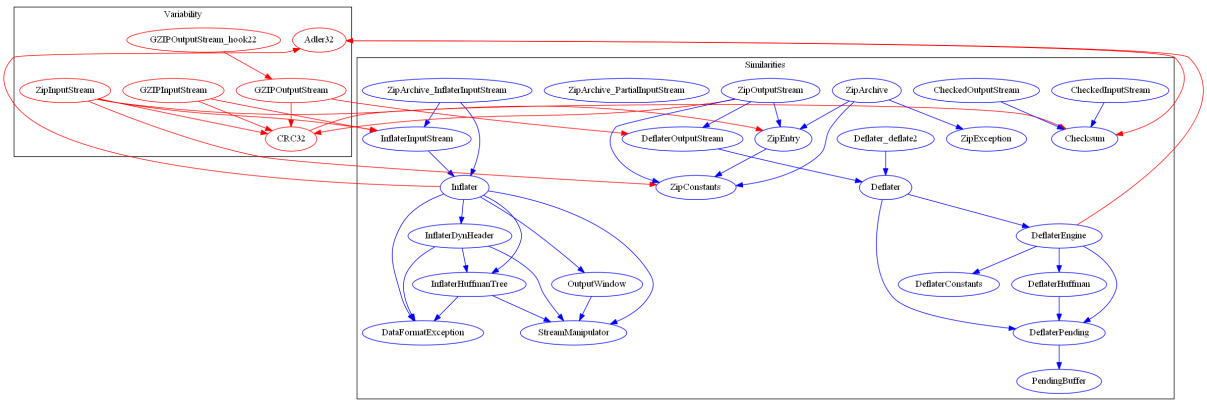

Figura 5.3 ZipMe PLA

5.4.4 Resultados para GOL

A SPL *Game of Life* foi a segunda maior SPL analisada (no que diz respeito a quantidade de produtos gerados). O valor da métrica SSC foi de 0.62% indicando que existem mais elementos comuns do que variáveis. A Tabela 5.5 apresenta os valores da métrica CRR para cada elemento da PLA. Coletamos a métrica CRR em três estágios distintos de comparação: em par, utilizando 45 produtos e utilizando todos os produtos.

Em relação ao número de elementos na comparação, utilizando apenas dois produtos, quatro elementos não foram mapeados (*GenerationSelector*, *PlaygroundIO*, *PopUpMenu*, and *Suite*). Ao adicionar mais produtos para análise, este elementos passaram a ser identificados.

Apesar de ter um valor menor da métrica SSC em comparação as SPL VOD e Zip Me, alguns dos elementos desta SPL apresentaram valores de CRR alto como 98% (*ClearGeneratorStrategy*) e 73% (*FormGeneratorStrategy*), o que indica que esta SPL faz um bom reúso de seus componentes. Um comportamento análogo foi observado nas relações desta PLA. A Figura 5.4 apresenta a DSM da PLA recuperada.

Observamos que a DSM da GOL não apresenta um nó central, comparado com as DSM da GPL e Prop4J, todas as relações estão espalhadas entre os elementos. Entretanto percebemos que existem alguns elementos que não possuíam quaisquer tipos de relacionamento, acreditamos que isso pode indicar features que foram descartadas ou não implementadas cuja as classes permaneceram no código fonte.

5.4.5 Resultados para GPL

A SPL GPL foi a maior SPL analisda (em relação a quantidade de produtos gerados) neste estudo. O valor da métrica SSC é de 0.6 indicando que existem mais elementos comuns do que variáveis na PLA. Entretando, o oposto acontece com as relações, que

Tabela 5.5 Valores da métrica CRR para a SPL GOL

Element	$\mathbf{CRR_{par}}$	$\mathbf{CRR_{45}}$	$\mathbf{CRR_{todos}}$
ButtonsToolBar	100.0	100.0	100.0
ClearGeneratorStrategy	100.0	100.0	98.46154
FormGeneratorStrategy	50.0	73.333336	73.84615
GODLModel	100.0	100.0	100.0
GenerationScheduler	100.0	100.0	100.0
GenerationSelector	NA	48.88889	49.23077
GeneratorStrategy	100.0	100.0	98.46154
GolView	100.0	100.0	100.0
LifeForm	100.0	100.0	100.0
LifeFormIterator	100.0	100.0	100.0
Main	100.0	100.0	100.0
ModelObservable	100.0	100.0	100.0
ModelObserver	100.0	100.0	100.0
Playground	100.0	100.0	100.0
PlaygroundIO	ND	46.666668	49.23077
PlaygroundMouseAdapter	100.0	100.0	100.0
PlaygroundPanel	100.0	100.0	100.0
PopUpMenu	ND	46.666668	49.23077
RandomGeneratorStrategy	50.0	75.55556	73.84615
RuleSet	100.0	100.0	100.0
Suite	ND	35.555557	49.23077

Legenda: [**ND**] Não disponível

possuem um valor de 0.42. Este cenário é perigoso pois indica que boa parte das relações entre os elementos da PLA são variáveis. De acordo com Zhang et. al. (2008), isso é um sintoma mal reúso dos componentes, o que indica que os desenvolvedores devem realizar melhorias nesta SPL.

A Tabela 5.6 apresenta os valores da métrica CRR para a GPL.

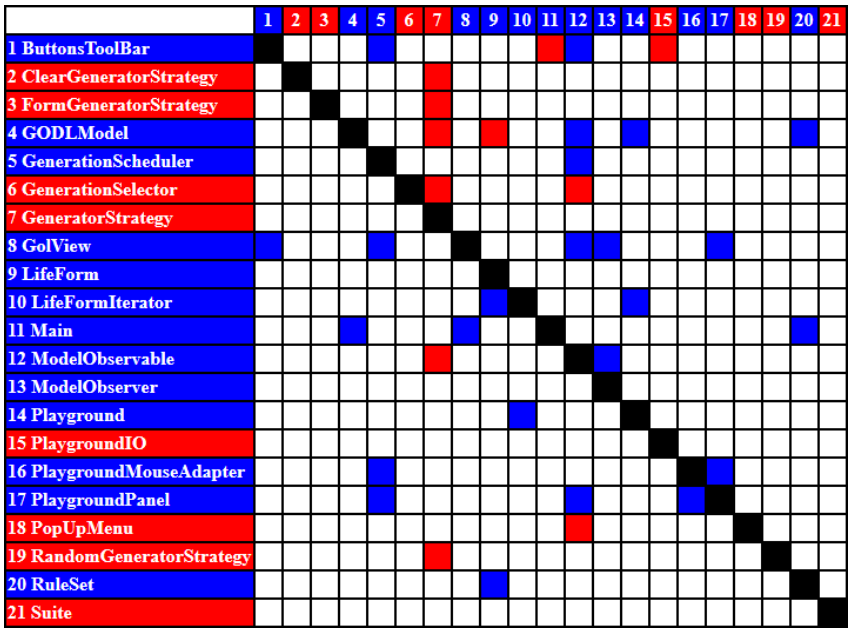

Figura 5.4 DSM da GOL

Os valores de CRR indicam que a maior parte dos elementos possuem um alto nível de reúso. Entretanto, identificamos que alguns destes elementos, como *CycleWorkSpace* e *GlobalVarsWrapper* apresentaram um baixo valor de CRR. Porém, a maioria das relações apresentaram baixos valores de CRR, estes valores indicam que a PLA possui falhas e uma baixa qualidade (ZHANG et al., 2008). A Figura 5.5 apresenta a DSM da PLA recuperada. Pudemos observar nesta DSM que as classes Graph e Vertex se relacionam com quase que todos os elementos da PLA, o que pode significar que estas classes podem apresentar o *smell* da *god class* (FOWLER, 2009).

5.4.6 Resultados para Prop4J

A SPL Prop4J não possui *features* mandatórias, implementando apenas *features* opcionais. O resultado das métricas refletem essas características (ver Tabelas 5.3 e 5.7). Além disso com a ausência de *features* mandatórias, o total de possíveis produtos que podem ser gerados por esta SPL chegou ao número de 5029 produtos. Por isso, utilizamos o método de geração de produtos T-Wise (HENARD et al., 2014), que gerou apenas 11 produtos para análise. Depois de realizar a recuperação com os 11 produtos gerados, foram identificados 1 elemento mandatório (*Node*) e 13 elementos variáveis. Todos relacionamentos entre estes elementos foram identificados como variáveis.

A Figura 5.6 apresenta da DSM da PLA recuperada do Prop4J. O elemento *Node* é o único elemento comum enquanto que todos os outros elementos e relações são variáveis, o que confirma os valores obtidos pela métrica CRR. Estes resultados sugerem que a PLA possui má qualidade e que esta SPL necessita de manutenção para que o grau de reúso dos componentes possa aumentar.

Identificamos neste projeto SPl duas classes centrais: *Node* e *Prop4JTest*, ambas as

Tabela 5.6 Valores da Métrica CRR para a SPL GPL

Element	$\mathbf{CRR_{par}}$	$\mathbf{CRR_{75}}$	$\mathbf{CRR_{todos}}$
CycleWorkSpace	ND	35.135136	38.064514
Edge	100.0	100.0	80.645164
EdgeIfc	100.0	100.0	100.0
EdgeIter	100.0	100.0	100.0
FinishTimeWorkSpace	ND	36.486485	46.451614
GlobalVarsWrapper	100.0	16.216215	15.483871
Graph	100.0	100.0	100.0
Main	100.0	100.0	100.0
Neighbor	100.0	100.0	100.0
NeighborIfc	100.0	100.0	100.0
NumberWorkSpace	100.0	64.86486	61.935486
Vertex	100.0	100.0	100.0
VertexIter	100.0	100.0	100.0
WorkSpace	100.0	100.0	100.0
WorkSpaceTranspose	ND	36.486485	46.451614

Legenda: [**ND**] Não disponível

classes se relacionam com quase todas as demais o que pode indicar que estas duas classes possuem o *smell* de *god class* (FOWLER, 2009) e devem ser refatoradas.

5.4.7 Respostas para as perguntas de pesquisa

Nos próximos parágrafos, discutiremos as respostas para as perguntas de pesquisa levantadas neste estudo.

RQ1: A precisão das métricas da PLA é afetada pelo número de produtos utilizado para comparação?

O número de produtos utilizado na comparação afeta a precisão das métricas recuperadas na PLA, confirmando a nossa hipótese H_{1a}. A Figura 5.7 apresenta os valores das métricas SSC (azul) e SVC (vermelho) para quatro SPLs (DPL, VOD, Zip Me e Prop4J) coletada durante diferentes estágios comparativos.

