CRISTIANE YANASE HIRABARA DE CASTRO

UMA ABORDAGEM PARA O DESENVOLVIMENTO DE LINHA DE PRODUTO ORIENTADO A ASPECTOS

MARINGÁ
2008

CRISTIANE YANASE HIRABARA DE CASTRO

UMA ABORDAGEM PARA O DESENVOLVIMENTO DE LINHA DE PRODUTO ORIENTADO A ASPECTOS

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Estadual de Maringá, como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação.
Orientadora: Prof ª. Drª. Itana Maria de Souza Gimenes

MARINGÁ
2008

Dados Internacionais de Catalogação – na – Publicação (CIP)

(Biblioteca Municipal de Maringá, Maringá – PR., Brasil)

Castro, Cristiane Yanase Hirabara de

C35a Uma abordagem para o desenvolvimento de linha de produto orientado a aspectos / Cristiane Yanase Hirabara de Castro. – Maringá: [s.n.], 2009.

152 f. : 1l.

Orientador: Profa. Dra. Itana Maria de Souza Gimenes

Dissertação (mestrado) – Universidade Estadual de Maringá. Programa de Pós- graduação em Ciência da Computação, 2009.

1. Engenharia de Software. 2. Desenvolvimento de Linhas de Produto de software. 3. Desenvolvimento de Software Orientado a Aspectos. 4. Reutilização. 5.Processo de Gerenciamento de Variabilidades. I. Universidade Estadual de Maringá.Programa de Pós- graduação em Ciência da Computação. II. Título.

CDD 21.ed. 001.61

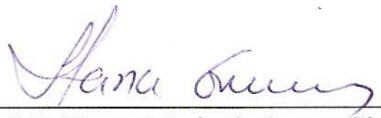
CRISTIANE YANASE HIRABARA DE CASTRO

UMA ABORDAGEM PARA O DESENVOLVIMENTO DE LINHAS DE PRODUTO ORIENTADO A ASPECTOS

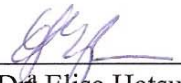
Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Estadual de Maringá, como requisito parcial para obtenção do grau de Mestre em Ciência da Computação.

Aprovada em 15/01/2009.

BANCA EXAMINADORA



Profª Drª Itana Maria de Souza Gimenes
Universidade Estadual de Maringá – DIN/UEM



Profª Drª Elisa Hatsue Moriya Huzita
Universidade Estadual de Maringá – DIN/UEM

Profª Drª Jandira Guenka Palma
Universidade Estadual de Londrina – DC/UEL

Para o meu esposo Marcelo,
com a minha gratidão pelos 15 anos compartilhados
e aos meus filhos Henrique e Heitor
com o mais profundo amor.

Agradecimentos

Muitos agradecimentos ficaram para trás ao longo deste dissertar, espero que haja espaço suficiente para citar todos os nomes.

Agradeço ao meu Deus, força divina que me criou, guiou os meus passos e encheu minha vida de oportunidades e de pessoas especiais. Com as oportunidades realizei sonhos e às pessoas segue aqui meus agradecimentos.

Agradeço ao meu esposo, Marcelo e aos meus filhos, Henrique e Heitor. É o amor de vocês que inspira o melhor de mim. Vocês são a minha estrela, minha lua, meu sol !!

Aos meus pais, Orlando e Celina, minhas irmãs, Luciane e Elizabete pelo amor incondicional e pelo exemplo de uma vida inteira. Amo muito vocês!

Agradeço à professora Itana, por ter acreditado em mim e aceitado me orientar. A você agradeço pela paciência e por me permitir compartilhar de suas experiências e de seu conhecimento. Obrigada, Itana! Dedico a você todo meu carinho, meu orgulho e minha admiração !!

Agradeço aos meus tios, Nilo e Elaine, e aos meus primos, Thais e Arthur, por terem sido minha segunda família, por me acolherem e me incentivarem durante minha estada em Maringá.

Sou grata também aos meus sogros, Selma e Aparecido, minhas cunhadas e meus cunhados, pelo carinho com que cuidaram da minha família na minha ausência e por terem me apoiado sempre.

Agradeço ao amigo querido e pesquisador fantástico, Roberto Pereira, por me permitir fazer parte de sua vida, por ter estado sempre ao meu lado e acima de tudo por me

compreender (mesmo quando "*non me lo so spiegare*"). Obrigada Roberto, você é a minha força e é meu porto seguro. Amo você !!

Agradeço ao amigo José Valderlei (JV, Jesus, Vander++) pelo inspirado apoio no início e durante toda essa caminhada. Você foi um verdadeiro presente nesse mestrado!

Agradeço ao Bruno, meu irmão de orientação, obrigado pelo companheirismo no compartilhamento de idéias, materiais e aprendizado. Valeu Tigrão !!!!

E aos demais amigos do mestrado: César, Márcio, Mário, Mauro, Rogério, Walter e Zé Luiz, porque por algum motivo o destino nos colocou juntos no inicio dessa caminhada. Valeu por ter conhecido vocês!

Agradeço aos amigos da CPV/FALM: Alba, Eder, Biluka e Walter Candioto. Obrigada pelo apoio e pela paciência !!

Meus agradecimentos ao Edson de Oliveira Junior e ao Marcelo Eler, por terem inspirado cada capítulo dessa dissertação através do excelente trabalho que fizeram.

Agradeço à Inês e ao Robson, da secretaria do mestrado. Obrigada pela extrema competência em nos atender, pela dedicação e pelo carinho de vocês!

Sou grata e preciso reconhecer o auxílio e a compreensão dos meus amigos e colegas do departamento de informática da UENP/CALM. Não sei que ordem seguir e se conseguirei ser justa com todos, afinal vocês sacrificaram seu tempo, se sobrecarregaram de atividades e funções e merecem um espaço especial nesses agradecimentos. Assim, segue meus agradecimentos ...

... ao **Ailton** Sérgio Bonifácio pela tonalidade inquietante, o brilho que contagia e pela alma exposta! Valeu, keridão !!

... ao **André** Menolli por compartilhar comigo o gosto pelos livros e pela leitura, pela sua paciência e por nunca ter me deixado brigar com você !!

... à nossa secretária Aparecida Taguti (**Cidinha**) por sempre ter cuidado de mim e de todos no departamento!!

... ao Carlos Eduardo Ribeiro (**Biluka**) pelas aulas de motivação e otimismo !!

...ao **Christian** James Bussman pelas sextas-feiras compartilhadas, pelas viagens matemáticas e pelas trocas de conhecimento !

... à **Daniela** de Freitas Guilhermino Trindade, por dividir comigo a jornada de desenvolvimento dessa dissertação. Agradeço a você e ao esposo Luciano, por me adotarem pelos quilômetros de estrada e por transformarem essas viagens em momentos tão agradáveis.

... ao Ederson Marcos **Sgarbi**, sou grata pelo seu incentivo e pela sua paciência. Muito obrigada, Sgarbi !!

... ao Luiz **Fernando** Legore do Nascimento, por sempre ter uma palavra amiga de consolo e incentivo.

... ao **Glauco** Carlos Silva por compartilhar comigo dos risos imprevisíveis e sempre explicar o que está implícito nas piadas!

...ao José Reinaldo **Merlin** por ter sempre algo (e ter sempre um tempo) para me ensinar! Aprendi muito com você e isso ... não tem preço !!

... ao Luiz Roberto **Lomba** pelos sábios conselhos e pelos silêncios estratégicos! E também é claro, pelos churrascos memoráveis !!

... à **Marília** Abrahão Amaral pelos ótimos momentos de descontração, pelos risos e sorrisos!

....ao **Ricardo** Gonçalves Coelho por ter sempre um tempo para ouvir e aconselhar. Você é um amigo incrível e uma pessoa absolutamente brilhante !!

... ao **Viní**cius Rodrigues Silva, do provedor, por estar sempre pronto a nos atender (e nos socorrer) com um sorriso!!

... à **Vivi**ane Bartholo pelo companheirismo, pela paciência e acima de tudo por ser essa pessoa meiga e tão especial!

Finalmente, agradeço à Fundação Araucária e ao CNPq pelo apoio financeiro

Resumo

Este trabalho apresenta uma investigação sobre como os conceitos do paradigma orientado a aspectos podem ser combinados com a abordagem de linha de produto de software (LPS) para o encapsulamento de grupos de características em componentes bases e transversais. Como resultado dessa investigação é proposta uma abordagem, denominada ProLineA, para o desenvolvimento de linhas de produto orientado a aspectos. A abordagem de LPS tem como objetivo gerar produtos específicos por meio da reutilização de uma infra-estrutura central estabelecida para um determinado domínio. O desenvolvimento de uma LPS envolve duas atividades: a engenharia de domínio e a engenharia de aplicação. O contexto desse trabalho abrange a engenharia de domínio combinada às técnicas que promovem a separação de características transversais do paradigma orientado a aspectos. São apresentados as etapas, atividades, notações e artefatos produzidos pela ProLineA por meio de um exemplo de LPS para sistemas de locação de carros. Foi dado ênfase ao Processo de Gerenciamento de Variabilidades (PGV) e ao método para o Desenvolvimento Baseado em Componentes e Aspectos (DBC/A). Um exemplo de aplicação de LPS para o Gerenciamento de Processos Seletivos é apresentado e comparado a uma solução para o mesmo domínio sem a aplicação da ProLineA.

Abstract

This work presents an approach, named ProLineA, that applies aspect-oriented concepts to the development of software product line (SPL). The SPL approach aims at generating specific products by reusing a core asset of a defined domain. It considers two main activities: domain and application engineering. The proposed approach extended the domain engineering process by adding separation of concerns concepts. Our approach is mainly based on two existing techniques for variability management (PGV) and component-based development with aspects (DBC/A). The development stages and notation of ProLineA is described illustrated with artifacts excerpts of the development of a SPL for the management of admission processes. In addition, an example of application of the domain of car rental is used to evaluated our approach.

