

UNIVERSIDADE ESTADUAL DE MARINGÁ
PROGRAMA INSTITUCIONAL DE BOLSAS DE INICIAÇÃO CIENTÍFICA – PIBIC – CNPQ

DEPARTAMENTO DE INFORMÁTICA (DIN)

ORIENTADOR: Prof. Dr. Edson Alves de Oliveira Junior

BOLSISTA: Leandro Flores da Silva

***SMartyAnalyzer*, um ambiente para aplicação de métricas SMM em Avaliação de
Arquiteturas de Linha de Produto de Software.**

Maringá, 31 de julho de 2017.

UNIVERSIDADE ESTADUAL DE MARINGÁ
PROGRAMA INSTITUCIONAL DE BOLSAS DE INICIAÇÃO CIENTÍFICA – PIBIC - CNPQ

DEPARTAMENTO DE INFORMÁTICA (DIN)

ORIENTADOR: Prof. Dr. Edson Alves de Oliveira Junior

BOLSISTA: Leandro Flores da Silva

***SMartyAnalyzer*, um ambiente para aplicação de métricas SMM em Avaliação de
Arquiteturas de Linha de Produto de Software.**

Relatório contendo os resultados finais do projeto de iniciação
científica vinculado ao PIBIC/CNPq-UEM.

Maringá, 31 de julho de 2017

RESUMO

Este projeto implementou o módulo do ambiente SMartyAnalyzer, responsável por permitir a definição e aplicação de métricas SMM em arquiteturas de Linha de Produto de Software, no contexto de avaliação de Linha de Produto de Software (LPS). A avaliação acontece pela entrada de dados via fontes externas como, por exemplo, arquivos XMI contendo os modelos UML de uma LPS. Em seguida, o usuário define métricas que permitem a avaliação da LPS e comparação das medições. As tecnologias utilizadas no desenvolvimento do módulo foram investigadas ao longo do projeto.

Palavras-chave: SMartyAnalyzer, SMarty, Métrica SMM.

INTRODUÇÃO

Linha de Produto de Software (LPS) é uma abordagem de desenvolvimento, que aplica o reuso de forma sistemática em um domínio de atuação (Linden; Schmid; Rommes, *et al.*, 2007), sendo que sua adoção vem se consolidando nos últimos anos, tanto na indústria como na academia (Etxeberria e Sagardui, 2008; Gomaa, 2005). Uma linha de produto de software (LPS) representa um conjunto de sistemas de software que compartilham características comuns e gerenciáveis, que satisfazem as necessidades de um segmento particular do mercado ou de uma missão (Clements e Northrop, 2001; Northrop, 2002). Esse conjunto de sistemas de software é também chamado de família de produtos.

Linha de Produto de Software (LPS) pode ser definida como um conjunto de sistemas de software com características (*features*) comuns para um determinado segmento, sendo construído utilizando como base as semelhanças entre os sistemas. Esse conjunto de produtos compartilha uma mesma arquitetura geral que está incluído no núcleo de artefatos (Capilla; Bosch; Kang, *et al.*, 2013).

O núcleo de artefatos constitui a referência da Linha de Produto, incluindo a Arquitetura da Linha de Produto, que é a base de qualquer produto instanciado, além de componentes reusáveis, modelos de domínios, requisitos, planos de teste e modelos de características e de variabilidades (Oliveira Junior *et al.*, 2013). O objetivo principal da abordagem de LPS é promover a geração de produtos específicos com base na reutilização dos artefatos da arquitetura da LPS. Essa arquitetura

descreve os elementos que compõem o sistema e as interações entre eles (Linden; Schmid; Rommes, *et al.*, 2007).

Nos últimos anos, a abordagem de Linha de Produto de Software vem definindo cada vez mais o Gerenciamento de Variabilidades (GV) como atividade fundamental para a instanciación de produtos em larga escala (Capilla; Bosch; Kang, *et al.*, 2013). Para a representação de uma variabilidade, existem dois elementos importantes para a representação: ponto de variação e variante. O ponto de variação descreve o ponto em que existe uma diferença nos produtos a serem instanciados. O ponto de variação identifica os locais nos quais as variações são combinadas para ocorrer. Os pontos de variação definem as diferenças da LPS. A variante define os elementos para a resolução de um ponto de variação, sendo uma alternativa de resolução para uma determinada variabilidade, correspondendo à uma possível solução (Linden; Schmid; Rommes, *et al.*, 2007).