Como esperado, a quantidade de informação a respeito da variabilidade aumentou

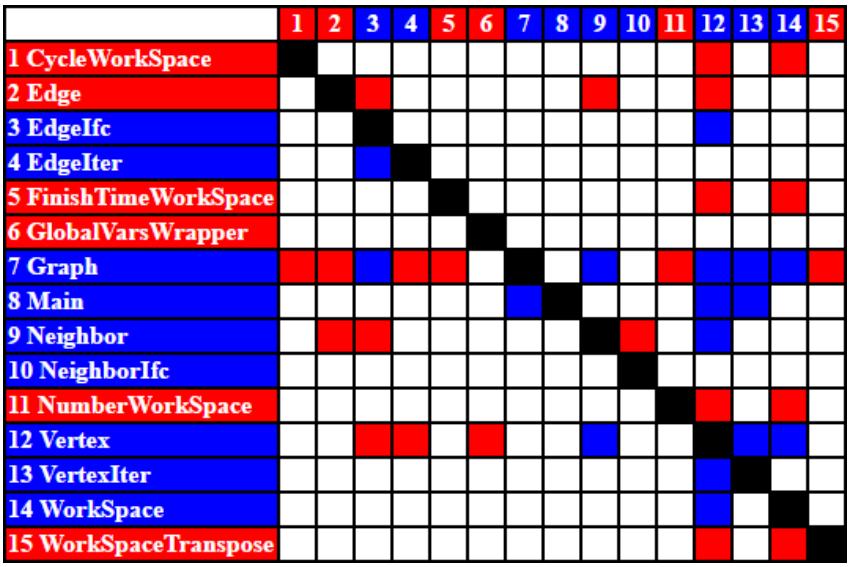

Figura 5.5 DSM da GPL

Tabela 5.7 Valores da métrica CRR para a SPL Prop4J

Element	**CRR_{par}**	**CRR_{8}**	**CRR_{todos}**
And	100.0	50.0	54.545456
AtLeast	50.0	37.5	36.363636
AtMost	50.0	37.5	36.363636
Choose	50.0	37.5	36.363636
Equals	100.0	37.5	27.272728
Implies	100.0	50.0	54.545456
Literal	100.0	87.5	90.909096
Node	100.0	100.0	100.0
NodeReader	100.0	62.5	54.545456
NodeWriter	50.0	75.0	54.545456
Not	50.0	50.0	36.363636
Or	50.0	37.5	36.363636
Prop4JTest	50.0	37.5	36.363636
SatSolver	50.0	37.5	45.454548

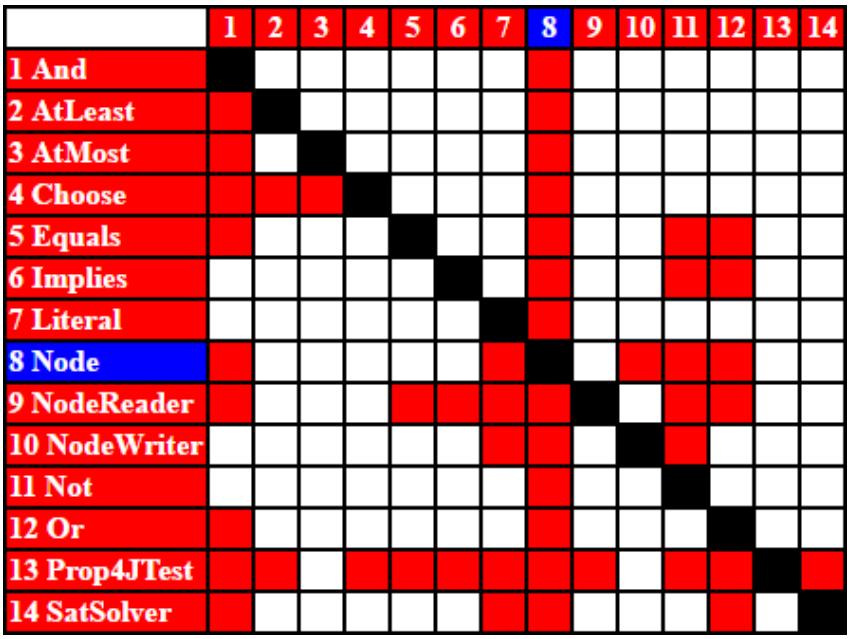

Figura 5.6 DSM da Prop4J

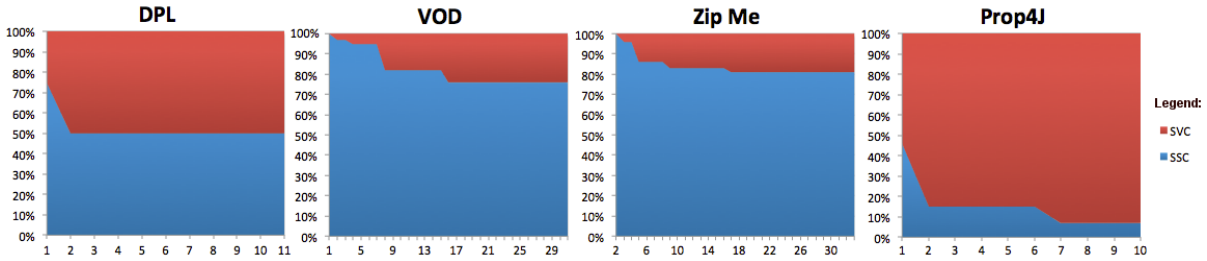

Figura 5.7 SVC vs SSC em quatro projetos

quando incluímos mais produtos na comparação. Entretanto, depois de uma determinada quantidade de comparações, percebemos que os valores das métricas estabilizaram. No caso da SPL Zip Me foram necessários apenas 18 produtos para capturar todos os detalhes de variabilidade desta PLA. Observamos o mesmo padrão nos demais projetos SPL.

Com o aumento do número de produtos na análise, os elementos identificados como comuns e variáveis bem como os valores das métricas coletadas tendem a se estabilizar. O conjunto de produtos (após estabilização das métricas) possuem uma estrutura comum, com variações presentes em detalhes de baixo nível (implementação de métodos, valores de atributos, entre outros).

RQ2: O reúso da PLA é afetado pelo número de *features* opcionais?

Na RQ2, verificamos se o número de *features* opcionais afeta o reúso da PLA. Por esta razão, apresentamos as seguintes hipóteses:

- H_{0b}: Todas as PLAs possuem o mesmo grau de reúso, e não há diferença estatística entre os valores.

- H_{1b}: Existe pelos menos um valor CRR diferente.

Utilizamos a combinação de todos os valores da métrica CRR de todos os projetos SPL para testar hipótese estatisticamente através de análise de gráficas, utilizando os testes ANOVA (Análise de Variância) e teste de Tukey (WOHLIN et al., 2012). Os testes apresentaram um nível de significância de 5%. O p-valor para o teste ANOVA foi de 7.43e-06, o que nos permitiu rejeitar a hipótese nula (H_{0a}), o que nos permitiu concluir que ao menos uma PLA possui um grau de reúso significativamente diferente das demais.

Para identificar as diferentes médias, aplicamos o teste de Tukey. Analisamos 15 comparações e apenas 4 delas apresentaram diferença significativa: (i) Prop4J–GOL (p = 6e-05), (ii) Prop4J–GPL (p = 9e-03), (iii) VOD–Prop4J (p = 5.2e-06), e (iv) ZipMe–Prop4J (p = 3.3e-06). Em outras palavras, a SPL Prop4J apresentou o pior grau de reúso entre todas as PLAs (Figura 5.8).

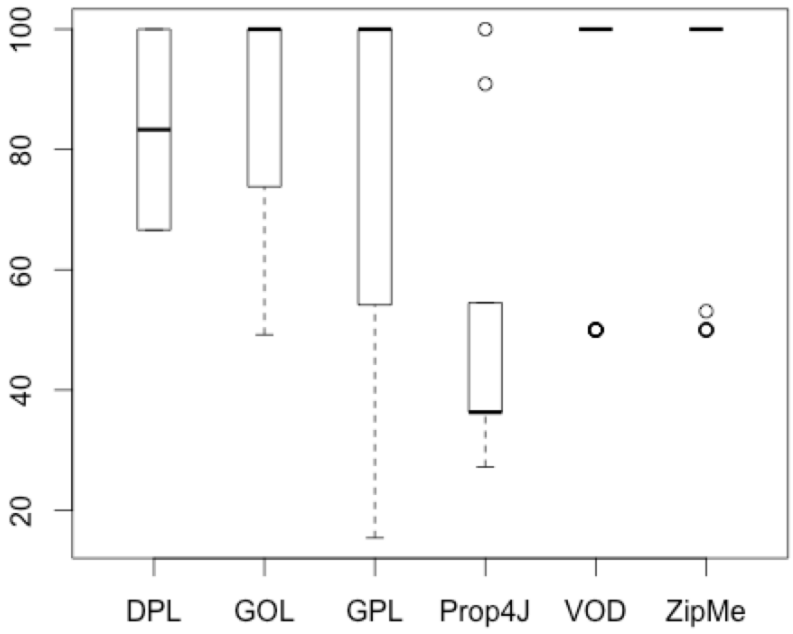

Figura 5.8 Boxplot comparando CRR por PLA

Por fim, identificamos que a quantidade de *features* opcionais impacta no grau de reúso. A SPL Prop4J apresentou os piores resultados pelo fato de todas as suas *features* serem opcionais o que permitia a instanciação de 5029 produtos. Este cenário demonstra a complexidade envolvida durante o desenvolvimento de um projeto SPL. O gerenciamento da variabilidade é uma tarefa complexa cujo resultado é refletido na PLA.

RQ3: Qual métrica melhor identifica o nível de reúso da PLA?

Duas métricas foram consideradas para responder esta questão de pesquisa, CRR e SSC, para tal as seguintes hipóteses foram formuladas:

- H_{0c}: Não existe diferença estatística significante entre as duas métricas.

- H_{1c}: A métrica CRR apresenta um melhor grau de reúso do que a métrica SSC.
- H_{2c}: A métrica SSC apresenta um melhor grau de reúso do que a métrica CRR.

Assim como na RQ2, combinamos todos os valores das métricas CRR e SSC das PLAs dos projetos SPL para testar as hipóteses estatisticamente utilizando o método Wilcoxon/Kruskal-Wallis, este método foi utilizado pois foi idealizado para comparar duas métricas diferentes, com valores não homocedásticos (WOHLIN et al., 2012). O p-valor ($p = 0.04113$) foi menor do que o grau de significância ($p = 0.05$), o que nos permitiu rejeitar a hipótese nula (H_{0b}). A Figura 5.9 apresenta o boxplot, onde podemos verificar a variação das médias dos valores da métrica CRR contra a métrica SSC para cada projeto SPL.

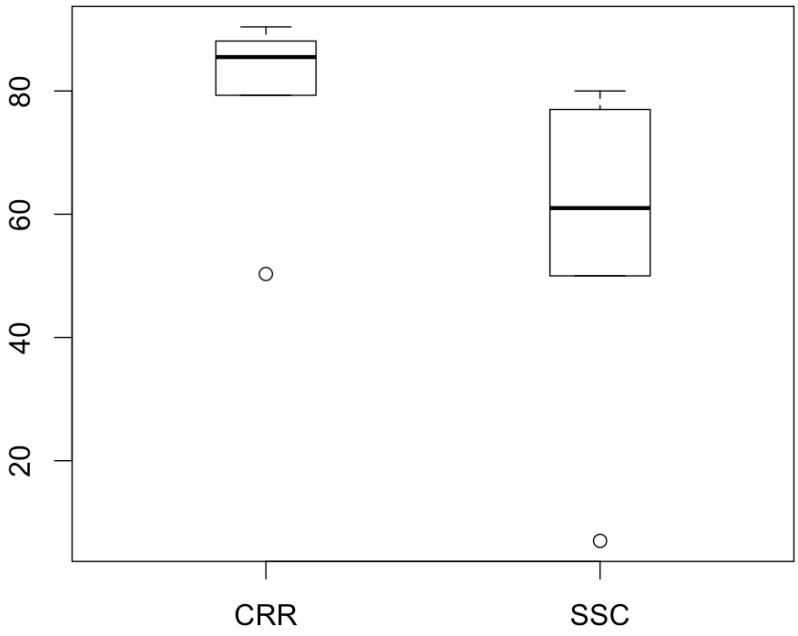

Figura 5.9 Boxplot SSC vs CRR

Por fim, identificamos que a métrica CRR dá uma visão melhor do grau de reúso dos componentes da PLA. A métrica calcula o grau de reúso de cada elemento individualmente e indica como os produtos estão reutilizando estes elementos. Ao contrário da CRR, a métrica SSC usa todos os elementos da PLA e fornece apenas uma visão geral sobre os elementos comuns a todos os produtos.