Lista de Ilustrações

Lista de Tabelas

Lista de Abreviaturas e Siglas

AOP	Aspect Oriented Programing
AORE	Aspect Oriented Requirements Engineering
AORA	Aspect Oriented Requirements Analisys
CMM	Capability Maturity Model
DBC	Desenvolvimento Baseado em Componentes
DBC/A	Desenvolvimento Baseado em Componenetes com Aspectos
DCU	Diagrama de Casos de Uso
DSOA	Desenvolvimento de Software Orientado a Aspectos
FPS	Família de Produtos de Software
GPS	Gestão de Processos Seletivos
LP	Linha de Produto
LPS	Linha de Produto de Software
LP-GPS	Linha de Produto de Gestão de Processos Seletivos
LP-SLC	Linha de Produto de Sistemas de Locação de Carros
LPS	Linha de Produto de Software
MOA	Modelagem Orientada a Aspectos
MCN	Modelo Conceitual de Negócio
MCN/F	Modelo Conceitual de Negócio Funcional
MCN/NF	Modelo Conceitual de Negócio Não Funcional
OO	Orientado a Objetos
OA	Orientação a Aspectos
PGV	Processo de Gerenciamento de Variabiliadade
PLP	Product Line Practice
POA	Programação Orientada a Aspectos
SEI	Software Engineering Institute
SLC	Sistemas de Locação de Carros
UEM	Universidade Estadual de Maringá
UML	Unified Modeling Process
RUP	Rational Unified Process

Sumário

CAPÍTULO

1

Introdução

1.1 Motivação

A busca de qualidade no processo de desenvolvimento de software vem absorvendo grandes esforços da comunidade de engenharia de software. Essa busca é resultante tanto do aumento da demanda por produtos de software cada vez mais complexos quanto da insatisfação dos clientes diante de falhas comuns em sistemas de informação existentes. Existe uma preocupação da comunidade em atender as necessidades de mercado, agilizar e buscar qualidade em todo processo de desenvolvimento de software.

Características relacionadas às mudanças no tipo e na arquitetura das aplicações e regras de negócios complexas e dinâmicas são alguns exemplos de exigências que passaram a interferir nas decisões de projeto. Desta forma, várias técnicas e abordagens foram disponibilizadas nas últimas décadas buscando atender essas necessidades. A reutilização de software é uma das abordagens que explora esse cenário e propõe soluções que permitem aumentar a qualidade dos produtos de software bem como diminuir seu *time-to-market*. A reutilização de software consiste na criação de sistemas a partir de módulos de software preexistente (KRUEGER, 1992).

Inicialmente, os esforços relacionados à reutilização foram direcionados ao desenvolvimento de um produto de software específico, porém com as técnicas de engenharia

de domínio e o Desenvolvimento Baseado em Componentes (DBC), passou-se a vislumbrar a concepção de modelos para um determinado "domínio" de mercado a partir do qual vários produtos específicos pudessem ser concebidos.

Produtos de software tendem a possuir características comuns com algumas variações ou características específicas que diferenciam um produto de outro. Tornou-se necessário então uma abordagem para sistematizar e controlar as configurações de tipos de produtos de software pertencente ao mesmo domínio. Assim, surgiu a abordagem de Linha de Produto de Software (LPS). Uma LPS consiste de um conjunto de sistemas com características comuns que visam satisfazer um segmento particular do mercado e que são desenvolvidas de forma sistematizada a partir de um núcleo comum de artefatos (SEI, 2004; GIMENES;TRAVASSOS, 2002). As técnicas de LPS destacam-se por serem bem aceitas no contexto industrial. Exemplos de empresas que fazem uso de LPS são Philips Medical Institute (HALL OF FAME, 2008), Bosch (STEGER, 2004), LSI Logic (HETRICK et al, 2006), General Motors (GMVISIT, 2008), Nokia (HALL OF FAME, 2008) entre outros. Isso é decorrente do fato de que grandes empresas de software são especializadas em determinados domínios a partir dos quais concebem produtos específicos para seus diferentes clientes, assim desejam alcançar um alto nível de maturidade, buscando qualidade e produtividade a custos reduzidos.

Abordagens de LPS procuram integrar várias técnicas de reutilização desde de linguagens orientadas a objeto até o DBC, incluindo técnicas como biblioteca de classes, *frameworks* e padrões. Esses são exemplos da busca de um maior nível de reutilização e manutenibilidade, aumentando assim, a produtividade do desenvolvimento e o suporte para a adição de novos requisitos.

Contudo, a simples adoção destas técnicas não garante a qualidade do software, já que o paradigma OO possui algumas limitações, como o entrelaçamento e o espalhamento de código com diferentes propósitos (OSSER; TARR, 1999). Existem algumas formas de contornar esses problemas usando técnicas OO como, por exemplo, a utilização de padrões de projetos (GAMMA, 2000). Para poucas classes e/ou poucos objetos isto pode funcionar adequadamente. Entretanto, à medida que o número de objetos aumenta, cresce também a quantidade de memória necessária e o número explícito de associações entre os objetos.

Para superar essas limitações, extensões do paradigma OO são propostas, como a Programação Orientada a Assunto (*Subject-Oriented Programming*) (OSSER; TARR, 1999),

a Programação Adaptativa (*Adaptive Programming*) (LIEBERHER, 1994) e a Programação Orientada a Aspectos (*Aspect- Oriented Programming*) (KICZALES, 1997). Essas novas técnicas de programação procuram aumentar a modularidade do código, onde a orientação a objetos e os padrões de software não oferecem o suporte adequado.

O objetivo das abordagens orientadas a aspectos é propor novas técnicas para promover a separação de características transversais de forma a complementar o paradigma orientado a objetos. No início, os primeiros estudos sobre aspectos eram relacionados à fase de implementação. A consolidação de tecnologias de apoio à programação como, o AspectJ e o JBoss, cujos resultados de pesquisas podem ver encontrados na literatura (KICZALES, 1997; FILMAN e FRIEDMAN, 2000; ELRAD et al, 2001; GARCIA, 2004), aumentaram o contexto das pesquisas relacionadas a aspectos para as fases preliminares do desenvolvimento de software. Neste contexto, estão incluidas pesquisas desde *early aspects*, até projeto e teste de software. São encontrados na literatura resultados como a abordagem TEMA (CLARKE E BANIASSAD, 2004), abordagens baseadas em UML (SUZUKI E YAMAMOTO, 1999; PAWLAK, 2002; CHAVEZ, 2003;) e até mesmo abordagens para LPS orientadas a aspectos (PACIOS, 2007). Porém, nenhum desses trabalhos relacionam o desenvolvimento de LPS, o gerenciamento de variabilidades e a utilização dos conceitos do paradigma orientado a aspectos, como é proposto nesse trabalho.

Os objetivos e a organização do trabalho são apresentados nas seções a seguir.

1.2 Objetivos

1.2.1 Objetivos Gerais

O objetivo deste trabalho de mestrado é propor uma solução para o desenvolvimento de LPS orientado a aspectos. A solução busca reunir os conceitos de gerenciamento de variabilidade, DBC e aspectos. A abordagem proposta é denominada ProLineA.

1.2.2 Objetivos Específicos

A concepção da abordagem ProLineA envolveu a realização de vários etapas:

- Estudar conceitos de LPS.
- Pesquisar e estudar abordagens de LPS.

- Estudar conceitos do paradigma orientados a aspectos que envolvem o desenvolvimento de software orientado a aspectos (DSOA) e a programação orientada a aspectos (POA).
- Pesquisar e estudar abordagens de DSOA e POA.
- Encontrar possíveis soluções para os problemas encontrados na pesquisa teórica e propor soluções com a concepção da abordagem ProLineA.

1.3 Metodologia

Em um primeiro momento, foram estudadas abordagens para o desenvolvimento de LPS, para o gerenciamento de variabilidades e para o desenvolvimento baseado em componentes. Para explorar os conceitos foi iniciado um projeto para o desenvolvimento de uma LPS para gestão de processos seletivos (LP-GPS). A partir deste exemplo, foi produzido a documentação da modelagem para a LP-GPS, que mais tarde foi reutilizada para uma análise comparativa e observacional, através da aplicação da ProLineA.

Em um segundo momento, foram estudadas abordagens do paradigma orientado a aspectos (OA) e assim foram formuladas duas hipóteses de solução: alterar um processo de desenvolvimento de LPS, mapeando os requisitos não funcionais e as variabilidades por meio de interesses transversais ou alterar um processo de desenvolvimento OA, buscando representar as variabilidades, similaridades e requisitos não funcionais de uma LPS. Optou-se pela primeira hipótese, pois a revisão de literatura, o estudo de caso exploratório e a experiência do grupo de pesquisa, evidenciaram que seria possível adaptar o processo de gerenciamento de variabilidade (PGV) de LPS proposto por Oliveira Junior (2005) e o método para DBC e aspectos (DBC/A) proposto por Eler (2006), assim unindo os conceitos de LPS e OA.

A concepção da abordagem ProLineA, ocorreu após a assimilação desses conceitos e o desenvolvimento do exemplo da LP-GPS os quais permitiram uma análise mais aprofundada do processo de desenvolvimento de LPS e do gerenciamento de variabilidades que mostrou como e quais seriam as atividades que deveriam ser modificadas e alteradas para considerar aspectos no processo de desenvolvimento.

1.4 Organização

Esta dissertação possui 6 (seis) capítulos e está organizada da seguinte forma: conceitos e abordagens recentes de LPS, DBC e o PGV serão apresentados no Capítulo 2.

Na sequência, no Capítulo 3, serão descritos os conceitos e abordagens existentes sobre o DSOA. No Capítulo 4 é apresentada ProLineA, uma abordagem para o desenvolvimento de Linhas de Produto de Software Orientado a Aspectos.

