Web-Atlas: Uma Ferramenta Web para Modelagem de Características

Fernando F. Lima[1], Luciano Marchezan[1], Marcelo Schmitt Laser[2], Elder Rodrigues[1], Maicon Bernardino[1]

[1]Universidade Federal do Pampa (UNIPAMPA)
Código Postal 97.546-550 – Alegrete – RS – Brasil

[2]Pontifícia Universidade Católica do Rio Grande do Sul (PUCRS)
Código Postal 90.619-900 – Porto Alegre – RS – Brasil

{fernando.fortlima,lucianomarchezan94,marcelo.laser}@gmail.com
elderrodrigues@unipampa.edu.br, bernardino@acm.org

Abstract. *This work presents Web-Atlas, a tool developed to assist the process of teaching-learning feature modeling of Software Product Lines (SPL). Web-Atlas is a stand-alone Web tool that searchc to provides an intuitive interface to the user. The tool provides basic features, such as: create, edit, save, load and validate feature models. With these features and characteristics, we believe that the tool may be used to support the teaching of SPL as well as be applie idn real projects in the industry.*

Resumo. *Este trabalho apresenta a Web-Atlas, uma ferramenta desenvolvida com o objetivo de auxiliar no processo de ensino-aprendizagem de modelagem de características em Linhas de Produtos de Software (LPS). A Web-Atlas é uma ferramenta Web stand-alone, a qual busca proporcionar uma interface intuitiva ao usuário. A ferramenta fornece funcionalidades básicas, tais como: criar, editar, salvar, carregar e validar os modelos de características. Com estas funcionalidades e características, acredita-se que a ferramenta possa ser utilizada no apoio ao ensino de LPS, bem como sua aplicação em projetos reais na indústria.*

1. Introdução

A área de Engenharia de Linhas de Produto de Software (ELPS) [Apel et al. 2013], como todas as outras engenharias, requer o uso de ferramentas para auxiliar no projeto e desenvolvimento de Linha de Produto de Software (LPS) ou na re-engenharia de uma família de sistemas para uma LPS. Estas ferramentas podem ser divididas entre *plugins*, como por exemplo a PlugSPL [Rodrigues et al. 2012] e FeaturePlugin [Antkiewicz and Czarnecki 2004], e sistemas *stand-alone*, tais como Software Product Line Online Tools (S.P.L.O.T.) [Mendonca et al. 2009] e webSPIFF [Bidarra et al. 2001] que são disponibilizadas via Web. Todas as ferramentas apresentam as funcionalidades essenciais para a modelagem de características, como importação e exportação de arquivos, disponibilidade online, e intuitividade na *interface* gráfica para o usuário. Sendo assim, algumas destas ferramentas são complexas ou não apresentam algumas funcionalidades para auxiliar a modelagem de modelos, tais como: representação gráfica ou validação.

Neste trabalho é apresentada a Web-Atlas, uma proposta de ferramenta que desempenha funções no auxilio da modelagem de caraterísticas, tais como: criar, editar, gravar, abrir e validar diagramas de características, disponibilidade online, e representação gráfica e textual do diagrama. A ferramenta Web-Atlas permite a criação de modelos utilizando a notação gráfica chamada Análise de Domínio Orientada a Características (*Feature-Oriented Domain Analysis* - FODA) [Kang et al. 1990].

Este trabalho está organizado em cinco seções. A Seção 2 apresenta os conceitos básicos e características de Linha de Produto de Software (LPS), na definição do que são modelos de característica e os motivos de seu uso, o que são diagramas de características e como são representados, bem como as definições e propriedades da notação FODA. A Seção 3 apresenta ferramentas similares e discussões sobre suas funcionalidades. A Seção 4 expõe os requisitos levantados para o desenvolvimento da ferramenta, incluindo discussões e decisões tomadas, as funcionalidades presentes, e explora um exemplo de uso. Por último, a Seção 5 aponta as considerações finais e trabalhos futuros.