5.5 INTERPRETAÇÃO

Discutiremos nesta seção algumas descobertas deste estudo exploratório, as ameaças a validade e as limitações encontradas. Com exceção das SPL GPL e Prop4J, todas as demais apresentaram mais elementos comuns do que variáveis. Normalmente, os valores destas métricas indica que as SPL tiveram um bom design (ZHANG et al., 2008; JUNIOR; GIMENES; MALDONADO, 2008).

5.5.1 Descobertas Gerais

Correlação entre as métricas - Dos resultados obtidos de todas as seis SPL estudadas, percebemos que quando o valor da métrica SSC é alto, consequentemente os valores da métrica CRR para os elementos da PLA tendem a ser altos também. Esta pode ser uma evidência inicial de que existe uma correlação entre as métricas SSC e CRR.

Algumas métricas servem como apoio para outras métricas - As métricas *ClassMandatory* e *ClassOptional* são métricas que contam o número de classes mandatórias e opcionais da PLA, respectivamente. Estes valores foram usados para confirmar os resultados obtidos pelas métricas SSC e SVC. Já as métricas *ClassVP* e *ComponentVariable* indicaram se determinada classe ou componentet da PLA apresenta variabilidade.

Estas quatro métricas em conjunto servem como *backup* para os valores obtidos nas métricas SSC, SVC e CRR, onde os valores obtidos foram utilizados de forma a ratificar os resultados.

Espalhamento de *features* e grau de reúso - Os projetos com melhores resultados nas métricas (Zip Me, VOD e GOL respectivamente) foram os projetos que possuíam mais classes do que *features*, o que implica num maior espalhamento das mesmas por todas as classes da SPL.

Visualização dos resultados do processo de recuperação - A comparação de dois produtos evidencia as diferenças entre os dois, por outro lado, ao comparar N produtos, existe um maior nível de detalhamento nas informações recuperadas da PLA. Para PLA com um grande número de elementos e relações, a PLAR Tool gera uma DSM que fornece uma visão compactada da PLA. A DSM também facilita a visualização da informação, por exemplo, observando a Figura 5.5, os módulos *Graph* e *Vertex* apresentam relacionamentos com praticamente todos os demais módulos do sistema, o mesmo padrão pode ser observado na Figura 5.6, com os módulos *Node* e *Prop4JTest*. Os desenvolvedores da SPL e arquitetos podem usar essa informação para analisar o impacto de mudanças neste módulos (ex: Uma alteração no módulo *Node* pode afetar todos os outros módulos do sistema).

Correlação entre o número de produtos inseridos para análise e o valor da métrica CRR - Identificamos que a depender da quantidade de produtos inseridos para a recuperação o resultado das métricas sofrem alterações, a medida que quanto mais produtos são inseridos para análise, maior é a precisão da métrica CRR.

5.5.2 Ameaças a validade

Identificamos as seguintes ameaças a validade do estudo:

- **Ameaças a Validade Interna**
  - **Limitações da PLAR Tool -** As limitações da ferramenta discutidas no Capítulo 3, impactaram nos resultados deste estudo exploratório. Atualmente, a PLAR Tool apenas processa os arquivos MDG criados pelo STAN4J, o qual funciona apenas com projetos Java. Estas limitações reduzem o escopo do projetos SPL que podem ser estudados e avaliados com apoio da ferramenta.

- **Ameaças a Validade Externa**

  - **Ausência de SPL utilizada na indústria -** Apenas projetos SPL *open source*, desenvolvidos e utilizados no contexto acadêmico, foram utilizados neste estudo exploratório.
  - **Ausência de validação da PLA recuperada por desenvolvedores -** A PLA recuperada para cada projeto SPL estudado não foi avaliada por criadores ou desenvolvedores da SPL, como feito no estudo anterior (capítulo 5). Para minimizar este risco fizemos a recuperação manual de um dos projetos SPL para validar os resultados gerados pela ferramenta. A Tabela 5.8 apresenta a comparação entre os elementos detectados como comuns e variáveis entre a recuperação manual e a feita pela ferramenta para a SPL VOD.

Tabela 5.8 Comparativo entre recuperação Manual x PLAR Tool

DPL	Manual	PLAR Tool
Comuns	32	32
Variáveis	10	10

A Tabela mostra que não houve diferença entre a quantidade de elementos classificados como comuns e variáveis, o que minimiza os riscos, entretanto é necessário a validação da PLA pelos desenvolvedores a fim de confirmar a precisão dos resultados gerados. Planejamos entrar em contato com os desenvolvedores e arquitetos para avaliar a precisão e abrangência de cada PLA recuperada neste estudo.

  - **Geração dos produtos -** Alguns dos projetos SPL utilizados neste estudo não puderam ter seus produtos gerados, pois estes projetos não forneciam o código fonte da SPL e sim dos produtos somente. Desta forma não pudemos garantir que todas as configurações válidas foram contempladas durante a análise deste estudo.

5.6 TRABALHOS RELACIONADOS

Wu *et. al.* apresenta um abordagem semi-automática de recuperação de PLA (WU et al., 2011). Métricas foram definidas para detectar a similaridade em classes e métodos. Um estudo de caso foi realizado em uma SPL industrial. Em seu trabalho, foi assumido que produtos legado possuem design e implementação similares. Da mesma forma, esta suposição se mantém em nossa abordagem, porém os produtos são gerados de forma automática.

Losavio *et. al.* propôs um processo *bottom-up* para reconstruir a PLA a partir de produtos similares de um mesmo domínio (LOSAVIO et al., 2013), que é expresso em visões lógicas da UML. O foco do seu trabalho foi a construção e representação de PLA

candidatas seguido de um processo para obter a PLA definitiva. O processo de refatoração foi aplicado em um estudo de caso na indústria de robótica. O foco de nosso trabalho é a avaliação da PLA recuperada baseada em métricas de reúso.

Torkamani *et. al.* apresentam um novo atributo de qualidade para linhas de produto de software, "extratibilidade"(em inglês, *extractability*) (TORKAMANI, 2014). Uma métrica para extratibilidade é apresentada, bem como alguns atributos de qualidade. Ela é calculada com base no peso dos componentes reusáveis em relação com o peso de todos os componentes, um processo similar ao cálculo da métrica CRR utilizada neste trabalho. Porém, a métrica CRR calcula o nível de presença de um elemento e não o seu o peso. A efetividade da métrica extratibilidade de seis projetos SPL de uma empresa de telecomunicação iraniana foram avaliados neste estudo.

5.7 CONSIDERAÇÕES

Neste capítulo apresentamos um estudo exploratório com o objetivo de avaliar a qualidade da arquitetura recuperada de seis projetos SPL *open source*. Quatro dos seis projetos apresentaram uma qualidade arquitetural alta de acordo com as métricas utilizadas para avaliar este aspecto.

Detectamos que um dos projetos era composto apenas por *features* opcionais o que acarretou numa grande quantidade de possíveis produtos a serem gerados, o que nos fez utilizar o método de geração de produtos T-Wise para tornar a análise deste projeto viável. Além disso pudemos detectar que a quantidade de produtos utilizadas para análise influencia o valor das métricas, em especial a CRR, sendo este um fator determinante para mensurar o grau de reúso dos componentes da PLA e consequentemente a sua qualidade.

A principal contribuição deste estudo é a disponibilização da documentação de PLA para os seis projetos SPL estudados, até então não disponível. Os dados de replicação deste estudo encontram-se disponíveis no enderenço: http://v.ht/pMLs.

Finalmente, a necessidade de utilizar o método de geração de produtos T-Wise (PERROUIN et al., 2010a) para viabilizar a reconstrução de PLA do projeto Prop4J nos motivou a realizar outro estudo empírico, utilizando o método T-Wise e o método padrão de geração de produtos, com o objetivo de comparar a arquitetura recuperada por cada método e investigar se há diferenças entre os valores das métricas obtidos. Um artigo sobre os resultados deste estudo está em preparação para submissão em evento científico.

Capítulo

6

AVALIAÇÃO DO MÉTODO DE GERAÇÃO PADRÃO E DO MÉTODO T-WISE

Este capítulo apresenta um estudo exploratório para comparar o resultado das métricas e da PLA recuperada, realizando a extração a partir de todas as configurações válidas versus configurações obtidas pelo método de geração T-Wise (PERROUIN et al., 2010a). O processo de recuperação é o mesmo utilizado no primeiro estudo (capítulo 5).

A seção 6.1 apresenta a motivação para a realização deste estudo, seus objetivos e o seu contexto. A seção 6.2 apresenta o planejamento feito para condução deste estudo exploratório. A seção 6.3 apresenta os processos de prepação para a execução do estudo. A seção 6.4 apresenta a análise individual de cada projeto SPL A seção 6.5 apresenta a análise dos dados coletados durante o estudo. A seção 6.6 apresenta os trabalhos relacionados e por fim a seção 6.7 apresenta as conclusões e trabalhos futuros.

6.1 MOTIVAÇÃO

No estudo anterior, durante a análise da RQ2 (*O nível de reúso dos componentes da PLA é afetado pelo número de features opcionais?*) percebemos que em todas as SPL analisadas, após um determinado número de produtos inseridos para análise, os valores das métricas tendiam a se estabilizar. Aliado a isso, percebemos também que vários produtos gerados por estas SPL possuíam arquiteturas contendo os mesmos módulos e relações. Fazendo uma investigação sobre estes produtos verificamos que a variabilidade estava implementada nos métodos e atributos, dessa forma não sendo mapeada pela visão de módulo fornecida pela ferramenta.

Em nosso estudo anterior, consideramos todas as configurações válidas para geração de produtos, sem priorizar e utilizar configurações para geração de produtos mais significativo ou representativos. Para a SPL Prop4J, que não possui *features* mandatórias, 5029 produtos poderiam ser gerados para a recuperação de PLA, comprometendo o processo de extração de arquivos MDG e a consequente recuperação da PLA.

Neste contexto, decidimos utilizar o método de geração de produtos T-Wise (HENARD et al., 2014), que busca gerar apenas as configurações de produtos mais significativas, o que contribui para a diminuição de produtos gerados que serão utilizados para análise. O método T-Wise originalmente foi utilizado para testes de projetos SPL. Os autores do estudo identificaram que em grande parte dos projetos SPL, a quantidade de produtos que pode ser gerada é muito grande, o que acaba pode dificultar o processo de testar cada produto individualmente. Dessa forma era necessário escolher um subconjunto de produtos que representassem bem a linha para tal tarefa. O mesmo problema foi encontrado no processo de recuperação da técnica PLAR. Alguns projetos SPL podem gerar muitos produtos, o que dificulta o processo de extração de dependências e a recuperação da PLA, desta forma o método T-Wise se mostra como possível solução para este problema além de trazer benefícios como: reduzir o tempo de recuperação da PLA, diminuir a quantidade de produtos gerados.

6.1.1 Definição do problema

A geração de um produto de um projeto SPL é realizada a partir da escolha das *features* que irão compor o produto. A depender da quantidade e dos tipos relacionamentos entre as *features*, uma SPL pode gerar muitos produtos.