No capítulo 5 é apresentado um exemplo de utilização da ProLineA, juntamente com as discussões sobre os resultados obtidos. Finalmente, no capítulo 6 são descritos as conclusões e os trabalhos futuros dessa dissertação.

CAPÍTULO

2

Linha de Produto de Software

2.1 Considerações Iniciais

ProLineA toma como base a abordagem de LPS. Assim, neste capítulo são apresentados os principais conceitos e técnicas que tem sido utilizados para desenvolver LPS. Nas seções a seguir serão apresentados alguns conceitos e definições relacionados à LPS (2.1), a relação entre o DBC e LPS (2.2), abordagens existentes de LPS (2.3) e as considerações finais (2.4).

2.2 Conceitos e definições

Uma LPS é um conjunto de sistemas com características comuns e variações bem definidas que visam satisfazer um segmento particular do mercado, normalmente chamado de domínio. O conjunto de sistemas também é conhecido como família de produtos, a Figura 2.1 ilustra uma LPS em que os produtos são representados por p1, p2, .. pn. Como mostra a figura, esses produtos são desenvolvidos de forma sistematizada a partir de um núcleo comum de artefatos (arquitetura genérica) (CLEMENTS,2001; SEI, 2004).

Uma LPS envolve técnicas como *frameworks*, padrões, componentes entre outras em um contexto comum e gerenciável. O desenvolvimento de uma LPS consiste de duas atividades (SEI, 2004; CLEMENTS, 2001): a engenharia do domínio e a engenharia de aplicação. A primeira é relacionada a captura de requisitos do domínio, sua análise e a identificação de um conjunto de artefatos reutilizáveis e configuráveis da LPS. O artefato principal decorrente da análise de domínio é a arquitetura comum aos produtos da LPS. A

engenharia de aplicação consiste no desenvolvimento de um produto da LPS a partir da infraestrutura básica da LPS.

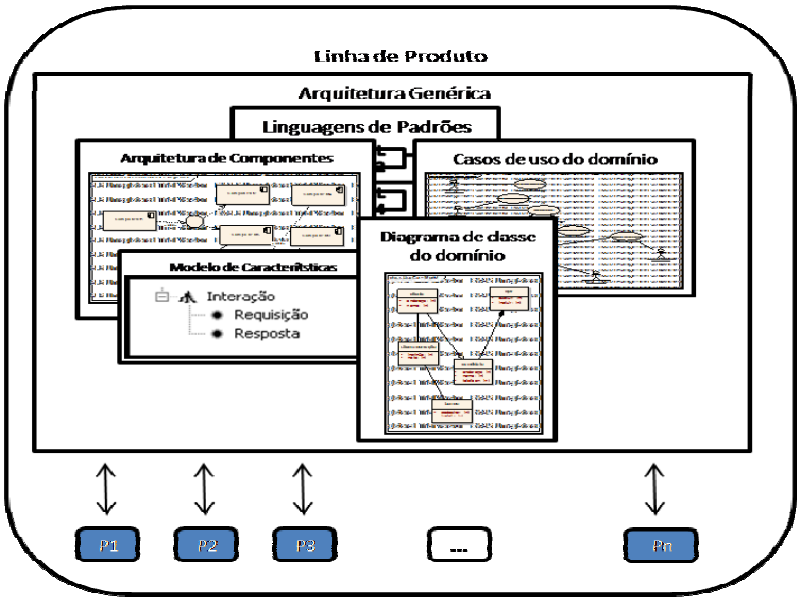


*Figura 2.1 - Representação de uma LPS (*ARAGON*, 2004).*

Na seção seguinte, serão apresentados conceitos e definições relacionados às características variáveis dos produtos gerados por uma LPS.

2.3 Variabilidade

Um dos principais conceitos que devem ser gerenciados em uma LP é o de variabilidade. Ele representa as diferentes características entre produtos de uma mesma família (TRIGAUX, HEYMANS, 2003; OLIVEIRA JUNIOR, 2005).

A principal tarefa no desenvolvimento de uma LPS (NYBEN, 2005) é representar as variabilidades. Por um lado, as variabilidades determinadas durante a fase de análise do domínio têm que ser mapeadas em um projeto arquitetural. Por outro lado, são necessários mecanismos para implementar essas variabilidades em código fonte. Esses mecanismos são necessários para rastrear as variabilidades ao longo das atividades de análise, projeto arquitetural e implementação.

As variabilidades podem ser identificadas pelo conceito de características. Para Van Gurp e Bosch (2001) característica é uma unidade lógica de comportamento relacionados a um conjunto de requisitos funcionais ou não funcionais.

As características podem ser classificadas como (GRISS et al, 1998; VAN GURP;BOSCH, 2001):

- **obrigatórias:** são características que estão sempre presentes e identificam um produto. Por exemplo uma característica efetuar pagamento em um sistema de *e-comerce;*
- **opcionais:** são características que podem ou não estar presentes em um produto. Quando presentes, adicionam algum valor realcionado às características obrigatórias de um produto. Por exemplo, a possibilidade de adicionar um cartão de aniversário na compra;
- **variáveis:** são um conjunto de características semelhantes em que zero ou mais dessas características podem ser selecionadas para estarem presentes em um produto. Por exemplo, a forma de pagamento, podendo ser on-line, cartão, boleto, entre outros;
- **externas:** são as características oferecidas pela plataforma-alvo do sistema. Por exemplo, o tipo de conexão do cliente, ou serviços fornecidos pelo provedor.

Alguns autores apresentam outros tipos de características, Anastasopoulos (2001) apresenta mais dois tipos de características, que são:

- **mutuamente inclusivas**: para que uma característica seja incluída, outras característica específicas devem ser também incluídas e vice-versa;
- **mutuamente exclusivas:** para que uma característica seja incluída, outras características específicas não devem ser incluídas e vice-versa.

2.4 Modelo de Características

As características podem ser estruturadas em modelos de característica. O modelo de características é utilizado no desenvolvimento de uma LPS para especificar e representar as variabilidades e similaridades existentes nos produtos de uma LPS

Existem várias propostas e tipos de representações de característica na literatura, porém todas tomam como base a representação em árvore (SOCHOS; PHILIPPOW; RIEBISCH, 2004).

A Figura 2.2 representa um modelo de características de uma operação de caixa eletrônico que mostra diferentes opções para identificação do usuário e para extrato.

O PGV, utiliza-se um diagrama de classes da UML, em que as *features* são representadas por classes. A relação representada pela notação  , na Figura 2.2 indica que uma característica é formada por outras características. Por exemplo, as características identificação usuário e extrato na Figura 2.2 são formadas, respectivamente, pelas características toque visor/Cartão e visor/impressora.

A notação  , em destaque na Figura 2.2 indica que a característica Impressora é obrigatória.

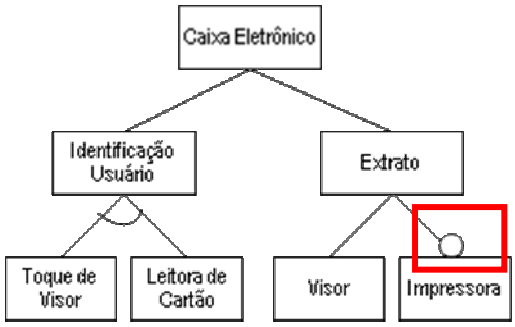


Figura 2.2 - Exemplo de um modelo de características para caixa eletrônico.Adaptado de (LOBO, BRITO; RUBIRA, 2006)

Outro exemplo de modelo de características pode ser visualizado na Figura 2.3, mostrando a representação do processo de inscrição para um concurso vestibular utilizando uma ferramenta denominada *Feature-plugin* (ANTKIEWICZ e CZARNECKI, 2004).

Figura 2.3 - Exemplo de um modelo de características para um processo de inscrição online

O modelo mostra algumas características obrigatórias : `pagamento`, `inscrição`, `provas` e `resultados` e características opcionais como tipo de pagamento, podendo ser: `boleto`, `cartão` ou `depósito`.

Como a ferramenta *Feature Modeling Plugin* não oferece apoio a todas as notações e atividades para o PGV, no contexto deste trabalho será utilizado a notação UML com as extensões propostas por Oliveira Junior (2005).

2.5 Desenvolvimento Baseado em Componentes - DBC

No contexto de LPS, o uso de componentes permite conceber o núcleo de artefatos de modo que um produto pode ser construído a partir de uma arquitetura de componentes na qual os componentes possuem variabilidades a serem configuradas.

O objetivo do DBC é fornecer, ao longo do processo de software, um conjunto de procedimentos, ferramentas e notações que possibilitem tanto a produção de novos componentes quanto a reutilização de componentes existentes. Um componente pode ser definido como “uma unidade de software independente, que encapsula, dentro de si, seu projeto e implementação, e oferece serviços, por meio de interfaces bem definidas, para o meio externo” (GIMENES; HUZITA, 2005). O nível de granulosidade de um componente é

maior do que o de um objeto. Um componente geralmente trabalha com outros componentes para apoiar uma particular ação em nível de negócio (caso de uso) enquanto um objeto geralmente representa uma instância de um conceito do negócio (D'SOUZA; WILLS, 1998).

Um maior detalhamento sobre os conceitos ou os métodos referentes ao DBC pode ser obtido em diversos trabalhos existentes na literatura (HERZUM E SIMS, 2000; BROWN, 2000; HEINEMAN E COUNCILL, 2001; GIMENES E HUZITA, 2005; ELER, 2006 OLIVEIRA JUNIOR, 2005).

2.5.1 Notação

As notações utilizadas para o projeto dos componentes base e transversais (descritas no Capítulo 4) foram adotadas dos conceitos do método *UML Components* (CHESSMAN e DANIELS, 2001) e do trabalho de Eler (2006).