2. Fundamentação Teórica

O conceito de Linha de Produtos de Software (*Software Product Line - SPL*) é baseado no conceito de Linha de Produto comum às Engenharias, ou seja, é um conjunto de softwares que compartilham de características semelhantes, com o objetivo de produzir softwares diferentes que utilizam deste mesmo conjunto de características [SEI 2017]. A LPS é composta por duas características: plataforma e variabilidade [Pohl et al. 2005]. Plataforma é qualquer base tecnológica sobre a qual outras tecnologias ou processos são construídos. Plataforma também pode ser definida como o "núcleo do sistema", contendo as características comuns a todos os sistemas de uma família de software. Plataformas auxiliam no planejamento e projeto dos artefatos que serão usados em dois ou mais produtos. Variabilidade se trata da flexibilidade, ou seja, da habilidade de criar artefatos que serão reusados em uma família de software [Kang et al. 1990]. Em outras palavras, é a habilidade de um sistema de dar suporte a produção de um conjunto de artefatos que possuem pontos diferentes entre eles [Bachmann and Clements 2005].

Um modelo de características documenta as características de uma linha de produto, seus relacionamentos, e é uma abordagem que expressa a variabilidade e semelhanças dentro de uma família de softwares. De acordo com Kang *et al.*, "Características são aspectos visíveis ao usuário ou características do domínio" [Kang et al. 1990]. Os modelos de características são usados no processo de desenvolvimento de LPS, representam todas as possibilidades de combinações entre as características de uma família para o desenvolvimento de produtos de software.

Um diagrama de características é uma notação gráfica para especificar graficamente um modelo de características, podendo ser construídos e apresentados de acordo com diferentes notações [Apel et al. 2013]. Neste trabalho, é adotada a representação em estrutura de árvore, em que os nós representam as características e as arestas são os relacionamentos pai-filhas. A representação de diagramas de características pode ser feita com diversas notações, tais como: a notação FODA [Kang et al. 1990] e *Generative Programming* [Czarnecki et al. 2004]. A notação FODA foi apresentada com o objetivo principal de representar, de forma gráfica, um modelo que permite a visualização de todas as configurações que podem ser produzidas a partir do mesmo conjunto de características.

Esta notação é composta pelas seguintes propriedades: **a)** *Raiz*: é usada para representar o conceito do domínio, ou seja, para definir o conceito genérico. Por meio dela serão especificadas as características e criadas as configurações para cada produto de software dentro da família; **b)** *Características*: são representadas como obrigatórias, opcionais ou alternativas. As obrigatórias são características que estão presentes em qualquer configuração. As opcionais são características que podem ou não ser selecionadas pelo usuário. As alternativas formam um grupo de características, sendo possível a seleção de apenas uma dessas características; **c)** *Grupos Alternativos*: restringem que o usuário escolha apenas uma das características contidas em um grupo; **d)** *Regras de Composição*: representam duas restrições textuais, "requer" e "mutuamente exclusivo com"; **e)** *Razão*: apresenta de forma textual motivos ou justificativas para auxiliar na tomada de decisão.

A Figura 1 ilustra estas características por meio de um exemplo da composição de um carro. Neste exemplo, a raiz, nomeada de *Carro*, representa o conceito do domínio. *Transmissão* e *Potência*, são exemplos de características obrigatórias para o sistema. As características *Manual* e *Automático* são especializações da *Transmissão*; elas representam um grupo alternativo, ou seja, uma e somente uma delas deve ser escolhida. *Ar Condicionado* é apresentado como característica opcional no sistema; esta característica pode ou não ser selecionada.