Em nosso estudo anterior, verificamos que em diversas situações, a escolha de determinados conjuntos de *features* distintos geravam a mesma arquitetura; as diferenças entre um produto e outro eram visíveis em detalhes do código-fonte, tais como implementação de métodos, valores de atributos, entre outros.

Neste contexto, verificamos que não seria necessário utilizar todos os produtos que podem ser gerados a partir de configurações válidas para recuperar a PLA e sim um subconjunto que apresentasse arquiteturas relevantes para a recuperação de PLA a partir de produtos da SPL.

O método de geração de produtos T-Wise (HENARD et al., 2014) gera um conjunto de produtos com as configurações de *features* consideradas mais significativas para o projeto SPL, e assim, apoiando a geração de uma quantidade de produtos menor para a recuperação da PLA quando comparado ao método de geração de produtos padrão (todos os produtos).

6.1.2 Objetivos de Pesquisa

O método GQM (BASILI; CALDIERA; ROMBACH, 1994) foi utilizado para definir o processo que guiou a condução do estudo, de natureza exploratória.

Objetivo (*Goal*) - O Objetivo deste estudo exploratório foi comparar os resultados obtidos pela recuperação utilizando todos os produtos gerados contra produtos gerados pelo método T-Wise, com o propósito de avaliação no que diz respeito aos valores das métricas obtidas do ponto de vista do pesquisador no contexto de projetos SPL *open source*. Para alcançar este objetivo definimos as seguintes questões:

- **RQ1:** Os valores das métricas coletadas durante o processo de recuperação são afetados pelo método de geração de produtos? O método de geração onde todos

os produtos são gerados (padrão) nos apresenta os valores das métricas esperados. Com esta RQ estamos investigando se através do método T-Wise, o valor das métricas obtidos sofre alterações significativas em relação as métricas obtidas ao utilizar o método de geração de produtos padrão.

- **RQ2:** A variabilidade e comunalidade detectada na PLA é afetada pelo método de geração de produtos? Com esta RQ investigamos se ao utilizar o método de geração T-Wise encontraremos os mesmos elementos comuns e variáveis na PLA recuperada.

6.1.3 Contexto

Dez projetos SPL foram escolhidos, todos coletados do site SPL2Go[1], que é um repositório público de projetos SPL para realização de estudos. Foram escolhidos apenas projetos desenvolvidos em Java, devido a limitação imposta pelo STAN4J, e que tivessem sido construídos usando o FeatureIDE, para poder realizar a geração de produtos usando o método T-Wise e o método padrão. Os 10 projetos SPL utilizado neste estudo foram:

- *BankAccount* - Uma SPL onde cada produto possui funções para gerenciar uma conta bancária.
- *BankAccountV2* - Esta SPL é uma extensão da SPL BankAccount.
- *DesktopSearcher* - Esta SPL gera programas para realizar indexação de arquivos e realização de buscas.
- *Elevator* - SPL que simula o funcionamento de um elevador. Utilizada para avaliação de checagem de modelos.
- *E-Mail* - SPL que implementa um serviço de servidor de emails.
- *ExamDB* - SPL que gerencia um banco de dados de provas a serem enviadas para alunos de uma instituição de ensino.
- *Graph Product Line (GPL)* - SPL que implementa geração de grafos.
- *PayCard* - Uma SPL que gera aplicações para gerenciamento de cartões de crédito.
- *PokerSPL* - Uma SPL que suporta as diferentes variações de Poker.
- *UnionFind* - Uma SPL que implementa os algoritmos do tipo UnionFind.

A Tabela 6.1 apresenta mais informações sobre estes projetos SPL, a saber, o número de *features* de cada projeto, total de produtos gerados, produtos gerados utilizando o método T-Wise, quantidade de classes de cada projeto e linhas de código.

[1] http://spl2go.cs.ovgu.de/

Tabela 6.1 Projetos SPL analisados neste estudo

	#F	*#P*	*#TP*	*#C*	*#LOC*	*Gener.*
BankAccount	6	24	6	2	110	FeatureHouse
BankAccountV2	8	72	8	3	131	FeatureHouse
DesktopSearcher	22	462	9	14	3779	AHEAD
Elevator	6	20	7	5	1046	FeatureHouse
E-Mail	9	40	6	3	1233	FeatureHouse
ExamDB	3	8	6	4	325	FeatureHouse
GPL	38	156	20	16	840	FeatureHouse
PayCard	3	6	5	7	441	FeatureHouse
PokerSPL	11	28	10	8	274	FeatureHouse
UnionFind	10	6	6	4	169	FeatureHouse

Legenda:[**#F**]Features [**#P**]Produtos [**#TP**]Produtos T-Wise [**#C**]Classes [**#LOC**]Linhas de Código [**Gener.**]Gerador de Produtos

6.2 PLANEJAMENTO DO ESTUDO

6.2.1 Formulação de Hipóteses

Foram formuladas as seguintes hipóteses para cada RQ:

- **RQ1 - Os valores das métricas coletadas durante o processo de recuperação são afetados pelo método de geração de produtos?**
  - H_{0a}: Os valores das métricas coletadas durante o processo de recuperação não são afetados pelo método de geração de produtos escolhido.
  - H_{1a}: Os valores das métricas coletadas durante o processo de recuperação são afetados pelo método de geração de produtos escolhido.
- **RQ2 - A variabilidade e comunalidade detectada na PLA é afetada pelo método de geração de produtos?**
  - H_{0b}: O método de geração não afeta a detecção.
  - H_{1b}: O método de geração afeta a detecção.

6.2.2 Instrumentação

Para cada método de geração de produtos utilizado, o processo de recuperação descrito no Capítulo 3 foi seguido.

6.2.3 Procedimento de Coleta de Dados

Neste estudo buscamos comparar o resultado da PLA gerada utilizando dois métodos de geração de produtos diferentes. Desta forma foram coletadas as métricas e as visualizações geradas pela ferramenta em dois estágios: Utilizando todos os produtos possíveis de serem gerados e utilizando todos os produtos gerados pelo método T-Wise. Para os projetos SPL em foi detectada variabilidade na PLA, alguns estágios adicionais foram acrescentados - usando 25% do total de produtos, 50% e 75%, todos escolhidos de forma arbitrária.

6.2.4 Procedimento de Análise

Para cada método de geração de produtos empregado, os valores das métricas obtidos foram comparados. Os arquivos de visualização gerados também foram comparados a fim de encontrar possíveis divergências entre as PLAs geradas pelos dois métodos

6.2.5 Avaliação da Validade do Estudo

Este estudo de caso foi avaliado quanto aos aspectos de validade externos (representatividade dos projetos analisados) e sobre os aspectos de confiabilidade dos dados coletados e analisados, a discussão sobre estas ameaças a validade encontra-se na Seção 6.5 .

6.3 EXECUÇÃO

6.3.1 Preparação

Neste estudo foram utilizados dois métodos de geração de produtos diferentes: O convencional e o método de geração de produtos T-Wise (HENARD et al., 2014). Cada método gerou um conjunto de produtos que foram analisados pelas ferramentas de extração de código para obtenção dos arquivos MDG.

Dessa forma, ao final do processo de geração de produtos e extração, obtivemos dois conjuntos de arquivos para análise e extração da PLA pela PLAR-Tool: os arquivos MDG que representam os produtos gerados pelo método convencional e os arquivos MDG que representam os produtos gerados pelo método T-Wise.

6.3.2 Coleta de Dados Realizada

Os dados utilizados para avaliar a PLA recuperada foram coletados a partir das métricas geradas pela ferramenta PLAR-Tool. Ao fim da análise cada projeto SPL, um relatório com as métricas coletadas foi gerado. Os dados contidos neste relatório foram utilizados para realizar a análise comparativa das PLAS obtidas através dos dois métodos de geração de produtos.

6.4 ANÁLISE

A Tabela 6.2 apresenta o comparativo entre os valores da métrica SSC e SVC utilizando todas as configurações válidas versus utilizando as configurações geradas pelo T-Wise.

Com exceção de um projeto SPL (GPL), todos demais tiveram resultados iguais para as métricas SSC e SVC. As configurações válidas usadas pelo método padrão e as configurações usadas pelo método T-Wise geraram subconjuntos de produtos representativos da variabilidade da SPL e levaram à recuperação de PLA com estruturas semelhantes. Discutiremos a seguir os resultados da métrica CRR de cada SPL.

Tabela 6.2 Comparativo SSC vs SVC

	Normal		**T-Wise**	
	SSC	**SVC**	**SSC**	**SVC**
BankAccount	1	0	1	0
BankAccountV2	0,66	0,34	0,66	0,34
DesktopSearcher	0,26	0,74	0,26	0,74
Elevator	1	0	1	0
E-Mail	1	0	1	0
ExamDB	1	0	1	0
GPL	0,6	0,4	0,5	0,5
PayCard	0,72	0,28	0,72	0,28
PokerSPL	0,5	0,5	0,5	0,5
UnionFind	1	0	1	0

6.4.1 Resultados para BankAccount

A SPL *BankAccount* é uma SPL voltada para o gerenciamento de contas bancárias, é uma SPL simples, sua PLA é composta apenas por componentes e relações mandatórias, o que indica que a variabilidade desta linha está inserida nos valores que os atributos recebem e na implementação dos métodos, para confirmar isto fizemos uma verificação no código fonte dos produtos e constatamos que de fato a variabilidade da linha estava implementada neste nível. Este tipo de variabilidade não é mapeado pela PLAR Tool.

A Tabela 6.3 apresenta o comparativo da métrica CRR utilizando o método de geração de produtos convencional (todas as configurações válidas) contra o método T-Wise.

Independentemente do método empregado para geração de produtos, o resultado obtido nas métricas é o mesmo. Não houve diferença entre as PLAs recuperadas.

Tabela 6.3 Comparativo CRR BankAccount

CRR	Normal	T-Wise
Account	100%	100%
Application	100%	100%

6.4.2 Resultados para BankAccountV2

A SPL *BankAccountV2* é uma extensão da SPL BankAccount, adicionando novas funcionalidades a manipulação de contas bancárias, assim como sua predecessora, a *BankAccountV2* possui um tamanho pequeno comparado aos demais projetos. A Tabela 6.4 apresenta o comparativo da métrica CRR para os dois métodos de geração de produtos.

Tabela 6.4 Comparativo CRR BankAccountv2

CRR	Normal 25%	Normal 50%	Normal 75%	Normal 100%	T-Wise
Account	100%	100%	100%	100%	100%
Application	100%	100%	100%	100%	100%
Transaction	-	16%	27%	33%	37%

Observando a tabela verificamos que há uma estabilização na identificação dos elementos da PLA a partir do momento que inserimos metade dos produtos para análise, o que dá um total de 36 produtos. Entretanto o método T-Wise se mostra vantajoso, pois gera apenas 8 produtos obtendo os mesmo resultados.

Não houve diferença em relação a identificação dos elementos, todos foram classificados da mesma forma, o que implica no mesmo resultado da métrica SSC e SVC para os dois métodos de geração de produtos. Entretanto existe uma diferença de valores da métrica CRR para a classe *Transaction*, essa diferença ocorre devido a quantidade de produtos analisada, o que acaba impactando nos resultados, apesar disso, a diferença entre os valores não foi significativa (4%).

A Figura 6.1 apresenta a PLA recuperada deste projeto SPL.