O exemplo apresentado na Figura 2.3 representa um componente que processa uma inscrição on-line. Para diferenciar uma classe de uma interface, os nomes das interfaces começam com I. A entrada dos dados do usuário é representada pela interface `IInscrição` do componente `ComponenteInscrição`. A inscrição só é concluída após verificação positiva do pagamento, através do componente `ComponentePgtoOnline.`

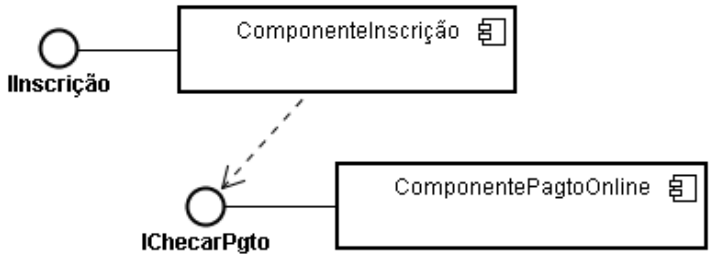


Figura 2.3 - Ilustração do ComponenteInscrição

O componente `ComponentePgtoOnlince` pode ser extendido como ilustrado na Figura 2.4 que possui duas interfaces `IDadosPgtoBanco` e `IDadosInscritos` que manipula as informações do pagamento e do inscrito.

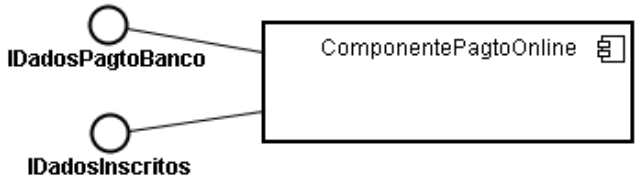


Figura 2.4 - Ilustração do ComponentePgtoOnline

O projeto e a arquitetura dos componentes são técnicas pertencentes ao DBC e são amplamente utilizados para o desenvolvimento de LPS, principalmente no que diz respeito á Engenharia de Domínio (ED) (KANG et al., 1990; SIMOS, 1996; GOMMA, 2005; GRISS et al., 1998).

A seguir serão apresentadas algumas abordagens existentes na literatura relacionados ao desenvolvimento de LPS e ao gerenciamento das variabilidades.

2.6 Abordagens para o Desenvolvimento de Linha de Produto de Software

No processo de desenvolvimento de uma LPS, algumas abordagens se consolidaram no final da década de 90, dentre as quais destaca-se o GenVoca de Batory e O'Malley(1992) que definiram conceitos de uma arquitetura reutilizável bem próximos aos conceitos atuais e que foram redefinidos em Batory et al.(2000) em termos de orientação a objetos com menção à herança parametrizável. Outra abordagem desse período é o Product Line Software Engineering (PuLSETM) (BAYER ET AL., 1999).

Ainda nessa fase histórica, estão o Family-Oriented Abstraction, Specification and Translation (FAST) de Weiss & Lai (1999) e o KobrA Approach de Atkinson et al. (2000), utilizado industrialmente pela Lucent Technologies apresentando resultados satisfatórios enquanto que o segundo foi responsável pela adaptação orientada a objetos do PuLSE e pela distinção entre produtos e processos.

Nas seções 2.3.1 e 2.3.2 serão apresentadas, respectivamente, duas abordagens essenciais para a concepção da ProLineA que são a Practice Product Line (PLP) do SEI (2004) e o PGV proposto por Oliveira Junior (2005).

2.6.1 Practice Product Line (PLP)

Um arcabouço genérico para o processo de desenvolvimento de uma LPS é proposto pela iniciativa *Product Line Pratice* (PLP), desenvolvida pelo SEI (CLEMENTS e NORTHROP, 2001). Assim como o Capability Maturity Model (CMM) (SEI, 2004), o foco da PLP é no uso de práticas de maturidade técnicas, de negócio e de aquisição de tecnologias que conduzem ao encontro de deficiências existentes em projetos de LPS relacionados aos requisitos, arquitetura, modelos, documentações de processos, testes, cronogramas, orçamentos e demais artefatos reutilizáveis.

Neste contexto, o processo de desenvolvimento de uma LPS envolve atividades relacionadas a concepção dos artefatos centrais, ao desenvolvimento do produto e ao gerenciamento da LPS utilizando tais artefatos, como mostra a Figura 2.5.

Figura 2.5 - Atividades essenciais de linha de produto de software. (SEI, 2004)

O SEI (2008), por meio da iniciativa PLP (Product Line Practice), estabeleceu algumas atividades essenciais de uma abordagem de LP. Essas atividades são:

- **Desenvolvimento do Núcleo de Artefatos**: responsável por estabelecer uma infra-estrutura central que é reutilizada pelos produtos gerados a partir da LP.
- **Desenvolvimento do Produto:** responsável pelo desenvolvimento de produtos, membros da família de produtos representados pela LP.
- **Gerenciamento da Linha de Produto:** responsável pelo gerenciamento da LP, garantindo que todas as atividades sejam realizadas de acordo com um

planejamento coordenado. O gerenciamento pode ser dividido em gerenciamento técnico, que coordena as atividades de desenvolvimento e gerenciamento organizacional, que deve garantir que as unidades organizacionais recebam os recursos corretos em quantidades suficientes.

Para executar essas atividades torna-se necessária a definição de áreas de trabalho que representam conjuntos de atividades menores e mais gerenciáveis. Cada área de trabalho possui um plano de trabalho e métricas associadas que permitem acompanhar sua execução e avaliar o sucesso dos trabalhos realizados. Usualmente, as áreas de trabalho permitem produzir artefatos concretos que levem à criação de artefatos centrais utilizados na LPS.

2.6.2 Processo de Gerenciamento de Variabilidades (PGV)

O PGV proposto por Oliveira Junior (2005) define um processo para o gerenciamento de variabilidades de uma família de produtos de software. Este processo tomou como base as atividades de gerenciamento proposto por Van Gurp e Bosch (2001) e a construção de modelos de características proposta por Griss, Favaro e D'Alessando (1998), as atividades definidas pelo processo vão desde a elaboração do modelo de rastreamento de variabilidades até o modelo de implementação conforme mostra a Figura 2.6. As atividades são representados pelas elipses e os artefatos de entrada e saída são representados pelos retângulos.

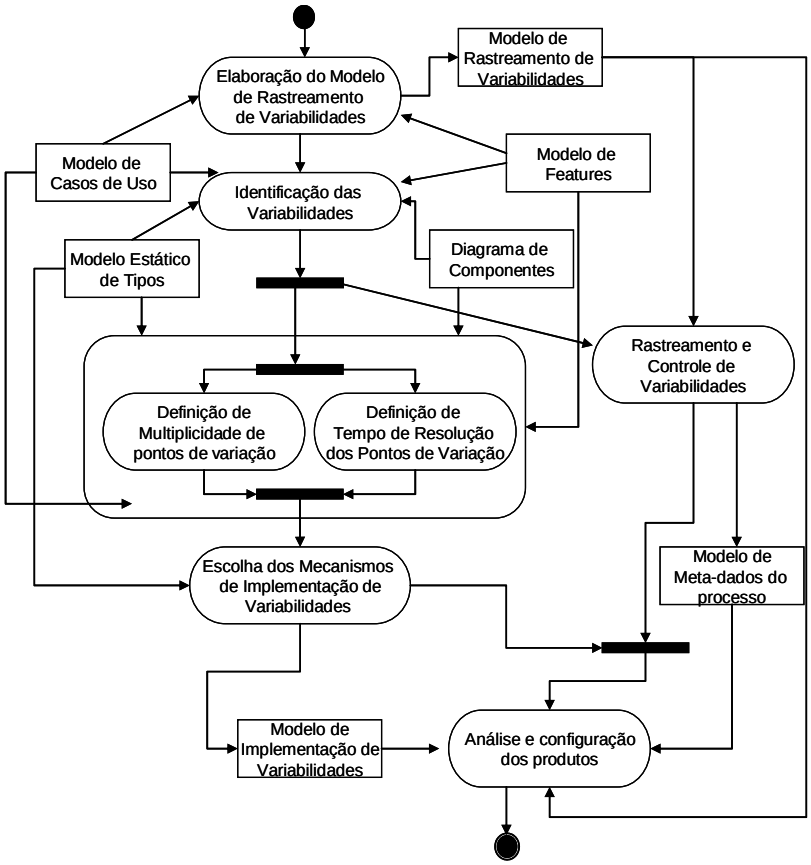


Figura 2.6 - Atividades do processo de gerenciamento de variabilidade para LP. (OLIVEIRA JUNIOR, 2005)

As atividades de desenvolvimento da LP são realizadas pelo engenheiro de LP enquanto que as do gerenciamento de variabilidade são realizadas pelo gerente de LP. A Figura 2.7 mostra a interação entre essas atividades, nota-se que ao final de cada atividade do desenvolvimento da LP é realizada uma iteração do gerenciamento de variabilidade.

As atividades alinhadas verticalmente no retângulo à esquerda correspondem ao ciclo superior esquerdo das atividades propostas pelo SEI (Figura 2.7), enquanto que as atividades do PGV, no retângulo a direita, correspondem ao ciclo inferior.