A Avaliação de LPS, assim como o gerenciamento dos artefatos, contempla um conjunto de atividades, possivelmente realizadas em diferentes momentos do ciclo de vida da LPS (Etxeberria e Sagardui, *et al.*, 2008). Entenda-se como ciclo de vida, o conjunto de atividades realizadas, desde o plano de adoção, até as atividades de manutenção e evolução da LPS.

As atividades de avaliação possibilitam tanto a verificação de artefatos isolados, quanto de produtos instanciados. Mais especificamente sobre a avaliação de produtos instanciados, diferentes atributos de qualidade podem ser considerados na avaliação, tais como Complexidade (Oliveira Junior *et al.*, 2012) e Extensibilidade (Oliveira Junior e Gimenes, 2014). A avaliação considerando os atributos de qualidade pode ser realizada por meio de métricas.

As métricas auxiliam na avaliação dos artefatos, possibilitando análises quantitativas e qualitativas. Como exemplo de análises quantitativo e qualitativo respectivamente tem-se a contagem do número de variabilidades existentes em uma arquitetura de LPS e a complexidade de um dado produto instanciado dessa arquitetura (Oliveira Junior *et al.*, 2013).

O *Object Management Group* (OMG) apresentou a especificação do *Structured Metrics Metamodel* (SMM) (SMM, 2015), para especificação de métricas. O SMM permite a definição das métricas e a representação dos resultados das medições, buscando estabelecer um padrão para representação (Honda, 2014).

OBJETIVOS

Este projeto teve como objetivo o desenvolvimento do ambiente SMartyAnalyzer, um ambiente automatizado para definição de métricas SMM e para a avaliação de LPS baseadas em UML. O projeto também apresentou outros objetivos, tais como: investigação de técnicas para avaliação de LPS, além da capacitação do aluno responsável pelo projeto, bem como possíveis transferências de tecnologia para a indústria.

DESENVOLVIMENTO

O ambiente SMartyAnalyzer considerou todas as definições relacionadas aos diagramas UML e LPS. Mesmo sendo um projeto de iniciação científica, houve o contato com outros alunos da graduação e pós-graduação, principalmente no que diz respeito às experiências sobre LPS e a pesquisa científica.

O período da pesquisa foi de 01/08/2016 à 31/07/2017. Abaixo são detalhadas as atividades que correspondem ao Plano de Trabalho executado ao longo do período.

- (1) Estudo de métricas e a importância para a avaliação de LPS, explorando o padrão SMM;
- (2) Estudo da tecnologia necessária para automatização da leitura dos modelos exportados em XMI, principalmente no que se refere às informações exportadas e execução das métricas;
- (3) Implementação do ambiente responsável por definir métricas SMM e automatizar a aplicação de métricas sobre os modelos UML;
- (4) Validação do ambiente e execução de testes;
- (5) Avaliação das métricas SMM sobre modelos UML com base nas LPS's.

A tabela abaixo apresenta todas as atividades executadas ao longo do projeto e os respectivos períodos de tempo atribuídos a cada etapa.

Tabela 1. Cronograma de atividades durante o projeto

DESCRIÇÃO DAS ATIVIDADES		1º	2	3º	4º	5º	6º	7º	8º	9º	10º	11º
<i>Estudo inicial de métricas, SMM e Avaliação de LPS.</i>		X	X	X								
<i>Estudo da tecnologia necessária para o desenvolvimento do ambiente.</i>				X	X							
<i>Implementação do ambiente que realizará a definição e aplicação de métricas.</i>				X	X	X	X	X	X			
<i>Escrever o relatório semestral</i>							X					
<i>Validação do ambiente SMartyAnalyzer.</i>											X	X
<i>Avaliação das métricas SMM com base nos modelos UML de LPS..</i>											X	X
<i>Escrever o relatório final</i>												X

O desenvolvimento do ambiente pode ser dividido em três etapas bem distintas. A primeira consistiu em selecionar entre as ferramentas de modelagem UML a mais adequada para exportação de informações, sendo escolhido o *Astah Professional*, levando em consideração tanto as informações exportadas, quanto a estrutura dos dados no arquivo. O arquivo é exportado com a extensão XMI, que é um arquivo de notação baseado no XML. A partir disso, usou-se o *XPath*, que é uma linguagem de consulta definida pela W3C para selecionar nós em arquivos XML. Portanto, a construção da estrutura dos modelos passa pela leitura dos arquivos XMI via *XPath*.