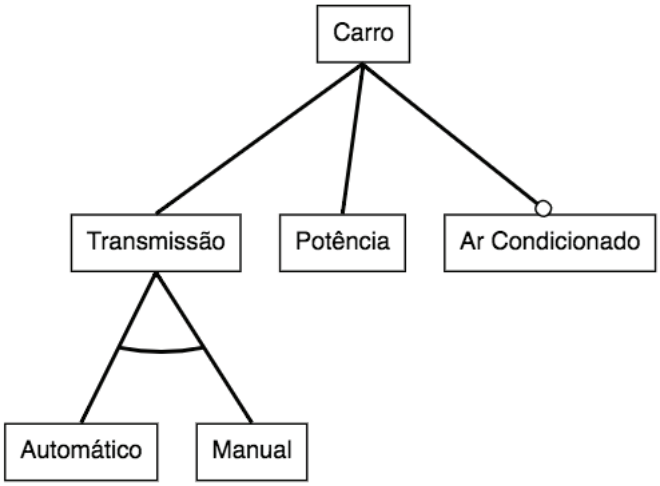


Figura 1. Exemplo da Notação FODA (Adaptado de [Kang et al. 1990])

3. Trabalhos Relacionados

Para o levantamento dos requisitos e funcionalidades requeridos pela ferramenta foram apontadas algumas ferramentas que podem ser usadas para auxiliar na LPS.

O WebSPIFF [Bidarra et al. 2001] é um sistema Web comercial para modelagem colaborativa de modelos de características. O sistema tem uma arquitetura cliente-servidor composto por um sistema de modelagem e um gerenciador de sessões. O servidor fornece a colaboração além de funcionalidades mais avançadas, tais como: gerenciamento de sessões e validação dos modelos. Enquanto que o lado cliente fornece as funcionalidades mais utilizadas, tais como a visualização do modelo e *interface* interativa de especificação.

A FeatureIDE [Kastner et al. 2009] se trata de um *framework open source* que suporta os processos da linha de produto em uma infra-estrutura de ferramenta. O resultado é uma ferramenta que da suporte ao *AHEAD Tool Suite*, tornando mais fácil a adoção de resultados de pesquisas para projetos da academia ou da indústria. O *framework* final dá suporte a múltiplas linguagens e ferramentas.

A FeaturePlugin [Antkiewicz and Czarnecki 2004] foi apresentada como um *plugin* para o Eclipse IDE, que suporta a notação baseada em cardinalidade [Czarnecki et al. 2004]. A *interface* de usuário para a apresentação dos modelos de características permite um meio eficiente na criação e edição de modelos com tamanhos relativamente grandes. Porém, a ferramenta suporta a edição somente via teclado.

A S.P.L.O.T. [Mendonca et al. 2009] é uma ferramenta Web para configuração e pensamento lógico para LPS. O sistema oferece uma *interface* Web, a qual utiliza um grande conjunto de algoritmos para configuração e *logic solvers* relacionado a modelos de características. Além disso, o sistema também mantém um repositório de modelos de características online para compartilhamento de conhecimento.

A PlugSPL [Rodrigues et al. 2012] é um ambiente automatizado que dá suporte a LPS baseadas em *plugins*. O ambiente é dividido em três módulos: Projeto de LPS, Configuração do Produto e Geração do Produto. A proposta se difere do que já existe, pois visa dar suporte a LPS baseadas em *plugins*, podendo importar e exportar modelos de características para outras ferramentas e gerar produtos de maneira eficiente.

As ferramentas citadas apresentaram certas características, tais como: validação dos modelos, de qual forma a ferramenta é disponibilizada, a possibilidade de realizar trabalhos colaborativos, construção e representação de modelos gráficos ou textuais. Sendo assim, a Web-Atlas possui características que correspondem com a maioria das características implementadas pelas outras ferramentas, exceto pelo trabalho colaborativo.