6.4.3 Resultados para DesktopSearcher

Esta foi a maior SPL analisada em termos de quantidade de produtos gerados (462) e também que apresentou a maior diferença entre a quantidade de produtos gerados utilizando o método convencional contra o método T-Wise (462 contra 9). Devido a quantidade de elementos da PLA, a tabela comparativa da métrica CRR entre os dois métodos de geração de produtos não foi inserida neste texto, a mesma pode ser visualizada no material de replicação.

Assim como no caso da SPL *BankAccountV2* não houve diferença em relação a identificação dos elementos, o que é refletido e confirmado nas métricas SSC e SVC. As

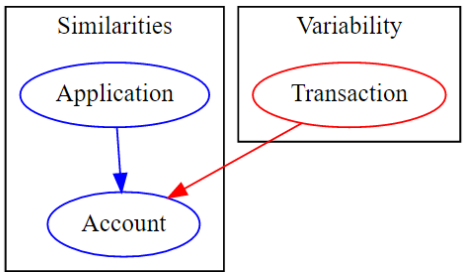

Figura 6.1 BankAccountv2 PLA

métricas CRR variam como esperado uma vez que a diferença de produtos analisado nos dois métodos foi muito alta.

6.4.4 Resultados para Elevator

A SPL Elevator é uma SPL voltada para fins educacionais, afim de realizar checagem de modelos de software. É uma SPL de tamanho mediano, considerando, *features* e quantidade de classes se comparada as demais. A Tabela 6.5 mostra o comparativo da métrica CRR para os dois métodos de geração de produtos.

Tabela 6.5 Comparativo CRR Elevator

CRR	Normal	T-Wise
Actions	100%	100%
Elevator	100%	100%
Environment	100%	100%
EvilPerson	100%	100%
Floor	100%	100%
JUnit_Scenario_Tests	100%	100%
PL_Interface	100%	100%
PL_Interface_impl	100%	100%
Person	100%	100%
SpecificationException	100%	100%
SpecificationManager	100%	100%

Novamente, não houve diferença entre os valores obtidos para o método normal e para o método T-Wise, neste caso todos os elementos estão presentes em todos os produtos o que nos permite concluir que a variabilidade está implementada nas variáveis e no funcio-

namento dos métodos destas classes. Entretanto o método T-Wise necessitou apenas de 7 produtos para realizar a recuperação, representando uma redução de 65% na quantidade de produtos analisados produzindo os mesmos resultados.

6.4.5 Resultados para E-mail

A SPL Email é uma linha de produto de pequeno porte, contém poucas *features* e classes, seus produtos implementam um gerenciador de emails. A Tabela 6.6 apresenta o comparativo da métrica CRR para os dois métodos de geração de produtos.

Tabela 6.6 Comparativo CRR E-Mail

CRR	Normal	T-Wise
Client	100%	100%
Email	100%	100%
Util	100%	100%

Não houve diferença entre os valores CRR obtidos, uma vez que todos os elementos foram classificados da mesma forma e possuem o mesmo valor da métrica CRR para os dois métodos de geração. Assim como outros projetos que apresentaram este comportamento, acreditamos que a variabilidade está na implementação dos métodos e/ou no valor das variáveis. Novamente o método T-Wise conseguiu reduzir o número de produtos para 6, representando uma redução de 85% do total, apresentando os mesmos resultados.

A Figura 6.2 apresenta a PLA recuperada deste projeto SPL.

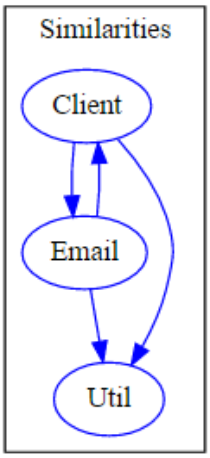

Figura 6.2 Email PLA

6.4.6 Resultados para ExamDB

A SPL ExamDB gera produtos cujo objetivo é o gerenciamento de banco de dados contendo avaliações e notas de estudantes de uma instituição de ensino. É uma SPL de pequeno porte comparada as demais. A Tabela 6.7 apresenta o comparativo da métrica CRR para os dois métodos de geração de produtos.

Tabela 6.7 Comparativo CRR ExamDB

CRR	**Normal**	**T-Wise**
ExamDataBase	100%	100%
ExamDataBaseException	100%	100%
ExamDataBaseImpl	100%	100%
Student	100%	100%

Novamente temos mais um exemplo de PLA onde a variabilidade não está explícita na arquitetura mas sim nos detalhes de implementação das classes. Não houve diferença nos valores das métricas, bem como na classificação dos elementos.

6.4.7 Resultados para GPL

Esta é a única SPL do estudo anterior utilizada neste estudo, considerando a quantidade de produtos gerados, temos uma SPL de grande porte. Porém esta SPL apresentou algo anormal em relação as outras, o método T-Wise gerou uma PLA com mais elementos do que o método convencional, como pode ser verificado na Tabela 6.8. Além disso a tabela apresenta também um comparativo do CRR capturado no método normal em quatro estágios: usando 25% dos produtos, 50%, 75% e 100%. Fizemos este comparativo para todos os projetos SPL que apresentaram variabilidade na PLA.

Verificando as informações contidas na tabela, podemos verificar que só houve estabilização dos elementos classificados como comuns e variáveis a partir do momento que são utilizados 75% dos produtos para análise. Isto apresenta uma redução em relação ao total de produtos, entretanto o método T-Wise ainda leva vantagem ao utilizar apenas 20 produtos, que é uma quantidade inferior a 25% do total de produtos gerados e alcança os mesmos resultados. Como observado na tabela com exceção do elemento *RegionWorkSpace*, todos os demais foram classificados igualmente, a diferença dos valores de CRR se dá por conta da quantidade de produtos inserida para análise da PLAR Tool, o que impacta diretamente no valor da métrica. O fato da classe *RegionWorkSpace* estar presente em uma PLA e não em outra gerou diferenças nas representação como pode ser visto na Figura 6.3 que apresenta o DSM das duas PLA geradas.

Na figura podemos observar que além de ter um novo elemento, a PLA gerada pelo T-Wise também apresenta relacionamentos para este elemento, não conseguimos apontar uma causa precisa para este fenômeno, no entanto verificamos que este projeto SPL possui

Tabela 6.8 Comparativo CRR GPL

CRR	Normal 25%	Normal 50%	Normal 75%	Normal 100%	T-Wise
CycleWorkSpace	-	38%	41%	38%	50%
Edge	100%	100%	90%	80%	65%
EdgeIfc	100%	100%	100%	100%	100%
EdgeIter	100%	100%	100%	100%	100%
FinishTimeWorkSpace	38%	38%	41%	46%	20%
GlobalVarsWrapper	30%	15%	20%	15%	35%
Graph	100%	100%	100%	100%	100%
Main	100%	100%	100%	100%	100%
Neighbor	100%	100%	100%	100%	80%
NeighborIfc	100%	100%	100%	100%	100%
NumberWorkSpace	69%	61%	69%	61%	60%
RegionWorkSpace	-	-	-	-	25%
Vertex	100%	100%	100%	100%	100%
VertexIter	100%	100%	100%	100%	100%
WorkSpace	100%	100%	100%	100%	100%
WorkSpaceTranspose	38%	38%	41%	46%	20%

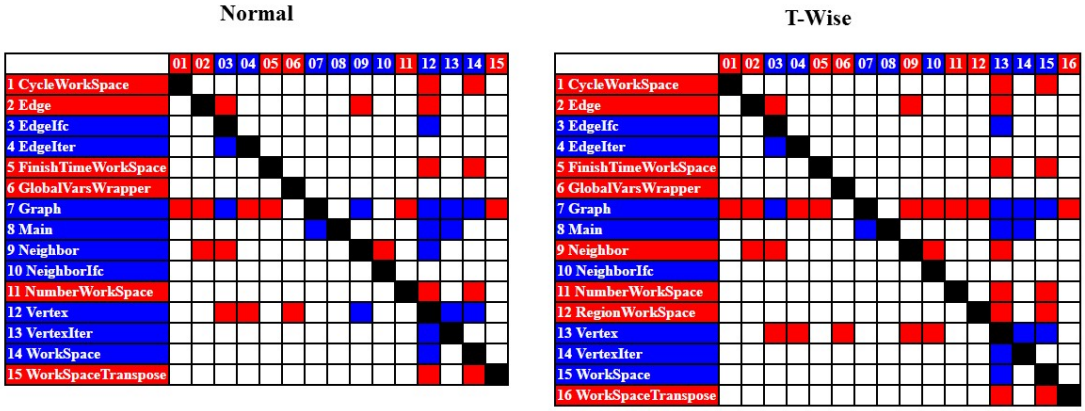

Figura 6.3 Comparativo entre as PLAs da SPL GPL

muitas restrições de relacionamento entre *features*, o que nos faz acreditar que isto seja um bug na ferramenta FeatureIDE. Estes fatos foram reportados para os desenvolvedores da SPL e a suspeita do bug foi informada aos desenvolvedores do FeatureIDE, porém não obtivemos resposta.

6.4.8 Resultados para PayCard

A SPL PayCard gera produtos cuja responsabilidade é gerenciar um cartão de crédito. É uma SPL pequena em termos de números gerados, a Tabela 6.9 apresenta o comparativo da métrica CRR para os dois métodos de geração.

Tabela 6.9 Comparativo CRR PayCard

CRR	Normal 25%	Normal 50%	Normal 75%	Normal 100%	T-Wise
CardException	100%	100%	100%	100%	100%
ChargeUI	100%	100%	100%	100%	100%
IssueCardUI	100%	100%	100%	100%	100%
LogFile	50%	66%	60%	66%	60%
LogRecord	50%	66%	60%	66%	60%
PayCard	100%	100%	100%	100%	100%
Start	100%	100%	100%	100%	100%

Nesta SPL os elementos identificados como comuns e variáveis estabilizam ao usar apenas 25% dos produtos, o que confere uma vantagem ao método normal para o caso desta SPL, ao contrário das demais. Consideramos que isto ocorre por esta SPL possuir uma pequena quantidade de produtos gerados (6) o que contribui para uma estabilização da PLA recuperada utilizando poucos produtos.

Não houve diferença em relação a classificação dos elementos da PLA nos dois casos, houve uma pequena diferença nos valores da métrica CRR para os elementos *LogFile* e *LogRecord* de 6%, comparando com os valores obtidos quando utilizados todos os produtos gerados, novamente isso deve ao fato da quantidade de produtos inseridas para análise ser diferente, impactando portanto o valor da métrica.

6.4.9 Resultados para PokerSPL

A PokerSPL simula em seus produtos todas as variantes do jogo Poker, é uma SPL mediana em termos de produtos de gerados. A Tabela 6.10 apresenta o comparativo da métrica CRR para os dois métodos de geração de produtos.

Tabela 6.10 Comparativo CRR PokerSPL

CRR	Normal 25%	Normal 50%	Normal 75%	Normal 100%	T-Wise
Blind	28%	42%	52%	57%	50%
Button	100%	100%	100%	100%	100%
Card	42%	50%	66%	75%	80%
Dealer	100%	100%	100%	100%	100%
Deck	42%	50%	66%	75%	80%
Player	100%	100%	100%	100%	100%
Pot	42%	71%	80%	85%	60%
Table	100%	100%	100%	100%	100%

Em relação a classificação dos elementos das duas PLA, não encontramos diferença. Verificamos também que a identificação dos elementos comuns e variáveis estabiliza ao utilizar 25% do total de produtos gerados para recuperar a PLA, o que dá uma leve vantagem em relação ao método T-Wise (7 produtos contra 10 produtos). Entretanto o valor da métrica CRR obtida pelo método T-Wise é mais próxima dos valores obtidos usando todos os produtos do que quando recuperar a PLA usando apenas 25% dos produtos.