A segunda etapa consistiu em definir uma maneira de especificar a operação das métricas, em resumo, definir o que ela deve buscar, sobre qual critério e como buscar. Para tanto, foi definido a seguinte estrutura: **palavra-chave**, que especifica o que deve ser buscado, levando em consideração qual diagrama-alvo e a **cláusula**, que especifica quais devem ser filtradas, também considerando qual o diagrama-alvo e o que se busca. Assim, o processamento encontra a palavra-chave, retorna as entidades encontradas e a seguir encontra a cláusula e filtra as entidades, permitindo inclusive combinações utilizando operadores matemáticos.

A terceira e última etapa consistiu em definir a interface gráfica do sistema e seus respectivos controladores, além de definir a arquitetura do sistema, incluindo as classes de persistência. O sistema foi construído seguindo os conceitos da arquitetura MVC (*Model-View-Controller*). Para a interface gráfica foi usado a biblioteca *swing* e para a persistência de dados utilizado o *Hibernate*.

RESULTADOS

O projeto apresentou como resultado as seguintes funcionalidades:

- integração da especificação SMM (*Structured Metrics Meta-model*), responsável por definir uma estrutura para representação das métricas e medições em modelos UML, permitindo que métricas sejam aplicadas para a arquitetura da LPS modeladas em UML;
- integração do módulo de aplicação de métricas sobre modelos UML, permitindo a aplicação de métricas personalizadas para a arquitetura de LPS e apresentação dos resultados das medições de forma amigável por meio de uma interface gráfica;
- integração do módulo de exportação e importação de métricas SMM para integração entre ferramentas;
- integração do módulo de exportação de código-fonte na linguagem de programação Java para um modelo UML;

A figura 1 apresenta a visão geral do ambiente SMartyAnalyzer para o processamento de métricas.

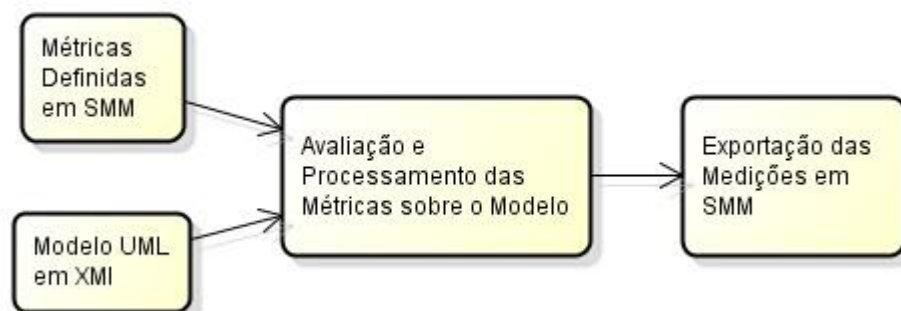


Figura 1. Visão Geral do ambiente.