4. Web-Atlas

A Web-Atlas é uma ferramenta que visa auxiliar no aprendizado acadêmico na área de modelagem de características de linhas de produtos de software. Foi desenvolvida visando apresentar uma *interface* intuitiva e ser disponível online. Para seu desenvolvimento, foram levantados alguns requisitos mínimos para o funcionamento da ferramenta (RQ). Estes requisitos são apresentados a seguir: **RQ1.** A ferramenta deve estar disponível para os usuários via Web para facilitar o acesso e não necessitar a instalação de programas e *plugins*; **RQ2.** A ferramenta deve dar suporte aos navegadores: Google Chrome (versão 4.0), FireFox (versão 2.0), e Safari (versão 3.1), incluindo também os navegadores para dispositivos móveis. Este requisito é necessário para alcançar maior número de usuários e evitar que o navegador venha a ser um obstáculo para a utilização da ferramenta; **RQ3.** A ferramenta deve possibilitar que o usuário crie, de forma gráfica, um modelo de características utilizando a notação FODA; **RQ4.** A ferramenta deve possibilitar que o usuário salve localmente o modelo criado de forma textual (formato .json). Salvar de forma textual possibilitará que o modelo seja carregado ou possa ser modificado sem a necessidade de conexão com a Internet. **RQ5.** A ferramenta deve possibilitar que o usuário salve localmente o modelo criado de forma gráfica (formato .png). Salvar de forma gráfica ajuda a visualizar o modelo de forma clara e simples, permitindo o uso da imagem em outros documentos; **RQ6.** A ferramenta deve disponibilizar a opção de

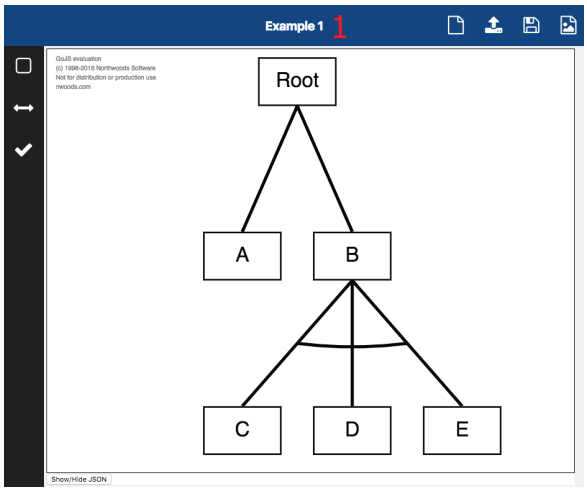
fazer *upload* de um modelo criado. Este requisito possibilita que um modelo criado, independente do usuário, possa ser carregado no sistema e modificado; **RQ7.** A ferramenta deve fornecer ao usuário a opção de validar a estrutura do modelo de características. Este requisito é necessário para auxiliar na modelagem de modelos por usuários inexperientes na notação ou em ELPS; **RQ8.** As telas de comunicação com o usuário devem ser minimalistas, contendo somente o que for necessário ao usuário para que a *interface* seja intuitiva e simples de usar.

Grande parte do esforço do levantamento de requisitos foi dedicado ao processo de seleção da notação e funcionalidades básicas disponibilizadas pela ferramenta. Foram considerados que este seja um ponto importante, visto que a ferramenta deve auxiliar uma pessoa que não possua experiência em ELPS possa representar, graficamente, uma família de software por meio de um processo de modelagem de características. Portanto, foi decidido que a primeira notação a ser disponibilizada seria a notação FODA, por conta de sua facilidade de implementação e quantidade de propriedades. Vale ressaltar que esta notação contém o menor número de propriedades dentre todas as estudadas.