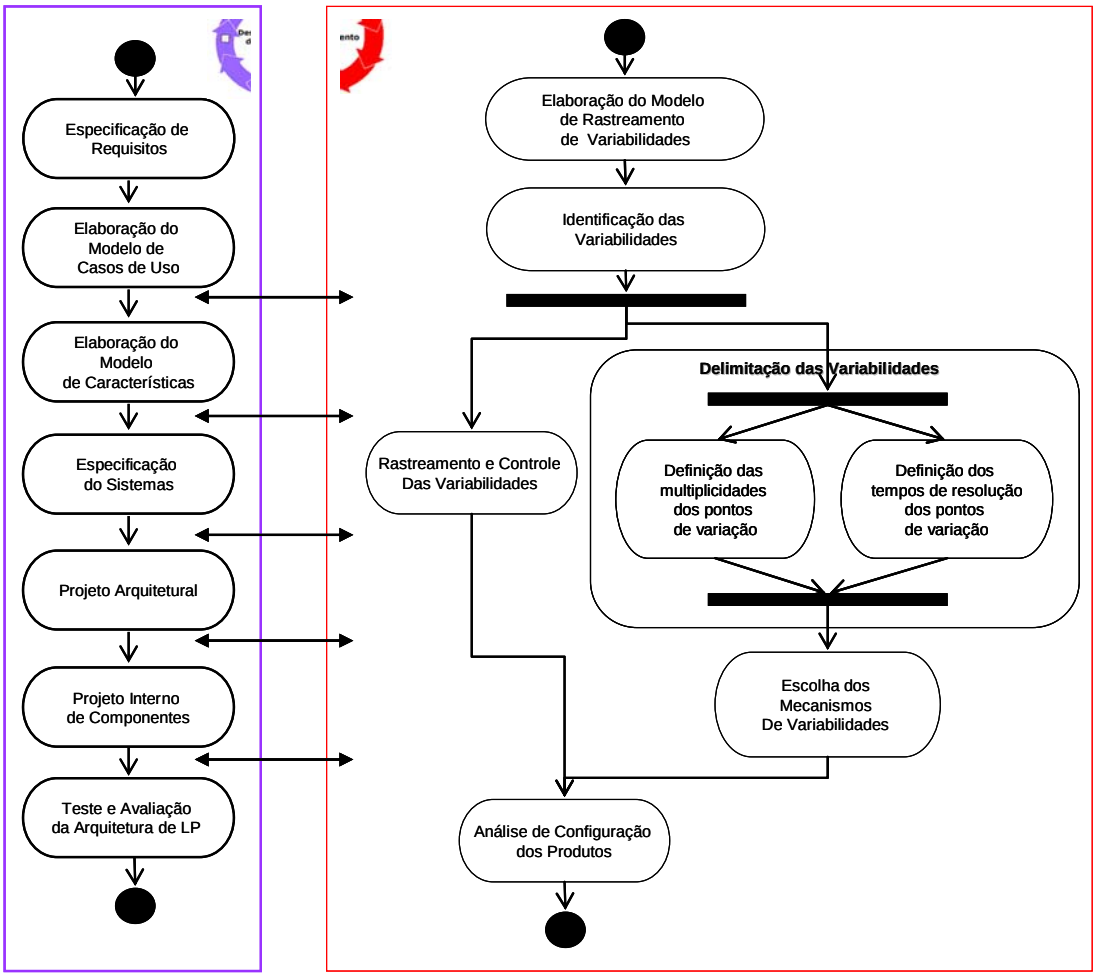


Figura 2.7 - Interação entre as atividades do processo de LPS e as atividades gerenciamento de variabilidades. Adaptado de (OLIVEIRA JUNIOR, 2005)

As atividades do PGV, mostrados nas figura 2.7, são:

Elaboração do Modelo de Rastreamento de Variabilidades: o objetivo desta atividade é criar um modelo de rastreamento de variabilidades entre casos de uso e as características, para permitir o relacionamento entre as variabilidades e os artefatos da LP.

Identificação das Variabilidades: nesta atividade as variabilidades de uma LP serão identificados em diagramas de casos de uso, diagramas de classes e diagramas de componentes. Além disso, essa atividade visa, também, a representação gráfica das variabilidades. Os tipos de características devem ser definidos: obrigatórias (<<mandatory>>), opcionais (<<optional>>), alternativas inclusivas (<<alternative_or>>), alternativas exclusivas (<<alternative_xor>>) e externas (<<external>>), podendo a relação entre elas ser do tipo requires (<<requires>>) e mutex (<<mutex>>). E após as características estarem definidas e identificadas é possível representar graficamente as relações entre elas.

Para isso utiliza-se a notação da UML, em que as características são representadas por classes e relacionadas pelas relações de composição (indica que uma característica é formada por outras características), generalização/especialização (usada exclusivamente para representar as relações entre as características exclusivas) e dependências (utilizada para representar as relações requires e mutex entre características).

Delimitação das Variabilidades: nesta atividade serão definidos a multiplicidade de cada ponto de variação, o tempo de resolução de cada ponto de variação e se os pontos de variação podem aceitar a adição de mais variantes. Os artefatos de entrada para esta atividade são diagramas de casos de uso, diagramas de classes e diagramas de componentes com variabilidades identificadas.

Escolha dos Mecanismos de Implementação de Variabilidade: o objetivo desta atividade é escolher os mecanismos que serão utilizados para implementar as variabilidades de uma LP. Estes mecanismos devem ser definidos no que diz respeito às classes e componentes.

Rastreamento e Controle das Variabilidades: esta atividade visa rastrear e controlar as variabilidades identificadas em uma LP. Para isto, utiliza-se um modelo de meta-dados para o processo. Esse modelo permite que os artefatos de uma LP possam ser relacionados, possibilitando, assim, que se faça o rastreamento e o controle das variabilidades a partir de qualquer artefato da LP que possua variação.

Análise de Configurações de Produtos Específicos: Esta atividade visa à especificação de diferentes produtos de uma LP com o objetivo de verificar se os artefatos da LP necessitam ser modificados ou se são identificadas novas variabilidades. Entende-se por configuração de um produto a especificação de quais características são escolhidas para o produto.

2.7 Considerações Finais

Neste capítulo foram abordados os principais conceitos de LPS e as abordagens utilizadas nesta dissertação. No capítulo seguinte serão apresentados os principais conceitos, técnicas e abordagens relacionadas ao desenvolvimento orientado a aspectos. LPS e aspectos são os princípios fundamentais utilizados para concepção da ProLineA.

CAPÍTULO

3

Desenvolvimento de Software Orientado a Aspectos

3.1 Considerações Iniciais

O desenvolvimento OA é uma área da engenharia de software que envolve estudos aplicados a todo o ciclo de vida de desenvolvimento de um software. O objetivo dessa abordagem é promover a separação de características transversais. As definições e conceitos relacionados ao Desenvolvimento de Software Orientado a Aspectos (DSOA) serão apresentados na seção a seguir (3.2), seguidos de algumas abordagens existentes na literatura de DSOA (3.3). A seção 3.4 apresenta os trabalhos relacionados à Linha de Produtos e Aspectos e a seção 3.5 as considerações finais.

3.2 Conceitos e definições

O DSOA é um paradigma de desenvolvimento de software que provê separação avançada de interesses[1] que surgiu como uma complementação ao paradigma orientado a objetos. Os primeiros resultados na área de aspectos surgiram na fase de implementação (KICZALES, 1997; FILMAN ET AL., 2005). Depois de consolidadas as técnicas para programação como o AspectJ e o JBoss, as pesquisas nas outras áreas começaram a avançar no sentido de apoiar fases anteriores à implementação. Assim, segundo Filman (2005), a abordagem de desenvolvimento orientado a aspectos podem ser claramente identificada em algumas categorias: a programação orientada a aspectos - POA, a modelagem orientada a aspectos – MOA, a engenharia de requisitos orientada a aspectos – AORE e a Análise de requisitos orientada a aspectos – AORA.

Atualmente, a modularização e programação em componentes separados vêm se tornando cada vez mais útil. Porém, durante o desenvolvimento de um software surgem propriedades que não se encaixam em nenhum componente funcional específico. Podemos citar como exemplos o tratamento de exceções e o controle de concorrência, que geralmente se encontram espalhados por várias partes do software. Essas propriedades espalhadas pelo sistema implicam em menor performance ou difícil compreensão do código, complicando assim o reuso de um componente ou até mesmo a manutenção do sistema.

Historicamente, abordagens para separação avançada de características, tais como, programação adaptativa (LIEBERHERR, 1994), filtros de composição (AKSIT,1994) e programação orientada a assuntos (HARRISON, 1993) foram responsáveis pela consolidação da POA.

[1] Do ingles concern.

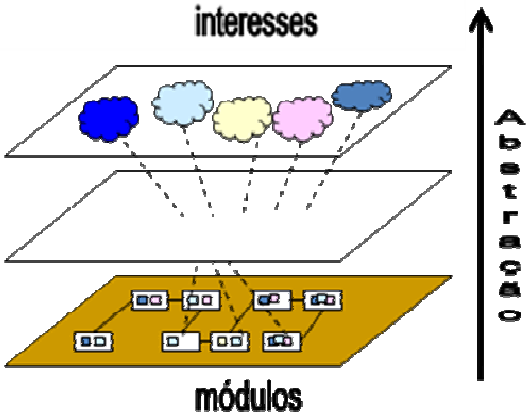


Figura 3.1 - Abstração da separação dos interesses
(CHAVEZ; GARCIA; KULESZA, 2003)

A decomposição e modularidade de sistemas de software complexos devem ser claramente separadas, com interfaces bem definidas, cada uma lidando com um único interesse. Os interesses de um sistema podem ser modularizados por meio de diferentes abstrações fornecidos por linguagens, métodos e ferramentas. Esses interesses podem ser um requisito, uma propriedade ou até mesmo uma funcionalidade de um sistema conforme ilustra a Figura 3.1. Os interesses identificados devem ser encapsulados em módulos coesos e pouco acoplados em todos os estágios do desenvolvimento e sua rastreabilidade deve ser mantida. Ferramentas que apoiam a POA dão suporte ao rastreamentos desses interesses transversais, também denominados, aspectos.

A orientação a aspectos possui quatro conceitos fundamentais (WINC; GOETTEN JR, 2006):

Join Points (os pontos de junção) são pontos bem definidos de um módulo ou software onde um aspecto pode atuar. Um ponto de junção é um conceito chave da POA, uma vez que permite separar partes do código inserindo ou adicionando comandos escritos à parte.