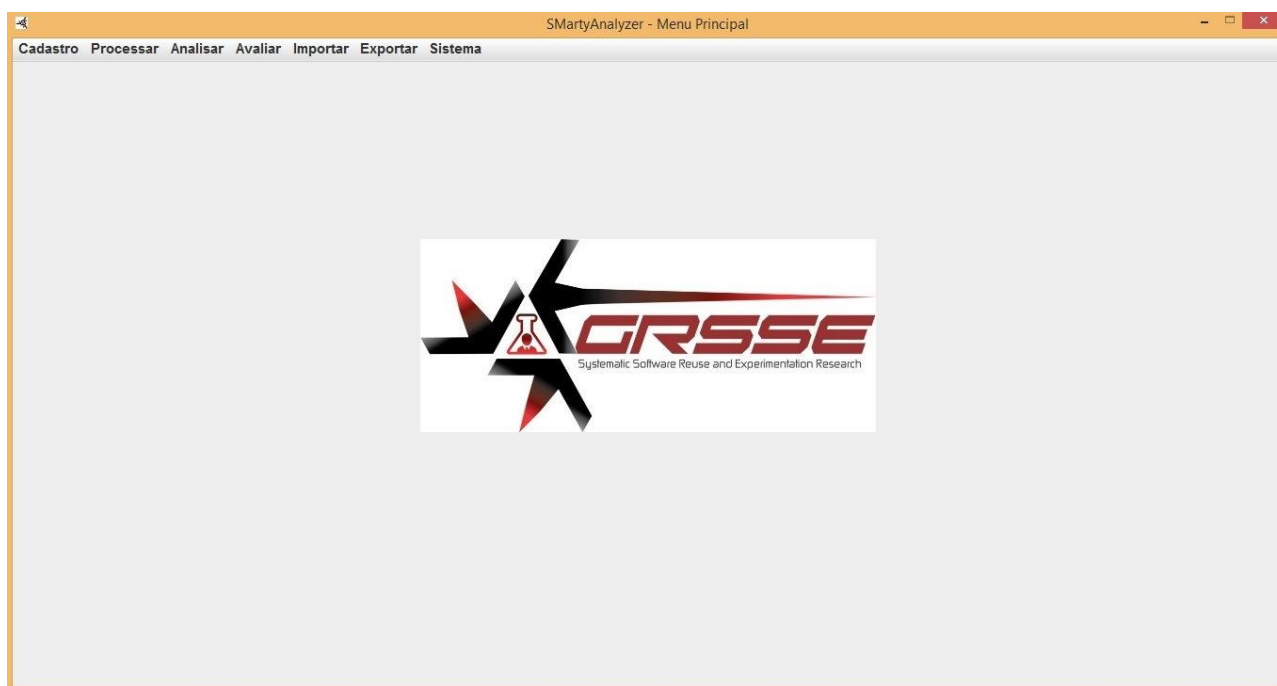



Figura 2. Interface inicial do ambiente.

SMartyAnalyzer - Consulta de Métricas

SMartyAnalyzer - Cadastro de Métrica

 Cadastro de Métrica

Os campos (*) são Obrigatorios

Nome*: NUMERO DE METODOS CONSTRUTORES Rotulo*: NumMetConst

Descricao*: Metrica responsavel por retornar o Numero de Metodos Construtores.

Diagrama Alvo*: DIAGRAMA DE CLASSE Tipo*: COUNTING

Medida*: METODO Operacao*: METODO(construtor)



 Inserir  Limpar  Voltar

Figura 3. Interface para Cadastro de Métricas.

SMartyAnalyzer - Processar Métrica sobre Diagrama de Classe

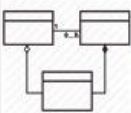
 **Processar Métrica**

Diagrama de Classe*: Modelos UML\Diagramas de Classe\Metodos\Export_02.xmi 🔍

Pacotes: 3

Classes: 3

Interfaces: 3

Métrica*: 3 - NUMERO DE METODOS PUBLICOS 🔍 ↺

Valor da Métrica: 12.0

Nome da Medicao:



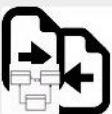
 Gravar  Cancelar

Figura 4. Interface para Processamento de Métricas sobre o Modelo.

SMartyAnalyzer - Avaliar Diagramas de Classe

 **Avaliar Diagramas de Classe**

delos UML\Diagramas de Classe\Realizacao\Export_01.xmi 🔍

os UML\Diagramas de Classe\Generalizacao\Export_01.xmi 🔍

Pacotes: 1

Classes: 2

Interfaces: 3

Metodos: 0

Atributos: 0

Pacotes: 1

Classes: 2

Interfaces: 0

Metodos: 0

Atributos: 0

Métrica*: 3 - NUMERO DE METODOS PUBLICOS 🔍

Resultado: 0.0 Resultado: 0.0



 Atualizar  Voltar

Figura 5. Interface para comparar Modelos usando uma Métrica.