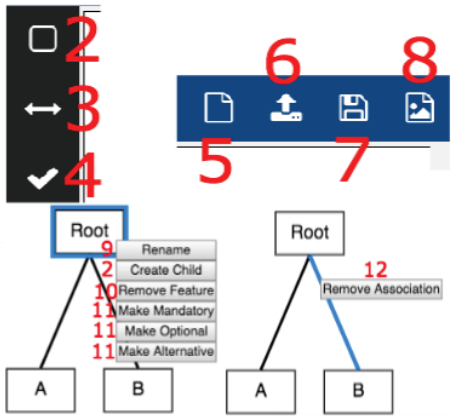
Um conjunto de funcionalidades básicas foram definidas para a ferramenta visando criar uma versão funcional, tais como criação, mudança do tipo e exclusão das características, e criação e exclusão de relacionamentos. Com estas funcionalidades implementadas e testadas, foram utilizadas validações informais com usuários durante a fase de desenvolvimento e as funcionalidades eram incluídas ou excluídas a cada validação.

4.1. Funcionalidades

As funcionalidades implementadas na Web-Atlas foram definidas a partir dos requisitos descritos nesta seção. A seguir são apresentadas três figuras representando as telas e funcionalidades da ferramenta. A Figura 2a apresenta a tela principal da ferramenta, em que estão presentes a barra de ferramentas de modelagem na lateral, menu superior da direita e a área central para modelagem. A Figura 2b mostra o menu lateral e o menu superior da direita. As figuras possuem uma numeração em vermelho, que identifica as funcionalidades juntamente às suas descrições:



(a) Tela Inteira



(b) Menu e Barra de Ferramentas

Figura 2. Tela e Menus

1. *Renomear Modelo*: permite que o usuário possa nomear seu modelo. O sistema restringirá algumas funcionalidades caso o modelo não tenha sido nomeado; **2.** *Criar Característica*: faz com que o sistema crie uma característica no modelo, esta característica, por padrão, é do tipo obrigatória. Esta função pode ser acessada no menu lateral ou clicando diretamente no modelo com o botão esquerdo. Porém, caso o usuário selecione uma característica e use o botão direito, o sistema criará uma nova característica com associação com a que estava selecionada anteriormente; **3.** *Criar Associação*: permite que o usuário crie associações entre as características. Quando selecionada esta função, o sistema solicita ao usuário o nome da característica pai e filha, criando assim a associação. Essa função pode ser acessada pelo menu lateral ou clicando com o botão direito sobre o modelo; **4.** *Verificar Modelo*: realiza a verificação das regras da estrutura de árvore e as propriedades da notação FODA. É verificado se há características sem associação, mais de uma raiz, um grupo de alternativas formado por apenas uma característica, entre outras restrições; **5.** *Criar Modelo*: permite que um novo modelo de características seja criado, ou seja, ele começa um novo documento, apagando todo o progresso de modelagem realizado anteriormente; **6.** *Carregar Modelo*: permite ao usuário abrir um modelo que foi criado e salvo como .json [Bray 2014]. Sendo assim, o usuário pode realizar alterações nos modelos; **7.** *Salvar Modelo*: faz com que o sistema mapeie tudo que foi criado no modelo pelo usuário, realize a conversão de linguagem JavaScript para JSON e transfira para a máquina do usuário um arquivo de extensão .json; **8.** *Exportar Modelo como Imagem*: realiza a captura de todos os elementos inseridos no modelos, criando assim uma imagem com extensão .png e salvando-a para a máquina do usuário; **9.** *Renomear Característica*: permite que o usuário modifique o nome da característica. Há duas formas de acessar essa função, clicando duas vezes sobre uma característica ou clicando com o direito sobre a característica e selecionando a opção "Renomear Característica"; **10.** *Remover Característica*: remove uma característica do modelo. Caso esta característica tenha associações filhas e uma associação pai, essas associações serão também removidas, mantendo as características separadas no modelo; **11.** *Alterar o Tipo da Característica*: permite que o usuário altere o tipo da característica. Os tipos podem ser: *Mandatory* (Obrigatório), *Alternative* (Alternativo) e *Optional* (Opcional). Para acessar essa função, deve-se clicar com o direito sobre uma característica e selecionar uma das três opções: *Make Mandatory*, *Make Optional*, *Make Alternative*. **12.** *Remover Associação*: possibilita ao usuário excluir uma associação do modelo, sem a necessidade de remoção das características pai e filha.