A Figura 6.4 apresenta a PLA recuperada deste projeto SPL.

6.4.10 Resultados para UnionFind

Os produtos desta SPL implementam várias versões do algoritmo UnionFind(SCHRIJVERS; FRÜHWIRTH, 2006) em seus produtos, onde cada produto implementa uma variante

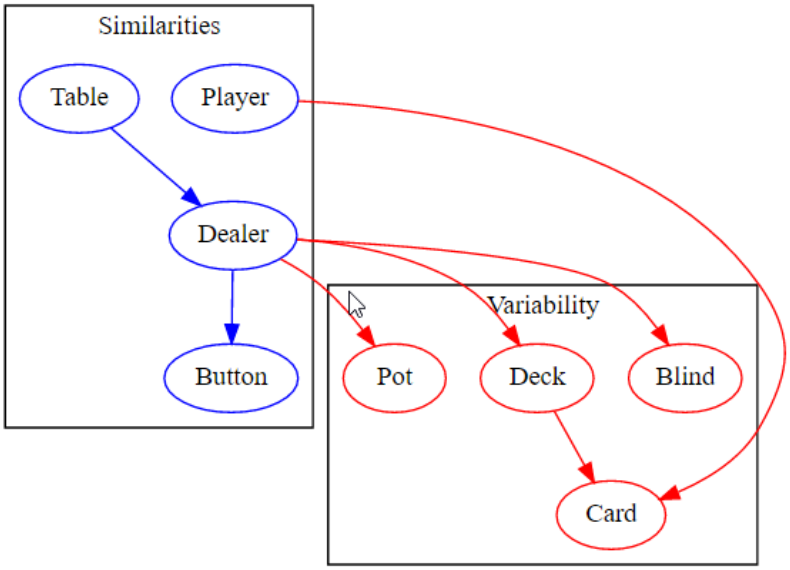

Figura 6.4 PokerSPL PLA

deste algoritmo. Foi a única SPL onde não houve diferença entre a quantidade de produtos gerados pelo método convencional e pelo método T-Wise, a Tabela 6.11 apresenta o comparativo da métrica CRR para os dois métodos.

Tabela 6.11 Comparativo CRR UnionFind

CRR	Normal	T-Wise
Stopwatch	100%	100%
TestPerformance	100%	100%
TestUF	100%	100%
UnionFind	100%	100%

Novamente temos um caso onde não há diferença entre as duas PLAs, por se tratar de uma PLA que implementa várias versõs diferentes de um mesmo método, a variabilidade está implementa no comportamento de cada um deste métodos, o que acaba não sendo refletido na PLA.

6.5 INTERPRETAÇÃO

Esta seção discute as respostas para as perguntas de pesquisa e os riscos a validade deste estudo. A PLA foi recuperada para todas as SPLs estudadas, com exceção para a PLA GPL, tendo como base produtos gerados tanto pelo método T-Wise, como pelo método

padrão.

6.5.1 Respostas para as perguntas de pesquisa

RQ1: A precisão das métricas coletadas pela PLAR Tool é afetada pelo método de geração de produtos?

Os valores da métrica CRR foram afetados pelo método de geração de produtos usado. O valor de CRR é calculado com base na quantidade de produtos inseridos para análise, e consequentemente, a redução no número de produtos usados com o método T-Wise interferiu no resultado desta métrica.

As métricas SSC e SVC são calculadas com base na quantidade de elementos comuns e variáveis encontrados na PLA. Não houve diferença na precisão das métricas coletadas. Possivelmente, as configurações válidas usadas pelo método padrão e as configurações usadas pelo método T-Wise geraram subconjuntos de produtos representativos da variabilidade da SPL e levaram à recuperação de PLA com estruturas idênticas. Porém, é importante destacar o caso atípico da SPL GPL, onde o método T-Wise foi capaz de detectar mais elementos na PLA do que o método padrão. Para este caso específico, houve alteração no valor das métricas SSC e SVC para produtos gerados com o método T-Wise (0,5 e 0,5) e o método padrão (0,6 e 0,4), como mostra a Tabela 6.2.

Com base nos resultados obtidos, pudemos verificar que existe um impacto no valor da métrica CRR em relação aos dois métodos. Entretanto é importante destacar que este impacto só pode ser notado para os elementos identificados como variáveis, uma vez que os elementos considerados comuns possuem o mesmo valor para os dois métodos de geração. A Figura 6.5 apresenta o comparativo da métrica CRR obtida para cada elemento da PLA utilizando o método padrão e o método T-Wise, para o projeto PokerSPL. Além desta

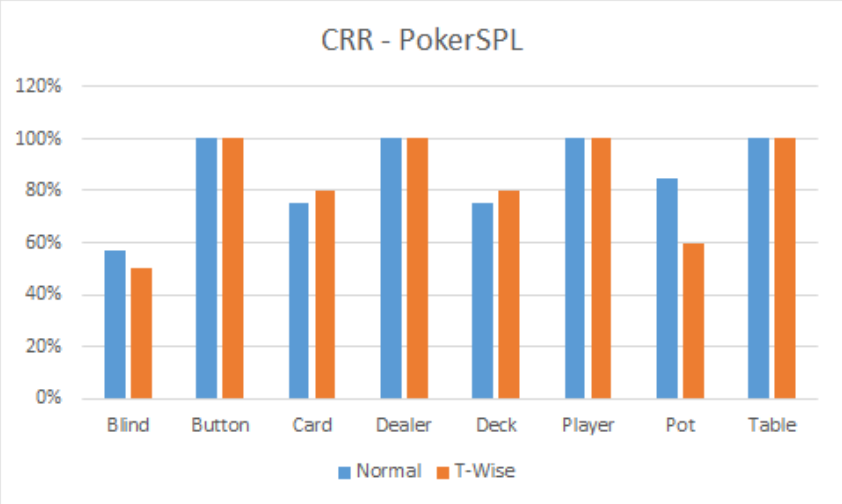

Figura 6.5 Compartivo CRR - PokerSPL

SPL outras quatro apresentaram elementos variáveis em sua arquitetura, escolhemos a PokerSPL pois o cenário observado neste projeto é semelhante aos demais. Esta SPL apresenta 4 elementos variáveis em sua PLA, ao observar o valor das métricas verificamos

que a diferença média entre o valor das métricas não ultrapassa 8% o que é um valor baixo, dessa forma não impactando dessa forma na precisão da métrica.

Da mesma forma nas métricas SCC e SVC, se desconsiderarmos o caso atípico da SPL GPL, onde o método T-wise detectou mais elementos que o método normal, nos demais projetos SPL não houve diferença entre os valores obtidos paras as métricas SSC e SVC em todos os projetos. Com estes resultados podemos confirmar a hipótese H_{0a} (Os valores das métricas coletadas durante o processo de recuperação não são afetados pelo método de geração escolhido).

RQ2: A variabilidade e comunalidade detectada na PLA é afetada pelo método de geração de produtos? Não existiram diferenças entre as PLAs recuperadas pelos dois métodos de geração; todos os elementos foram identificados e classificados corretamente como pode ser confirmado pela Tabela 6.2. A métrica SSC calcula a quantidade de elementos comuns em relação aos variáveis enquanto que a métrica SVC calcula a quantidade de elementos variáveis em relação aos comuns, como os valores das métricas são iguais para os dois métodos de geração podemos concluir que a PLA recuperada pelo método T-Wise é idêntica a arquitetura recuperação pelo método de geração padrão. A exceção foi a SPL GPL, onde o método T-Wise identificou um elemento a mais que o método convencional, entretanto os demais elementos e relações foram classificados corretamente. O caso da SPL GPL é atípico pois o método de geração utilizando todas as configurações válidas deveria identificar todos os elementos e relações da PLA, o que não ocorreu, este problema foi relatado aos desenvolvedores da SPL.

Devido a esta situação atípica, a SPL GPL não foi considerada na resposta para essa pergunta de pesquisa. Dessa forma podemos concluir que não há diferença entre a PLA recuperada utilizando os dois métodos de geração de produtos o que confirma a hipótese H_{0b}.

6.5.2 Discussão

A Tabela 6.12 apresenta um comparativo entre a quantidade de produtos gerados por cada método de geração de produtos e o diferencial percentual entre cada um deles.

Pela tabela fica claro que ao utilizar o método T-Wise é possível reduzir drasticamente o número de produtos gerados para realização da recuperação. Isto traz alguns benefícios para o processo de recuperação, por exemplo:

- **Processo de recuperação realizado de forma mais rápida -** Ao reduzir o número de produtos que serão utilizados no processo de recuperação temos o aumento de desempenho no processo.
- **Aumento da escalabilidade da recuperação -** Ao utilizar um número reduzido de produtos no processo de recuperação é possível analisar projetos SPL com uma quantidade maior de módulos e relações uma vez que temos uma redução média de 60% na quantidade de produtos necessários para realizar a recuperação.

Em dois projetos SPL (PayCard e PokerSPL) verificamos que o método de geração normal consegue gerar a mesma PLA que o método T-Wise utilzando somente 25% do

Tabela 6.12 Comparativo da quantidade de produtos gerados por cada método de geração de produtos

	Normal	**T-Wise**	**Diferença %**
BankAccount	24	6	75
BankAccountv2	72	8	89
DesktopSearcher	462	9	98
Elevator	20	7	65
E-Mail	40	6	85
ExamDB	8	6	25
GPL	156	20	87
PayCard	6	5	17
PokerSPL	28	10	64
UnionFind	6	6	0

total de produtos, o que dá um número de produtos menor para análise. Entretanto o método T-Wise ainda se sai melhor no cálculo das métricas, pois os seus resultados são mais próximos do valor obtido utilizando todos os produtos do que os valores obtidos utilizando apenas 25% do total.

Além destas vantagens trazidas apenas ao reduzir o número de produtos, aliado com o resultado das questões de pesquisa proposta o uso do método T-Wise nos permite concluir que o seu uso é extremamente recomendado para o processo de recuperação da PLA de projetos SPL, pois ele não só reduz o número de produtos necessários para análise, como também torna o processo de recuperação mais rápido e escalável mantendo a mesma arquitetura e não impactando no valor das métricas coletadas.

Com este estudo verificamos que é mais vantajoso recuperar a PLA utilizando o método de geração T-Wise, pois em todos os casos tivemos redução na quantidade de produtos gerados, sendo que na pior hipótese tivemos o mesmo número de produtos que seriam gerados pelo método padrão. E com este número reduzido de produtos foi possível chegar a mesma PLA gerada pelo método convencional sem haver impacto significante no valor das métricas.

6.5.3 Ameaças a Validade do Estudo

Identificamos as seguintes ameaças a validade do estudo:

- **Ameaças a Validade Interna**
  - **Limitações da PLAR Tool -** As limitações da ferramenta discutidas no

Capítulo 3 impactaram nos resultados deste estudo exploratório. Atualmente, a PLAR Tool apenas processa os arquivos MDG criados pelo STAN4J, o qual funciona apenas com projetos Java. Estas limitações reduzem o escopo do projetos SPL que podem ser avaliados.