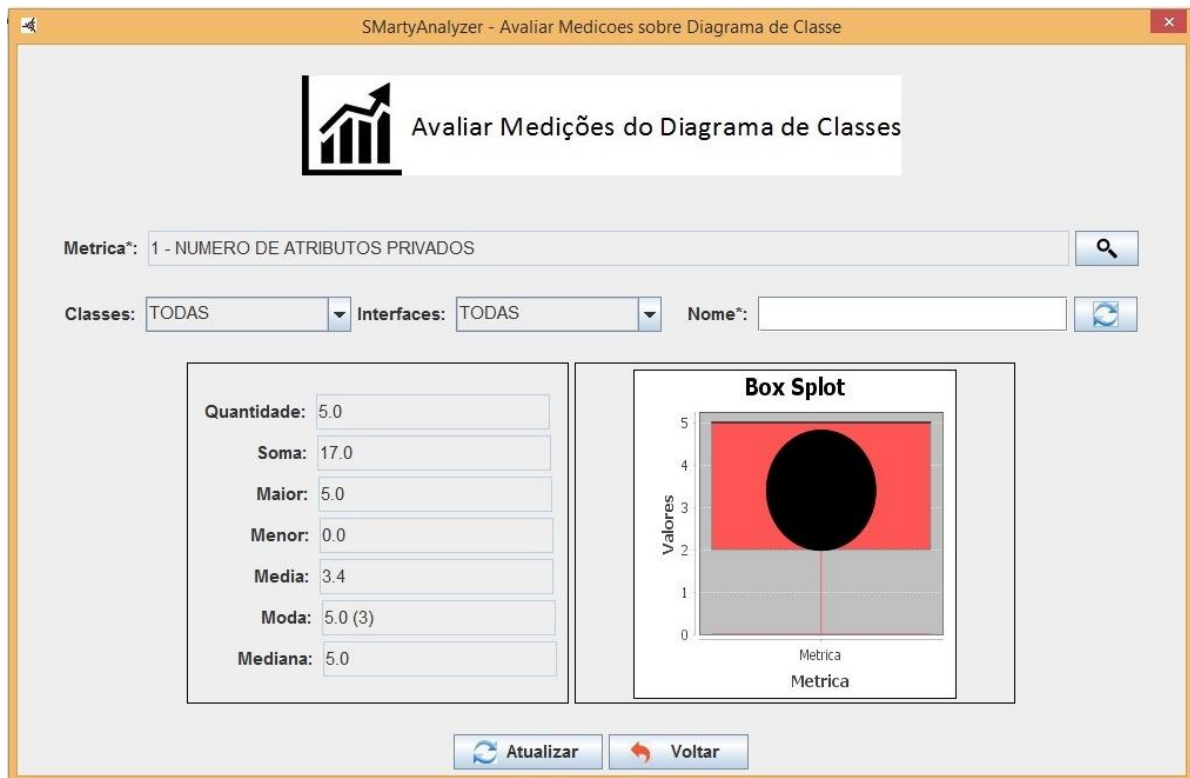


Figura 6. Interface para Avaliar as Medições.

Na figura 2 é apresentado o menu inicial do ambiente, expondo as opções para o usuário navegar pelo sistema.

Na figura 3 é apresentada a interface para cadastro de métricas, contendo os atributos definidos pelo SMM: nome, rótulo, descrição, diagrama-alvo (diagrama de classe e diagrama de caso de uso), tipo, unidade de medida e a operação. O objetivo é quando criada uma nova métrica, armazená-la e opcionalmente reutilizá-la ou exportá-la para outras ferramentas.

Na figura 4 é apresentada a interface para o processamento de uma métrica sobre o modelo UML, sendo possível tanto para o diagrama de classe, quanto para o diagrama de caso de uso. Em síntese tem como entrada o arquivo XMI com o modelo UML, são exibidas as principais informações sobre o modelo UML, sendo que é requerida a métrica a ser aplicada, e a seguir é processado e exibido os resultados da medição, sendo possível associar um nome a medição para posterior comparação de medições.

Na figura 5 é apresentada a interface para comparação de dois modelos UML, selecionando uma métrica, sendo aplicado para diagrama de caso de uso e de classe. Tem como entrada dois arquivos XMI com os modelos UML a serem comparados e a métrica usada como comparativo e então realiza a aplicação no dois modelos, exibindo o resultado das medições.

Na figura 6 é apresentada a interface para a comparação entre as medições, também é possível a análise para o diagrama de classe e de caso de uso. Para a comparação deve-se selecionar uma métrica em específico ou o nome gravado durante o processamento da métrica. São apresentados as informações gerais da medição, o número de medições, o maior valor, o menor valor, a média, a moda, a mediana e o *boxplot* expondo a distribuição dos valores das medições filtradas.