Pointcuts (ponto de corte/interferência) é um conjunto de *join points*. Eles que definem na prática onde o módulo ou software será afetado.

Advices (adendos) capacitam o aspecto a condicionar um comportamento. Eles contêm a implementação ou comportamento adicional a ser inserido no *join point*, especificando quando serão executadas (antes, depois ou durante) em relação ao *join point*.

Inter-type Declarations (declarações intertipos) são os métodos ou atributos que podem ser definidos no aspecto. Esses métodos ou atributos interferem nas classes ou trechos de códigos afetados pelo aspecto.

Aspects (aspectos) são módulos que encapsulam e combinam *pointcuts*, *advices* e *inter-type declarations* relacionados a um mesmo interesse. É como se fossem as "classes" de determinado interesse, tornando a modularização do sistema mais completa.

A POA necessita de um editor de códigos que posteriormente deverá mesclar os códigos desenvolvidos utilizando aspectos ao código alvo, resultando em um único código final (*weaving*).

Uma das linguagens utilizadas para POA mais conhecida atualmente é a AspectJ, que complementa a linguagem orientada a objetos Java. Os aspectos criados a partir do AspectJ podem influenciar nas estruturas estáticas e dinâmicas de um sistema. No caso da estrutura estática, quando adiciona atributos, métodos, modifica a hierarquia das classes, entre outros. No caso da dinâmica, através de *join points* interceptando as mensagens e alterando ou adicionando métodos e comportamentos (KICZALES, 2001). A Figura 3.2 mostra um exemplo de aspecto gerado no AspectJ.

```
public aspect Rastreamento {
    public pointcut metodos();
    execution (**.*(..))&&
    !execution (**.set*(..))&&
    !execution (**.get*(..));

    before():metodos();
    {
    system.out.println("entrou no método:" + thisJoinPoint.toString());
    }

    after():metodos();
    {
    system.out.println("saiu do método:" + thisJoinPoint.toString());
    }

}
```

Figura 3.2 - Exemplo de código de rastreamento de métodos em AspectJ

Na Figura 3.2, o aspecto "Rastreamento" implementa um código de rastreamento de métodos, notificando via impressão na tela a entrada e saída de cada

método. Sem a criação desse aspecto, para se rastrear os métodos executados seria necessário a adição das duas linhas de código em destaque (alterando "thisJoinPoint.toString()" pelo respectivo nome de cada método) em cada método do sistema.

3.3 Abordagens para o Desenvolvimento Orientado a Aspectos

Na literatura encontram-se algumas abordagens e métodos para o desenvolvimento de softwares orientados a aspectos. Nas subseções a seguir serão apresentadas, de forma sucinta, algumas dessas abordagens. A subseção 3.3.1 apresentando uma abordagem de Aspectos aplicada no DBC proposta por Eler(2006), é descrita por ser um dos estudos utilizados como base para a abordagem proposta neste trabalho.

TEMA(CLARKE E BANIASSAD, 2004): propõe o desenvolvimento de software por meio da composição de temas que são unidades que encapsulam os interesses de um sistema e são identificados a partir dos requisitos. Para isso é estabelecida uma relação entre cada requisito e um tema que encapsule este requisito.

Chavez (2004) propôs uma abordagem baseada em modelos com enfoque no projeto orientado a aspectos, denominada ASideML, que apresenta uma linguagem de modelagem que define uma notação gráfica, a semântica associada e regras para a especificação e a comunicação de projetos orientados a aspectos, denominada aSideML.

Desenvolvimento orientado a aspectos com casos de uso : Jacobson e Ng (2004) apresentarem um enfoque de desenvolvimento de software orientado a aspectos com base em casos de uso com o objetivo de identificar interesses transversais a partir dos casos de uso de um sistema e implementá-lo separadamente.

Extensões da UML: Dentre as diferentes abordagens de extensão da UML para modelagem de desenvolvimento de software orientado a aspectos, estão os resultados apresentados por Suzuki e Yamamoto (1999) e Pawlak et. al (2002). O primeiro apresentou a primeira proposta para o projeto orientado a aspectos e utiliza uma notação denominada "aspect" que representa uma nova meta-classe UML. O Segundo, propôs para o projeto de software orientado a aspectos por meio da extensão da UML. São definidos três novos conceitos na UML: grupos , relações de ponto de intersecção (*pointcut relations*) e a metaclasse aspecto (*aspect-classes*) O problema dessa

abordagem, segundo Garcia(2004) é a utilização do papel de um relacionamento para representar os pontos de combinação o que acaba poluindo visualmente o modelo.

3.3.1 Desenvolvimento Baseado em Componente e Aspectos.

Eler (2006) apresenta um método para o desenvolvimento de software orientado a componentes e aspectos, que é uma extensão do método *UML Components* e também utiliza a UML, mas com algumas adaptações. Propõe também uma estratégia para generalização e documentação de componentes transversais que possam ser reutilizados.

Em seu nível mais alto o método DBC/A possue as mesmas etapas do método UML Components: Requisitos, Especificação, Provisionamento, Montagem, Teste e Implantação (Figura 3.3).

O método proposto é uma adaptação do método UML Components com algumas modificações e a inclusão de algumas atividades (Figura 3.3). As principais modificações considerados no método proposto são relacionados a especificação dos requisistos não funcionais desde a análise e especificação dos requisitos, por meio do modelo conceitual de negocio funcional e não funcional e o provisionamento por meio da especificação dos componentes base e transversais.

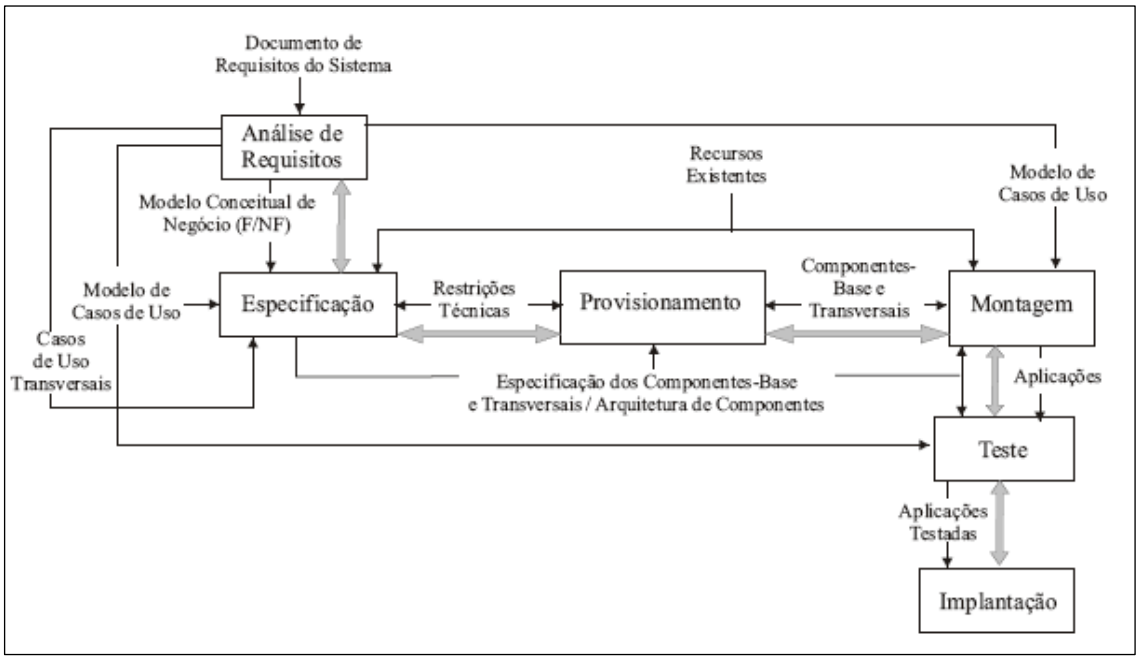


Figura 3.3 - O método UML Components com algumas modificações. (Eler, 2006)

O objetivo do metodo é produzir uma arquitetura de componentes base e transverais a partir do documento de requisitos. Os componentes base possuem

comportamento semelhante a de um objeto dentro de uma aplicação. Já os componentes transversais são componentes que possuem comportamento semelhante a um aspecto, tendo a capacidade de entrecortar outros componentes nas operações de suas interfaces e adicionar ou substituir algum comportamento.

O autor considera interesse transversal como os interesses que afetam várias partes de um sistema, funcionais ou não funcionais,

O DBC/A tem como entrada os requisitos de um sistema, que são usados pela etapa de requisitos para produzir um modelo conceitual de negócios (funcional e não funcional), um modelo de casos de uso (funcional e não funcional) e uma lista de casos de uso transversais. Esses modelos e os recursos existentes no sistema e as restrições técnicas dos projeto, formam as entradas necessárias para que na etapa de especificação sejam produzidos as especificações dos componentes base e transversais. É produzida a partir disso, a arquitetura dos componentes (base e transversais).

3.4 Trabalhos Relacionados à Engenharia de Linha de Produto e Aspectos

O modelo de características tem sido a forma mais utilizada para mapear, identificar e controlar as similaridades e variabilidades no processo de desenvolvimento de uma linha de produto de software. Muitos esforços estão sendo direcionados no sentido de viabilizar a implementação dessas características utilizando-se aspectos.

A utilização de aspectos em várias fases do processo de desenvolvimento de uma linha de produto é um exercício almejado, visto que as vantagens de utilizar aspectos para simplificação do código implementado já é uma realidade para a engenharia de software.

Vários resultados foram obtidos, frutos de pesquisas e esforços no sentido de consolidar a utilização de aspectos no desenvolvimento de linhas de produtos de software.

Alfert (2005) utiliza o modelo de característica para representar as similaridade e variabilidades utilizando aspectos e mostra também que é possível representar, através da utilização do paradigma orientado a aspectos, os requisitos funcionais e não funcionais do sistemas.