- **Ameaças a Validade Externa**
  - **Ausência de SPL utilizada na indústria -** Os projetos SPL utilizados neste estudo exploratório são projetos *open source* além de serem utilizados no contexto educacional, os resultados obtidos para estes projetos podem não ser os mesmo para outros tipos de projetos SPL.
  - **Não validação da PLA recuperada pelo desenvolvedores -** A PLA recuperada não foi certificada pelos criadores da SPL. Para minimizar esta ameaça selecionamos um dos projetos SPL e realizamos uma recuperação manual e comparamos com os resultados obtidos pela ferramenta para verificar a corretude da informação apresentada. A Tabela 6.13 apresenta a comparação entre a recuperação manual feita pelo autor e a feita pela PLAR Tool para a SPL DesktopSearcher com relação aos módulos identificados como comuns e variáveis.

Tabela 6.13 Comparação entre recuperação Manual x PLAR Tool

PayCard	Manual	PLAR Tool
Comuns	11	11
Variáveis	30	30

Como pode ser visto na tabela, não houve diferenças entre os elementos catalogados pelo autor e pela ferramenta. Apesar de minimizar os riscos, ainda há a necessidade de contactar os desenvolvedores para validação da PLA recuperada.

  - **Tamanho dos projetos SPL -** O tamanho dos projetos SPL são pequenos se comparado com os projetos vistos na indústria, os resultados obtidos neste estudo podem não ser válidos para projetos encontrados na indústria.

6.6 TRABALHOS RELACIONADOS

Perrouin *et. al.* conduziu um estudo para investigar a efetividade do método T-Wise na geração de produtos de uma SPL. Em seu estudo, a SPL aspectOPTIMA (PERROUIN et al., 2010b) foi utilizada para realizar os testes. Foi constatado que o uso do método T-Wise para geração de produtos fornece uma redução do número de produtos gerados pela linha sem perder sua representatividade em relação ao modelo de *features*. Em nosso estudo avaliamos o método T-Wise em relação ao aspecto de qualidade arquitetural da PLA, baseado nas métricas de reúso propostas por Zhang *et. al.* e De Oliveira *et. al.*

Henard *et. al.* apresenta em seu trabalho melhorias para o método T-Wise a fim de reduzir mais o número de produtos gerados pelo método, uma vez que considerou os resultados obtidos por Perrouin elevados (HENARD et al., 2014). Em seu estudo, 10 projetos SPL foram analisados com as otimizações implementadas, porém este estudo não procurou analisar a representatividade do método T-wise mas sim o desempenho e a quantidade de produtos gerados, os quais aumentaram e diminuíram respectivamente. Neste estudo utilizamos esta versão melhorada do T-Wise para avaliar a qualidade da PLA recuperada.

6.7 CONSIDERAÇÕES FINAIS

Neste estudo, utilizamos dois métodos de geração de produtos diferentes: o método padrão (gerar todos os produtos válidos) *versus* o método T-Wise, a fim de comparar a precisão e a abrangência de cada método em relação às PLAs recuperadas para dez projetos SPL. O grau de reúso de cada PLA foi avaliado.

Em relação ao grau de reúso, nove entre dez projetos SPL não apresentaram diferenças com base nas PLAs recuperadas, possuindo os mesmos resultados para as métricas SSC e SVC.

Em relação à qualidade da recuperação, não houve diferenças significativas entre os dois métodos usados.

Na linha de produtos GPL, a análise baseada nos produtos gerados pelo método T-Wise foi capaz de identificar mais elementos do que a análise realizada utilizando todas as configurações possíveis. Para as outras linhas de produtos não houve diferenças significativas.

Como trabalho futuro, planejamos ampliar o número de projetos SPL investigados por meio de um experimento controlado para verificar se os resultados obtidos neste estudo de caso pode ser replicados no contexto de projetos SPL industriais e/ou com projetos SPL de grande porte. O material de replicação para este estudo encontra-se disponível no endereço: http://v.ht/T-Wise.

Capítulo 7

CONCLUSÃO

Este trabalho apresentou a técnica PLAR e a ferramenta PLAR Tool, uma ferramenta para recuperação da PLA de projetos SPL. A técnica foi construída tomando como base abordagens similares, observando seus pontos fortes.

Além da recuperação da PLA, a ferramenta dá suporte à visualização da PLA, gerando diferentes tipos de saída para a PLA, e à avaliação do grau de reuso da PLA, implementando métricas de reuso referenciadas na literatura.

A Seção 7.1 apresenta as principais contribuições deste trabalho e a Seção 7.2 apresenta sugestões para trabalhos futuros.

7.1 PRINCIPAIS CONTRIBUIÇÕES

As contribuições desta dissertação são um resultado de uma revisão extensa sobre ferramentas e técnicas de recuperação de arquitetura no contexto de *single systems* e SPL. Infelizmente, a maior parte das ferramentas e técnicas publicadas não estão disponíveis ou não apresentam estudos de avaliação.

As principais contribuições alcançadas com estre trabalho são: (i) Desenvolvimento da técnica PLAR e da ferramenta PLAR Tool que permitem a recuperação da PLA de projetos SPL, (ii) dois estudos exploratórios onde pudemos verificar a eficácia da PLA recuperada pela ferramenta PLAR Tool e (iii) processo de avaliação da ferramenta.

7.1.1 PLAR Tool

A ferramenta PLAR Tool, desenvolvida no contexto deste trabalho, fornece suporte a recuperação automática da PLA servindo como prova de conceito da técnica PLAR, sendo capaz de identificar a comunalidade e variabilidade na PLA recuperada.

Adicionalmente, apóia a avaliação da PLA recuperada com diferentes métricas e a geração de alguns tipos de visualização da PLA recuperada.

A PLAR Tool também pode servir como base para o reuso de software, para apoiar a evolução da SPL, para verificação de conformidade entre produtos e a PLA, para

documentar linhas de produtos de software *open source*, que, em geral, não dispõem de arquitetura documentada.

7.1.2 Estudos de Caso

Dois estudos de caso de caráter exploratório foram conduzidos, um com o objetivo de avaliar o resultado da recuperação feita pela PLAR Tool e investigar a relação entre a eficácia das métricas em relação a quantidade de produtos inseridos para análise e outro estudo que comparou o resultado gerado pela ferramenta a partir de dois métodos de geração de produtos: o método convencional onde todas as configurações válidas são geradas e o método T-Wise onde um conjunto de configurações mais significativas é produzido.

7.1.3 Estudo de avaliação da Ferramenta

Um estudo foi realizado a fim de avaliar a ferramenta sob o ponto de vista de desenvolvedores SPL e potenciais usuários para que falhas no processo de recuperação implementando pela ferramenta pudessem ser verificadas e melhorias tanto na técnica quanto na ferramenta pudessem ser implementadas.

7.2 TRABALHOS FUTUROS

Devido as restrições de tempo à pesquisa de mestrado, certos pontos não foram investigados em profundidade e podem servir de direcionamento para trabalhos futuros de nosso grupo de pesquisa ou outros pesquisadores.

- **Melhorias na Técnica de Recuperação.** A técnica proposta é simples e depende de geração e seleção adequadas do menor número de produtos relevantes para funcionar corretamente. Pretende-se investigar outras técnicas que permitam reduzir tal dependência externa. Além disso, existem outras técnicas com propostas diferentes para recuperação de arquitetura de linhas de produtos de software, a análise detalhada destas técnicas pode nos ajudar a melhorar a técnica PLAR através da criação, modificação de algoritmos específicos para a recuperação da arquitetura de linha de produto de software.

- **Novas Métricas.** Esta dissertação considerou um conjunto pequeno de métricas encontradas na literatura, existe um conjunto amplo de métricas aplicadas ao contexto de SPL como de arquitetura que não foram consideradas. Assim como métricas específicas para o contexto SPL, existem outras métricas específicas para código-fonte, detecção de *code smells*, entre outros tipos que podem ser aplicadas na ferramenta para permitir novos tipos de análise da PLA recuperada e do código-fonte do produtos.

- **Replicação dos Estudos.** Os estudos exploratórios realizados consideraram um conjunto de projetos SPL *open source*, com nenhum exemplo da indústria, seria interessante a replicação destes estudos no contexto de projetos SPL industriais,

bem como a replicação dos estudos realizados neste trabalho comparando com outras ferramentas e com validação da PLA pelos desenvolvedores do projeto a fim de minimizar as ameaças a validade detectadas durante o planejamento do estudo.

- **Suporte a outras ferramentas de extração de dependências.** Atualmente a PLAR Tool oferece suporte a extração realizada pela ferramenta STAN4J, o que limita a recuperação da PLA para projetos SPL desenvolvidos somente em Java. Estamos estudando a ferramenta de extração Analizo[1], que permite a extração de dependências de projetos escritos em C/C++, o que ampliaria o escopo da recuperação realizado pela PLAR Tool. Existem outras ferramentas com a mesma função que podem fazer parte do conjunto de arquivos suportados pela PLAR Tool, como por exemplo o Doxygen[2].

- **Novos tipos de visualização de arquitetura.** As saídas da PLA recuperada com a PLAR Tool permitem a geração de visões arquiteturais para o tipo de visão de módulo, que contemplam apenas classes e algumas relações. Futuramente, planejamos estender a ferramenta para explorar diferentes estilos arquiteturais e diferentes tipos de visão, por exemplo, componente-e-conector.

[1]http://www.analizo.org/
[2]http://www.stack.nl/ dimitri/doxygen/

REFERÊNCIAS BIBLIOGRÁFICAS

ALBAUM, G. The likert scale revisited: an alternate version. *Journal of the Market Research Society*, World Advertising Research Center Ltd., v. 39, n. 2, p. 331–332, 1997.

APEL, S. et al. *Feature-Oriented Software Product Lines*. [S.l.]: Springer, 2013.

BASILI, V. R.; CALDIERA, G.; ROMBACH, H. D. The goal question metric approach. In: *Encyclopedia of Software Engineering*. [S.l.]: Wiley, 1994.

BERGER, T. et al. A survey of variability modeling in industrial practice. In: *Proceedings of the Seventh International Workshop on Variability Modelling of Software-intensive Systems*. New York, NY, USA: ACM, 2013. (VaMoS '13), p. 7:1–7:8. ISBN 978-1-4503-1541-8. Disponível em: ⟨http://doi.acm.org/10.1145/2430502.2430513⟩.

CLEMENTS, P. *Software Architecture in Practice*. [S.l.: s.n.], 2002.

CLEMENTS, P. et al. *Documenting Software Architectures: Views and Beyond*. [S.l.]: Pearson Education, 2010. ISBN 0321552687.

DUCASSE, S.; POLLET, D. Software architecture reconstruction: A process-oriented taxonomy. *IEEE Transactions on Software Engineering*, v. 35, n. 4, p. 573–591, July 2009. ISSN 0098-5589.

EIXELSBERGER, W. Software architecture recovery of product lines. 2000.

EIXELSBERGER, W. et al. Software architecture recovery of a program family. In: *Proceedings of the 20th International Conference on Software Engineering*. Washington, DC, USA: IEEE Computer Society, 1998. (ICSE '98), p. 508–511. ISBN 0-8186-8368-6. Disponível em: ⟨http://dl.acm.org/citation.cfm?id=302163.302225⟩.

EIXELSBERGER, W. et al. Software architecture recovery of a program family. In: IEEE. *Software Engineering, 1998. Proceedings of the 1998 International Conference on*. [S.l.], 1998. p. 508–511.

EIXELSBERGER, W. et al. A framework for software architecture recovery. In: CITESEER. *International Workshop on Development and Evolution of Software Architectures for Product Families in Las Navas del Marques, Avila, Spain*. [S.l.], 1996.