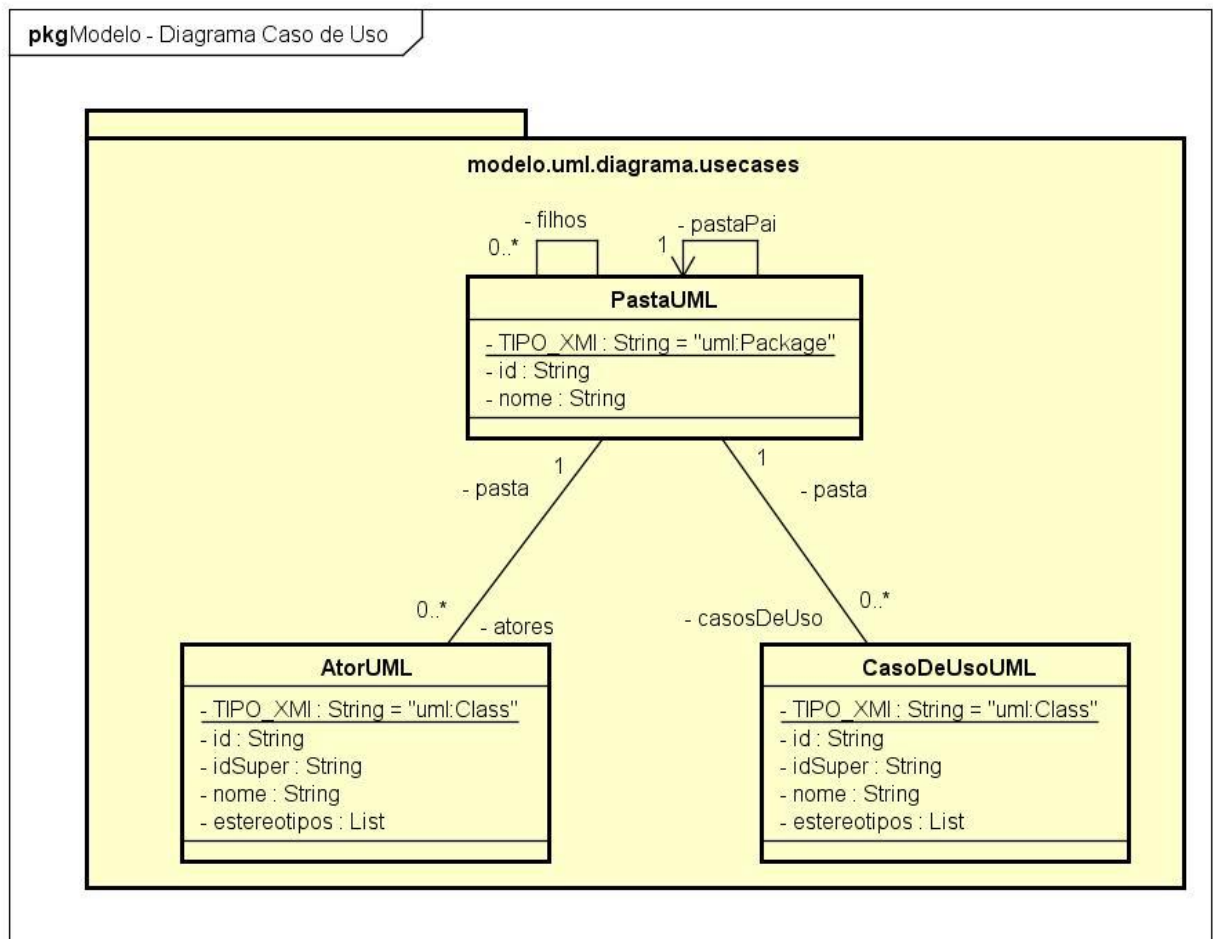


Figura 7. Estrutura usada para o Diagrama de Caso de Uso.