4.2. Exemplo de Uso: Linha de Produto de Celular

Com o objetivo de demonstrar as funcionalidades básicas da ferramenta Web-Atlas utilizou-se como exemplo um diagrama de funcionalidades de uma LPS de celular[1]. Este exemplo tem por objetivo mapear as característica de multimídia de um celular, tais como organização, ações, e gerenciamento dessas multimídias. O primeiro passo é utilizar a função de nomear o modelo, pois algumas funções necessitam do nome para criar arquivos externos. Com o modelo nomeado, a próxima etapa é usar a função de renomear as características para definir o nome do domínio a ser representado, neste caso, o nome atribuído foi "Multimídia para Celular".

Após a definição do domínio, são utilizadas as funções de criar e renomear as ca-

[1]Modelo retirado do repositório da S.P.L.O.T. [Mendonca et al. 2009]

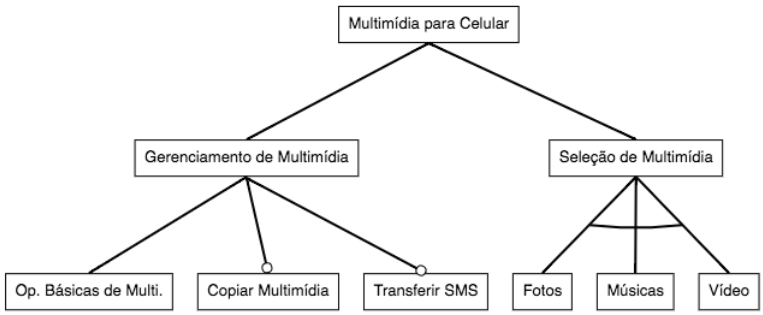


Figura 3. Modelo representando o domínio Multimídia para Celular

racterísticas, definindo assim as características mais genéricas e as filhas da raiz, sendo elas "Gerenciamento de Multimídia" e "Seleção de Multimídia". Nas próximas etapas, deve-se repetir este mesmo procedimentos, para especializar estas características. Para a especialização (filhas) da característica "Gerenciamento de Multimídia" (pai), foram definidas as filhas "Operações Básicas de Multimídia", "Copiar Multimídia", e "Transferir SMS", que são características opcionais, portanto, deve-se usar a função "*Make Optional*" para modificá-las para opcionais. A última característica filha da raiz é definida como pai de um grupo de características alternativas, sendo elas "Foto", "Vídeo" e "Música", em que utilizou-se a função "*Make Alternative*" para torná-las alternativas, então, o modelo finalizado é apresentado na Figura 3.

Após a finalização da representação das características do domínio, há a opção de utilizar a função "*Check Model*", para validar o modelo criado. Esta validação verifica certos critérios da notação, tais como: o modelo deve possuir somente uma raiz, um grupo alternativo não pode conter somente uma característica, uma característica não pode possuir dois pais, não podem haver características sem pai, não podem haver características sem nome, e a raiz não pode ser alternativa e nem opcional. Com o modelo validado, o próximo passo é salvar o modelo ou exportar como imagem. Caso selecione a função "*Save as .json*", o modelo será convertido para um arquivo .json e salvo para o usuário. O JSON gerado é em um formato específico utilizado pela Web-Atlas, como o apresentado na Figura 4. Caso seja escolhida a função "*Save as .png*", o modelo será transformado em uma imagem e transferido para o usuário.

```
{ "class": "go.GraphLinksModel","linkLabelKeysProperty": "labelKeys","nodeDataArray": [
{"key":"Multimídia para Celular", "category":"mandatory"},
…
"category":"LinkLabel"},{"key":"Alt1", "category":"LinkLabel"}],"linkDataArray": [
{"from":"Seleção de Multimídia-Vídeo", "to":"Seleção de Multimídia-Música", "labelKeys":
[ "Alt1" ],"category":"linkToLink"},
…
{"from":"Seleção de Multimídia", "to":"Vídeo", "labelKeys":[ "Seleção de Multimídia-
Vídeo" ]}]}
```