FOWLER, M. *Refactoring: improving the design of existing code*. [S.l.]: Pearson Education India, 2009.

GANTER, B.; GODIN, R. *Formal concept analysis*. [S.l.]: Springer Berlin/Heidelberg., 2005.

HENARD, C. et al. Bypassing the combinatorial explosion: Using similarity to generate and prioritize t-wise test configurations for software product lines. *IEEE Transactions on Software Engineering*, IEEE, v. 40, n. 7, p. 650–670, 2014.

ISO/IEC/IEEE International Standard for Architecture Descriptions. [S.l.], 2011.

JANSEN, A.; BOSCH, J. Software architecture as a set of architectural design decisions. In: IEEE. *5th Working IEEE/IFIP Conference on Software Architecture (WICSA'05)*. [S.l.], 2005. p. 109–120.

JUNIOR, E. A. de O.; GIMENES, I.; MALDONADO, J. A metric suite to support software product line architecture evaluation. In: *XXXIV Conferencia Latinoamericana de Informatica*. [S.l.: s.n.], 2008. p. 489–498.

KANG, K. C. et al. *Feature-oriented domain analysis (FODA) feasibility study*. [S.l.], 1990.

KRUEGER, C. New methods behind a new generation of software product line successes. *KC Kang, V. Sugumaran, and S. Park, "Applied Software Product Line Engineering", Auerbach Publications*, p. 39–60, 2010.

KRUEGER, C. W. Easing the transition to software mass customization. In: *Revised Papers from the 4th International Workshop on Software Product-Family Engineering*. London, UK, UK: Springer-Verlag, 2002. (PFE '01), p. 282–293. ISBN 3-540-43659-6. Disponível em: ⟨http://dl.acm.org/citation.cfm?id=648114.748909⟩.

KRUEGER, C. W. Biglever software gears and the 3-tiered spl methodology. In: ACM. *Companion to the 22nd ACM SIGPLAN conference on Object-oriented programming systems and applications companion*. [S.l.], 2007. p. 844–845.

LIMA-NETO, C. R. et al. Initial evidence for understanding the relationship between product line architecture and software architecture recovery. In: IEEE. *Components, Architectures and Reuse Software (SBCARS), 2015 IX Brazilian Symposium on*. [S.l.], 2015. p. 40–49.

LINDEN, F. J. v. d.; SCHMID, K.; ROMMES, E. *Software Product Lines in Action: The Best Industrial Practice in Product Line Engineering*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2007. ISBN 3540714367.

LINSBAUER, L.; LOPEZ-HERREJON, R. E.; EGYED, A. Variability extraction and modeling for product variants. *Software & Systems Modeling*, Springer, p. 1–21, 2016.

LOSAVIO, F. et al. Graph modelling of a refactoring process for product line architecture design. In: *2013 XXXIX Latin American Computing Conference (CLEI), Caracas (Naiguata), Venezuela, October 7-11, 2013*. [s.n.], 2013. p. 1–12. Disponível em: ⟨http://dx.doi.org/10.1109/CLEI.2013.6670632⟩.

NAKAGAWA, E. Y.; ANTONINO, P. O.; BECKER, M. Reference architecture and product line architecture: a subtle but critical difference. In: SPRINGER. *European Conference on Software Architecture*. [S.l.], 2011. p. 207–211.

NAKAGAWA, E. Y.; BECKER, M.; MALDONADO, J. C. Towards a process to design product line architectures based on reference architectures. In: *Proceedings of the 17th International Software Product Line Conference*. New York, NY, USA: ACM, 2013. (SPLC '13), p. 157–161. ISBN 978-1-4503-1968-3. Disponível em: ⟨http://doi.acm.org/10.1145/2491627.2491651⟩.

PERROUIN, G. et al. Automated and scalable t-wise test case generation strategies for software product lines. In: *2010 Third International Conference on Software Testing, Verification and Validation*. [S.l.: s.n.], 2010. p. 459–468. ISSN 2159-4848.

PERROUIN, G. et al. Automated and scalable t-wise test case generation strategies for software product lines. In: IEEE. *Software Testing, Verification and Validation (ICST), 2010 Third International Conference on*. [S.l.], 2010. p. 459–468.

POHL, K.; BöCKLE, G.; LINDEN, F. J. v. d. *Software Product Line Engineering: Foundations, Principles and Techniques*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2005. ISBN 3540243720.

RUBIN, J.; CHECHIK, M. Locating distinguishing features using diff sets. In: ACM. *Proceedings of the 27th IEEE/ACM International Conference on Automated Software Engineering*. [S.l.], 2012. p. 242–245.

SCHRIJVERS, T.; FRÜHWIRTH, T. Optimal union-find in constraint handling rules. *Theory and Practice of Logic Programming*, Cambridge Univ Press, v. 6, n. 1-2, p. 213–224, 2006.

SHATNAWI, A.; SERIAI, A.; SAHRAOUI, H. Recovering architectural variability of a family of product variants. In: *Software Reuse for Dynamic Systems in the Cloud and Beyond*. [S.l.]: Springer, 2014. p. 17–33.

SHATNAWI, A.; SERIAI, A.-D.; SAHRAOUI, H. Recovering software product line architecture of a family of object-oriented product variants. *Journal of Systems and Software*, Elsevier, 2016.

SULLIVAN, K. J. et al. The Structure and Value of Modularity in Software Design. In: *8th European Software Engineering Conference*. [S.l.]: ACM, 2001. p. 99–108. ISSN 0163-5948.

TAYLOR, R. N.; MEDVIDOVIC, N.; DASHOFY, E. M. *Software Architecture: Foundations, Theory, and Practice*. [S.l.]: Wiley Publishing, 2009. ISBN 0470167742, 9780470167748.

TILLEY, S. R.; SMITH, D. B.; PAUL, S. Towards a framework for program understanding. In: IEEE. *wpc*. [S.l.], 1996. p. 19.

TORKAMANI, M. A. Extractability effectiveness on software product line. *International Journal of Electrical and Computer Engineering*, IAES Institute of Advanced Engineering and Science, v. 4, n. 1, p. 127, 2014.

WOHLIN, C. et al. *Experimentation in software engineering*. [S.l.]: Springer Science & Business Media, 2012.

WU, Y. et al. Recovering object-oriented framework for software product line reengineering. In: *Proceedings of the 12th International Conference on Top Productivity Through Software Reuse*. Berlin, Heidelberg: Springer-Verlag, 2011. (ICSR'11), p. 119–134. ISBN 978-3-642-21346-5. Disponível em: ⟨http://dl.acm.org/citation.cfm?id=2022115.2022130⟩.

ZHANG, B.; BECKER, M. Code-based variability model extraction for software product line improvement. In: ACM. *Proceedings of the 16th International Software Product Line Conference-Volume 2*. [S.l.], 2012. p. 91–98.

ZHANG, B.; BECKER, M. Recovar: A solution framework towards reverse engineering variability. In: IEEE. *Product Line Approaches in Software Engineering (PLEASE), 2013 4th International Workshop on*. [S.l.], 2013. p. 45–48.

ZHANG, T. et al. Some metrics for accessing quality of product line architecture. In: IEEE. *International Conference on Computer Science and Software Engineering*. [S.l.], 2008. v. 2, p. 500–503.

Apêndice

A

QUESTIONÁRIO

Informações Gerais

Idade: _____

Sexo: [ ] masculino [ ] feminino

Para as questões a seguir é permitido selecionar mais de uma opção.

Experiência previa com desenvolvimento de software:
[ ] Nunca desenvolveu software antes
[ ] Já desenvolvi projetos anteriormente, sozinho
[ ] Já desenvolvi projetos anteriormente, em equipe
[ ] Já desenvolvi projetos anteriormente, em empresas

Qual a sua experiência com programação (em meses ou anos, especifique):

[ ] Já trabalhei com desenvolvimento na indústria
Quanto tempo: _____

[ ] Já trabalhei com desenvolvimento na academia
Quanto tempo: _____

Quais são as linguagens de programação que você já utilizou/trabalhou?
_____
_____
_____

Já teve experiência prévia com desenvolvimento de projetos de Linha de Produto de Software (LPS)?
[ ] Nenhuma
[ ] Já estudei, conheci através de livros ou em sala de aula
[ ] Já desenvolvi projetos de LPS na academia
Quanto tempo: _____

[ ] Já desenvolvi projetos de LPS na indústria
Quanto tempo: _____

Já teve experiência prévia com arquitetura de software?
[ ] Nenhuma
[ ] Já estudei, conheci através de livros ou em sala de aula
[ ] Já desenvolvi projetos utilizando arquitetura de software na academia
Quanto tempo: _____

[ ] Já desenvolvi projetos utilizando arquitetura de software na indústria
Quanto tempo: _____

Informações Específicas

1 - Existe algum elemento da PLA que está classificado incorretamente?

2 - Dadas as configurações dos produtos escolhidas para extração, existe alguma configuração que desejariam que fosse adicionada?

Feature	T/F	Feature	T/F

3 – Para os itens abaixo, marque com um X o que melhor se adequa a sua resposta:

Critério	Discordo Totalmente	Discordo	Neutro	Concordo	Concordo Totalmente
a. As configurações dos produtos representam bem a linha?	[ ]	[ ]	[ ]	[ ]	[ ]
b. A representação da PLA é satisfatória?	[ ]	[ ]	[ ]	[ ]	[ ]
c. As relações presentes na PLA condizem com as relações implementadas?	[ ]	[ ]	[ ]	[ ]	[ ]
d. As visões fornecidas pela PLA facilitam sua manutenção?	[ ]	[ ]	[ ]	[ ]	[ ]
e. Os elementos mapeados no arquivo *report* estão de acordo com a configuração dos produtos?	[ ]	[ ]	[ ]	[ ]	[ ]

Justificativas:

4 – Com relação ao Diagrama de Classe

Critério	Discordo Totalmente	Discordo	Neutro	Concordo	Concordo Totalmente
a. A representação utilizando pacotes é melhor?	[ ]	[ ]	[ ]	[ ]	[ ]
b. As cores utilizadas para representação dos pacotes contribuíram para melhor entendimento do diagrama?	[ ]	[ ]	[ ]	[ ]	[ ]
c. A representação de detalhes como classes abstratas e interfaces são importante ?	[ ]	[ ]	[ ]	[ ]	[ ]
d. O nível de detalhes do diagrama de classe foi suficiente para entender a visão geral da arquitetura?	[ ]	[ ]	[ ]	[ ]	[ ]

5 – Questões de feedback

Critério	Discordo Totalmente	Discordo	Neutro	Concordo	Concordo Totalmente
a. Você precisou de ajuda para entender os gráficos gerados pela ferramenta?	[ ]	[ ]	[ ]	[ ]	[ ]
b. Você utilizaria esta ferramenta no seu cotidiano profissional?	[ ]	[ ]	[ ]	[ ]	[ ]
c. Você sugeriria o uso desta ferramenta para os seus amigos?	[ ]	[ ]	[ ]	[ ]	[ ]

6 – Quais outras representações ou diagramas você considera como indispensável para representação de uma boa arquitetura?

_____
_____
_____

7 – Quais foram as suas principais dificuldades?

_____
_____
_____

8 – Informe, caso ache relevante, pontos de melhoria para representação da PLA.

9 – Por favor utilize o espaço abaixo pra sugerir informações que você acredita útil para melhoria deste trabalho

_____
_____
_____