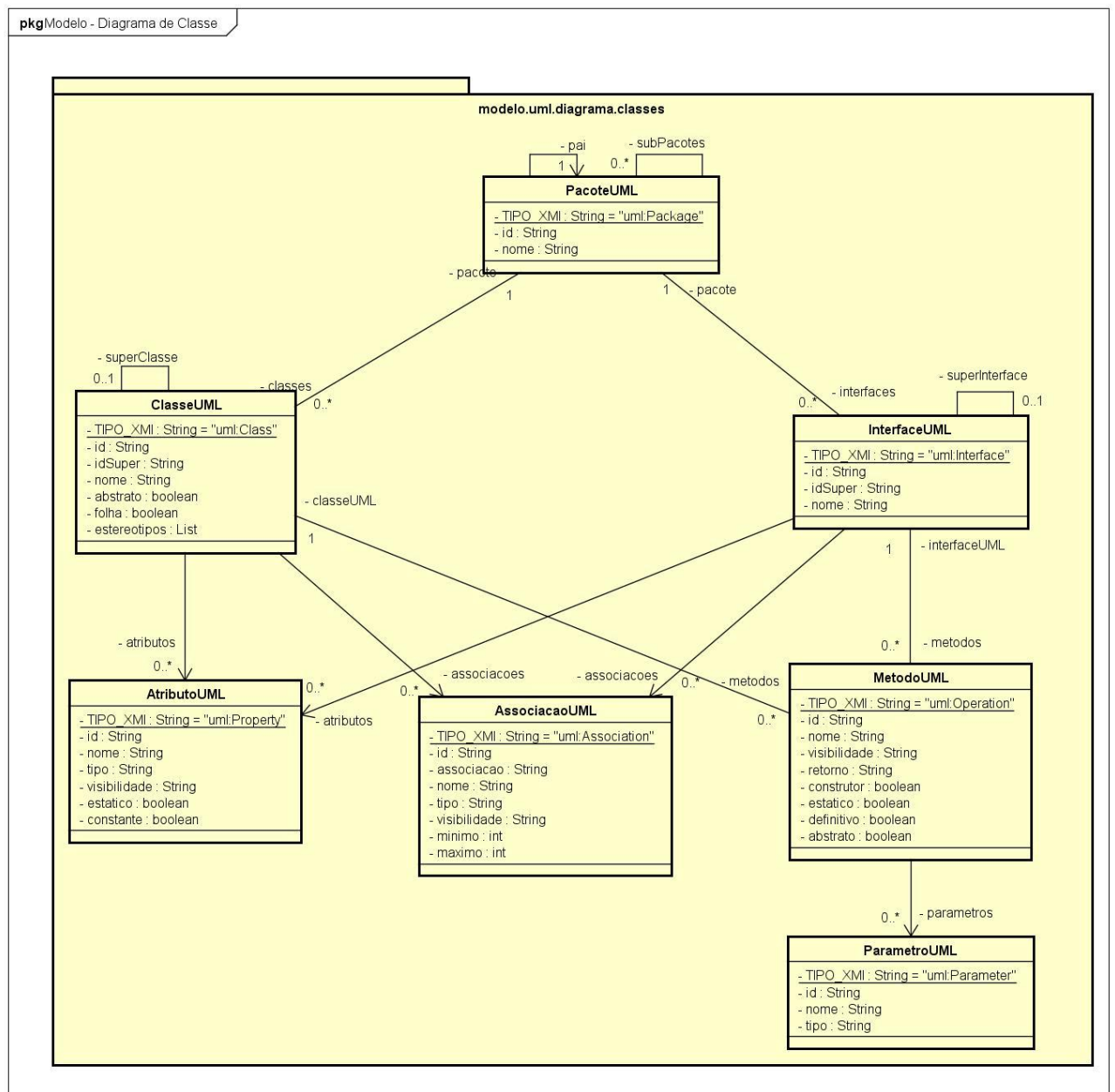


Figura 8. Estrutura usada para o Diagrama de Classe.

Nas figuras 7 e 8 são apresentadas as estruturas definidas para representar o Diagrama de Caso de Uso e o Diagrama de Classe. Durante a leitura do arquivo XMI, conforme se recupera os nós, vai se construindo sua respectiva estrutura. Para a aplicação de métricas e recuperação de informações nos processos posteriores de avaliação, usa-se essas estruturas para navegar e recuperar todas as informações pertinentes ao modelo.

A construção de ambas estruturas levou em conta principalmente a maneira como o arquivo XMI é exportado pelo *Astah Professional*, além de considerar as informações relevantes para a identificação do modelo e para extração de informações via aplicação de métricas.

DISCUSSÃO

Este projeto desenvolveu o ambiente SMartyAnalyzer, responsável por auxiliar a Avaliação de LPS. Esse ambiente permite a definição de métricas, aplicação das métricas sobre modelos UML e a comparação das medições, representando um ambiente não encontrado na literatura até o presente momento.

A aplicação de métricas definidas pelo usuário para os modelos UML permite que a avaliação seja realizada de maneira mais simplificada, com o ambiente permitindo inclusive a comparação de arquiteturas usando como referência uma determinada métrica, além de permitir a comparação de medições.

DIFICULDADES

A dificuldade inicial se deu na investigação e análise das ferramentas utilizadas para modelagem UML, sendo feito um levantamento entre as ferramentas abertas e comparando a maneira como permite a modelagem com estereótipos para LPS e se é exportada as informações do modelo e de uma maneira estruturada para posterior leitura.