Figura 4. Exemplo simplificado do JSON gerado pela Web-Atlas

5. Conclusão

Inicialmente, neste trabalho foram apresentados os conceitos básicos de LPS, modelos e diagramas de características, incluindo as propriedades da notação FODA. Realizando-se o levantamento da características essenciais das ferramentas relacionadas e da Web-Atlas,

percebe-se que há apenas uma característica que não está implementada na Web-Atlas, sendo ela a possibilidade de trabalho colaborativo. Apenas a WebSPIFF contém todas as características implementadas, porém, sua licença é comercial. Outras ferramentas possuem características próprias, tais como a S.P.L.O.T., que mantém um repositório online com modelos criados por seus colaboradores, que podem servir como exemplos de uso ou aprendizado.

A Web-Atlas é a evolução da primeira versão da Atlas, desenvolvida visando auxiliar a área acadêmica, sendo possível sua utilização na área industrial. Esta ferramenta tem como funcionalidade criar, editar, salvar, abrir e exportar os modelos de características baseados na notação FODA. Para trabalhos futuros, as principais metas são: a implementação de novas notações, um estudo sistemático para verificar quais são as notações em destaque na comunidade de LPS, a validação com usuários e casos de testes, e a implementação de transformação entre os modelos suportados pela ferramenta.

Referências

Antkiewicz, M. and Czarnecki, K. (2004). FeaturePlugin: Feature Modeling Plug-in for Eclipse. In *OOPSLA*, pages 67–72, New York, NY, USA. ACM.

Apel, S., Batory, D., Kästner, C., and Saake, G. (2013). *Feature-Oriented Software Product Lines: Concepts and Implementation*. Springer-Verlag Berlin Heidelberg.

Bachmann, F. and Clements, P. C. (2005). Variability in software product lines. Technical report, Carnegie-Mellon Univ. Pittsburgh PA software engineering Inst.

Bidarra, R., van den Berg, E., and Bronsvoort, W. F. (2001). Web-based collaborative feature modeling. In *SMA*, pages 319–320, New York, NY, USA. ACM.

Bray, T. (2014). The javascript object notation (json) data interchange format.

Czarnecki, K., Helsen, S., and Eisenecker, U. (2004). Staged configuration using feature models. In *SPLC*, pages 266–283. Springer.

Kang, K. C., Cohen, S. G., Hess, J. A., Novak, W. E., and Peterson, A. S. (1990). Feature-oriented domain analysis (foda) feasibility study. Technical report, DTIC Document.

Kastner, C., Thum, T., Saake, G., Feigenspan, J., Leich, T., Wielgorz, F., and Apel, S. (2009). FeatureIDE: A tool framework for feature-oriented software development. In *ICSE*, pages 611–614. IEEE.

Mendonca, M., Branco, M., and Cowan, D. (2009). S.P.L.O.T.: Software Product Lines Online Tools. In *ACM SIGPLAN*, pages 761–762, New York USA. ACM.

Pohl, K., Böckle, G., and van Der Linden, F. J. (2005). *Software product line engineering: foundations, principles and techniques*. Springer Science & Business Media.

Rodrigues, E., Zorzo, A., de Oliveira, E., Gimenes, I., Maldonado, J. C., and Domingues, A. (2012). PlugSPL: An Automated Environment for Supporting Plugin-based Software Product Lines. In *SEKE*, pages 647–650.

SEI, T. S. E. I. (2017). Software product lines. `http://www.sei.cmu.edu/productlines/`. Acessado: 30-09-2017.