Para Alfert(2005) e Nyben (2005) apesar de ser possível utilizar o modelo de características e aspectos, como o nível de abstração do modelo de característica é muito alto em relação a implementação de aspectos e como não existe ainda uma forma de automatizar o modelo de características.

Outras abordagens visam a utilização de ferramentas que combinem a utilização de ferramentas que combinem o modelo de características com a implementação (SVEN, 2006) , ou seja, o modelo de características com o AspectJ por exemplo.

A combinação entre os conceitos do paradigma orientado a aspectos e a engenharia de linha de produto de software é um dos assuntos relacionados a otimização, reutilização e qualidade de software que vem chamando atenção não apenas da comunidade acadêmica mas também sob o ponto de vista empresarial devido à diminuição do tempo de desenvolvimento e a redução de esforços no desenvolvimento de sistemas. Esse fato pode ser evidenciado, por exemplo, com a existência de um *workshop* internacional específico sobre o assunto o *Workshop on Aspect-Oriented Product Line Engineering*.

Pacios (2007) propôs uma abordagem orientada a aspectos para o desenvolvimento de LPS, dando ênfase a abordagem TEMA (CLARKE E BANIASSAD, 2004) utilizada como base para todo trabalho. A abordagem especifica práticas que vão desde a análise à implementação, os aspectos são considerados desde as fases iniciais do projeto, propondo ainda algumas técnicas para implementação de aspectos.

3.5 Considerações Finais

Neste capítulo foram apresentados conceitos relacionados ao desenvolvimento orientado a aspectos através da descrição dos pressupostos básicos, abordagens e ferramentas de desenvolvimento e implementação. O capítulo apresentou também algumas abordagens e estudos relacionados à LPS e DSOA. As tecnologias de desenvolvimento de software orientado a aspectos e também aos componentes transversais já apresentam vários estudos, mas todos ainda em estágio inicial. Alguns métodos apresentados não são apresentados de forma objetiva. As diretrizes desses métodos não são claramente observadas no processo de desenvolvimento.

No item 3.4 nota-se que embora existam trabalhos relacionados a aspectos e LPS, esses estudos ainda não compreendem de forma objetiva e clara esta lacuna. Desta forma, o capítulo a seguir apresenta um processo de desenvolvimento LPS orientada a aspectos.

CAPÍTULO

4

ProLineA: uma Abordagem para o Desenvolvimento de Linha de Produto Orientada a Aspecto

4.1 Considerações Iniciais

Neste capítulo é apresentada a abordagem ProLineA para o desenvolvimento de LPS orientada a aspectos. A abordagem é uma adaptação do processo de gerenciamento de variabilidade desenvolvido por Oliveira Junior (2005) conforme apresentado no Capítulo 2. Além disso, as adaptações propostas pela ProLineA levaram em conta os trabalhos de Eler (2008), Clement at. al (2002), Araujo e Moreira (2004), Jacobson e Ng (2004) e Gomaa (2005).

As principais modificações propostas pela ProLineA envolvem a identificação dos casos de usos transversais, a elaboração do modelo de características considerando com aspectos, o rastreamento mostrando a relação entre os casos de uso transversais e as características e o encapsuladas de características transversais em componentes transversais.

A seção 4.2 apresenta uma caracterização da ProLineA, em linhas gerais, sem aprofundamento na explanação das atividades. Na seção 4.3 serão apresentadas as atividades relacionadas à especificação de requisito e especificação do sistema. O gerenciamento de variabilidades é abordado em duas partes, na primeira parte a seção 4.5 descreve as atividades de elaboração do modleo de variabilidades e a identificação das variabailidades. Em seqüência, na seção 4.6 é descrita as atividades relacionadas a etapa de projeto arquitetural e projeto interno de componentes.

A segunda parte da etapa de gerenciamento de variabilidades é mostrada na seção 4.7 em que são descritas as atividades delimitação das variabilidades e características transversais e a escolha dos mecanismos de implementação.

4.2 Caracterização da ProLineA

O objetivo da abordagem ProLineA é produzir uma infra-estrutura para o desenvolvimento de LPS que a partir da especificação dos requisitos, realize o projeto da arquitetura dos componentes base e dos componentes transversais, considerando as características comuns e as variabilidades de famílias de produtos de software.

No mais alto nível de abstração, a ProLineA é composta pelas mesmas etapas do método de desenvolvimento de linha de proposto por Oliveira Junior (2005). Assim, os estágios do processo de desenvolvimento da LP são: Especificação de Requisitos, Especificação do Sistema, Projeto Arquitetural e Projeto Interno dos Componentes . Além disso, mecanismos de representação de variabilidade foram utilizados com base nas propostas de Jacobson, Griss e Favaro (1997), para diagramas de casos de uso, Jacobson e Ng (2004), e para o diagrama de classes e o projeto interno de componentes, Oliveira Junior (2005) e Eler (2006).

A ProLineA envolve a realização de 5 etapas conforme mostra a Figura 4.1: **especificação dos requisitos e do sistema**, **a elaboração do modelo de características**, **o gerenciamento de variabilidades**, **o projeto arquitetural e o projeto interno de componentes** e **os testes e avaliação da arquitetura**.

Na figura 4.2 observa-se que as estapas de especificação dos requisitos e do sistema, a elaboração do modelo de características, o projeto arquitetural e o projeto interno de componentes e teste e avaliação da arquitetura (retângulo a esquerda) pertecem ao ciclo de desenvolvimento do núcleo de artefatos da proposta PLP do SEI

(Capítulo 2, Figura 2.5). Enquanto que o gerenciamento de variabilidades pertence ao ciclo de gerenciamento (SEI, 2006).

Observa-se também que neste nível de abstração essas etapas são coincidentes com a proposta de Oliveira Junior, no entanto todas essas etapas possuem diversas atividades, como mostradas a seguir nas Figuras 4.3, 4.4, 4.5, que foram modificadas para considerar aspectos em seu desenvolvimento.

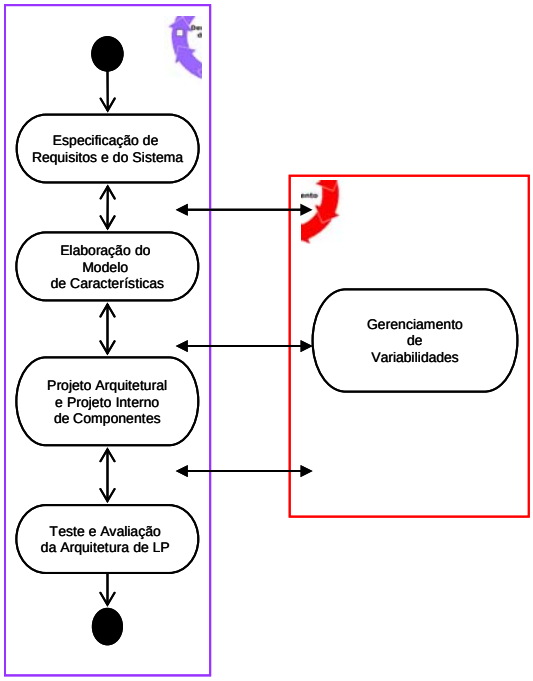


Figura 4.1 – Etapas da ProLineA

As atividades associadas à especificação de requisitos e do sistema, representadas pela Figura 4.2, incluem :

Especificação de requisistos: a identificação dos casos de uso funcionais e não funcionais e dos atores; a construção do diagrama de casos de uso funcionais e do diagrama de casos de uso não funcionais; a integração dos casos de uso não funcionais e os casos de uso funcionais que originarão o modelo de casos de uso do domínio e a identificação dos casos de uso transversais da LPS.

Especificação do sistema: construição do modelo conceitual de negócio e construção dos diagramas de seqüências.