Dentre as ferramentas analisadas, o *Astah Professional* permite a modelagem dos diagramas UML fazendo uso de estereótipos, permitindo duas opções de exportação: a primeira incluindo a maior parte dos dados necessários, porém não seguindo um padrão de sequência e incluindo muitas informações irrelevantes para a construção do modelo UML. Já a segunda maneira, exporta seguindo uma estrutura muito bem definida, seguindo uma ordem lógica para extração e incluindo apenas informações relevantes para o modelo UML, incluindo todos os elementos do modelo, seus relacionamentos e objetos Java, porém omitindo estereótipos.

Outra dificuldade encontrada foi que, apesar de o SMM ser um padrão estabelecido pela OMG para representação de métricas e medições, não existiam exemplos na literatura encontrada, demonstrando a maneira como é feita a representação e como é feita a extração e posterior representação dessas medições.

Por fim, também foi necessário um tempo considerável para o estudo e adaptação dos *frameworks* utilizados para o desenvolvimento, sendo necessário por vezes consultar a documentação formal e fazer adaptações para o contexto do ambiente.

CONCLUSÕES

O projeto atendeu aos objetivos propostos, visto que aplicou técnicas computacionais para construção do módulo, investigando e definindo formas de facilitar o processo de desenvolvimento sem abrir mão da qualidade no resultado final. Algumas técnicas investigadas foram: desenvolvimento orientado a objetos, considerando a interação entre objetos do mundo real, técnicas de reuso de software e *frameworks*. A aplicação dessas técnicas estão contidas no módulo.

O módulo construído integra-se com o padrão SMM (*Structured Metrics Metamodel*), para a definição de métricas. O ambiente é responsável por realizar a aplicação de métricas em modelos UML, estabelecendo uma forma de definir a operação da métrica através de palavra-chave e cláusula. Os resultados das medições coletadas pelo ambiente, opcionalmente podem ser salvas, permitindo a apresentação mais detalhada, incluindo dados comparativos, com o objetivo de facilitar a visualização.

Como trabalhos futuros, tem-se a construção do módulo responsável pela Modelagem da Arquitetura seguindo as especificações definidas para a abordagem de Linha de Produto, incluindo o gerenciamento de variabilidades.

REFERÊNCIAS

Capilla, R.; Bosch, J.; Kang, K. (2013) Systems and software variability management – concepts, tools and experiences. *Springer, New York, NY, USA*.

Clements, P.; Northrop, L. (2001) Software Product Lines: Practices and Patterns. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc.

Etzeberria, L.; Sagardui, G. (2008) Variability Driven Quality Evaluation in Software Product Lines. In: Proceedings of the Software Product Line Conference, Washington, DC, USA: IEEE Computer Society, p. 243- 252.

Gomaa, (2005) H. Designing Software Product Lines with UML: from Use Cases to Pattern-based Software Architectures. Boston, MA, USA: Addison-Wesley.

Honda, R. R. Modelagem e cômputo de métricas de interesse no contexto de modernização de sistemas legados. 2014. 121f.. Dissertação (Mestrado em Ciência da Computação) – Centro de Ciências Exatas e de Tecnologia, Programa de Pós-Graduação em Ciência da Computação, Universidade Federal de São Carlos, São Carlos, São Paulo.

Linden, F. J. v. d.; Schmid, K.; Rommes, E. (2007) *Software Product Lines in Action: The Best Industrial Practice in Product Line Engineering*. Secaucus, NJ, USA: Springer-Verlag New York, Inc.

Northrop, L. M. (2002) SEI's Software Product Line Tenets. *IEEE Software*, v. 19, n. 4, p. 32-40.

Oliveira Junior, E.; Gimenes, I.; Maldonado, J. Empirical Validation of Variability-based Complexity Metrics for Software Product Line Architecture. In: *Proc. Software Engineering and Knowledge Engineering*, 2012, p. 622-627.

Oliveira Junior, E.; Gimenes, I.; Maldonado, J.; Masiero, P.; Barroca, L. Systematic Evaluation of Software Product Line Architectures. *Journal of Universal Computer Science*, 2013, 19(1): 25-52.

Oliveira Junior, E.; Gimenes, I. Empirical Validation of Product-line Architecture Extensibility Metrics. In: *Proc. 16th International Conference on Enterprise Information Systems*, 2014, p. 111-118.

SMM. *A Structured Metrics Meta-model*. Disponível em: <http://www.omg.org/spec/SMM/1.1/>. Acesso em 02/02/2016.