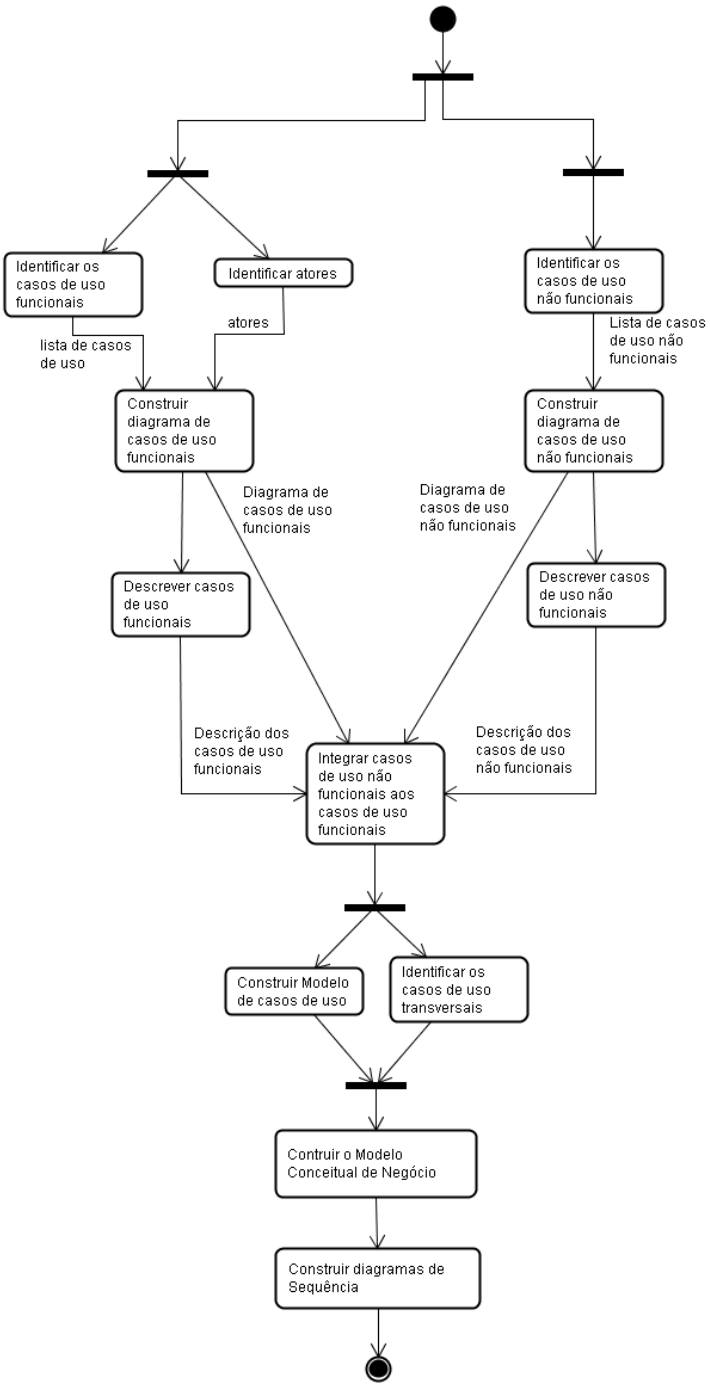


Figura 4.2– Atividades da Especificação dos Requisitos e do Sistema

A Figura 4.3 mostra as atividades do gerenciamento de variabiliades, nota-se que duas atividades sofreram modificações se comparadas com o processo proposto por Oliveira Junior (2005), as atividades: **elaboração do modelo de rastreamento de variabilidade e aspectos** e **identificação das variabilidades e características transversais**. Além disso uma atividade foi incluída, o **rastreamento e controle das características transversais**.

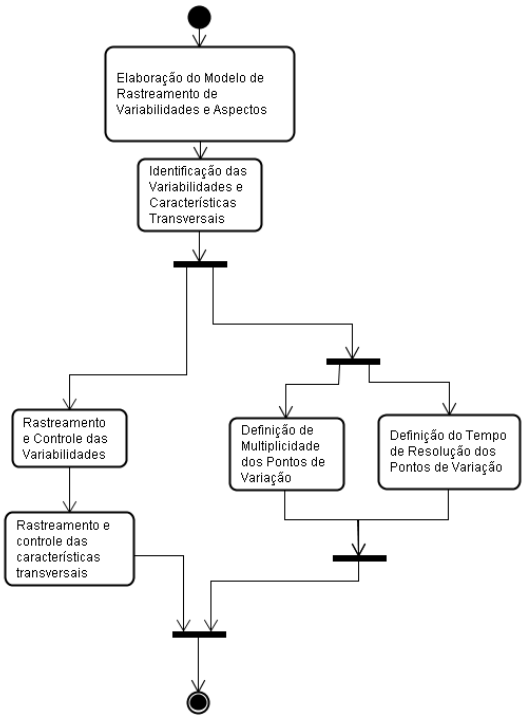


Figura 4.3– Atividades do gerenciamentode variabilidades.

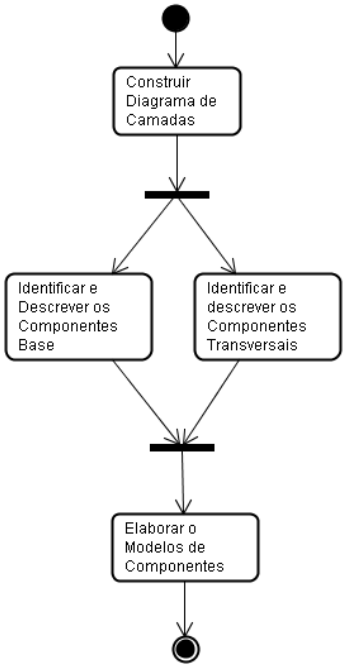


Figura 4.4 - Atividades do Projeto Arquitetural e o projeto Interno de Componentes

A Figura 4.4 mostra as atividades do projeto arquitetural (construir diagrama de classes e o modelo de camadas) e do projeto interno de componentes (identificar e descrever os componentes base e transversais e elaborar o modelo de componentes). Nesta etapa, são identificados tanto componentes regulares, os quais são chamados de componentes base quanto componentes transversais que possuem comportamento semelhantes a um aspecto e podem entrecortar outros componentes.

Nas seções a seguir, serão descritas as atividades envolvidas em cada etapa do processo, suas características, e também os artefatos gerados. A descrição das atividades serão apresentadas de modo seqüencial da análise de requisitos ao teste e avaliação da arquitetura dando a impressão de que o modelo de ciclo de vida utilizado é o cascata. Porém, a ProLineA utiliza o modelo iterativo e incremental, justificadas pela complexidade das práticas para cada etapa do desenvolvimento de LPS. Desta forma, muitos artefatos são modelados e re-modelados em várias iterações no processo de desenvolvimento.

O projeto de uma LPS para o domínio de locação de carros é utilizado para ilustrar a apresentação das atividades considerando da ProLineA. Nessa dissertação esse projeto é denominado Linha de Produto para Sistemas de Locação de Carros (LP SLC)

A principal característica desse domínio é o gerenciamento da locação de carros e o controle de itens opcionais que podem ser adicionados a uma locação, tais como combustível, seguros (danos materiais e pessoais), taxa de retorno, cadeiras de bebê, motoristas e entrega do carro. Os sistemas desse domínio devem possibilitar a emissão de diversos tipos de relatórios e consultas.

A documentação completa do exemplo da LP SLC estão disponíveis para consulta no CD-ROM (em anexo), bem como os diagramas de seqüência (da especificação do sistema) e o modelo de camadas (do projeto arquitetural) que não são mostrados neste capítulo pois não possuem vairibilidades e aspectos representados.

4.3 Especificação dos Requisitos e Especificação do Sistema

O objetivo da etapa de especificação dos requisitos e especificação do sistema é capturar e analisar os requisitos do domínio bem como descrever as funcionalidades dos produtos a serem desenvolvidos. A etapa envolve as atividades de elaboração do

modelo de casos de uso dos requisitos funcionais e do modelo de casos de uso dos requisitos não funcionais do domínio. Para cada uma das atividades podem ser realizadas atividades internas relacionadas aos produtos do domínio da LPS. Os artefatos de saída para esta etapa são o modelo de casos de uso e a descrição dos casos de uso transversais do domínio, conforme mostra a Figura 4.5.

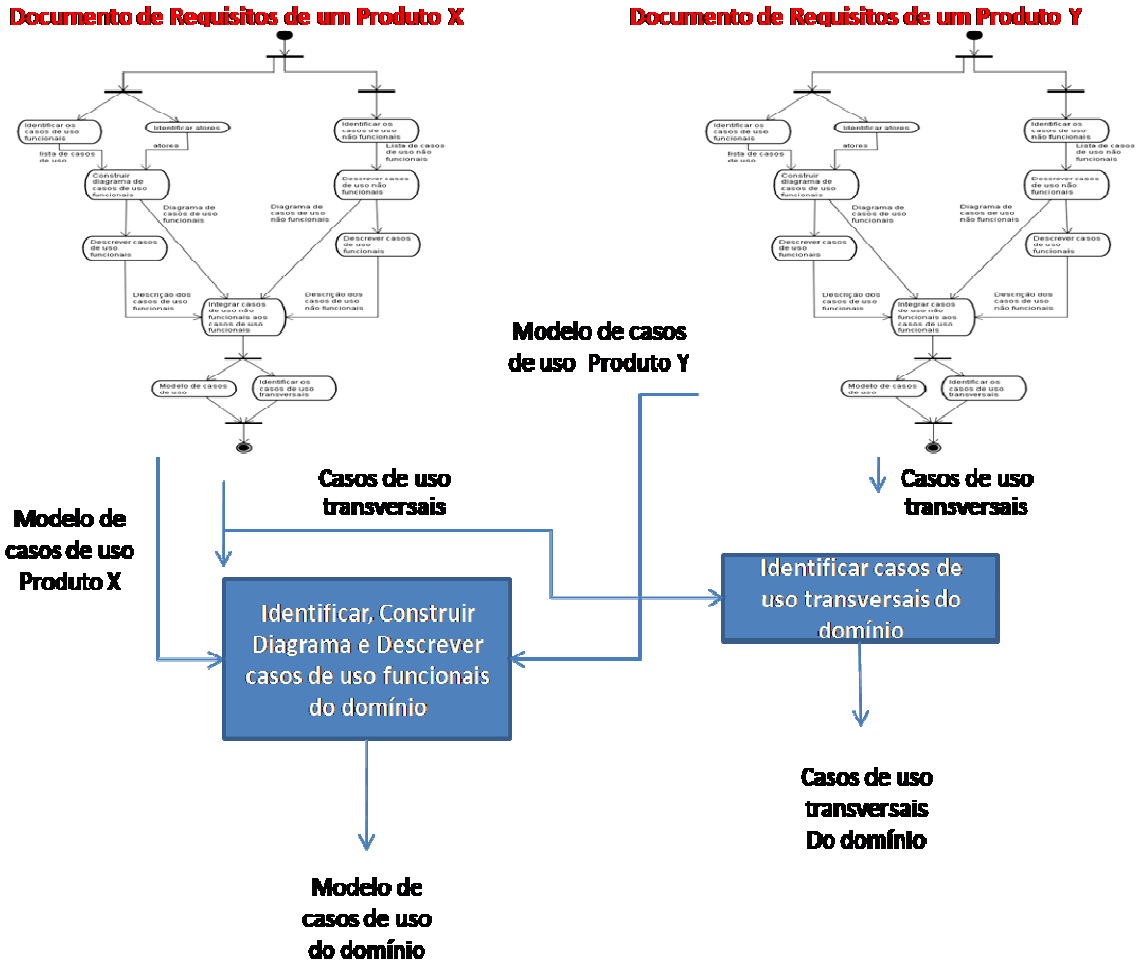


Figura 4.5 - Atividades da Análise de Requisitos de um domínio para LPS.

A seção 4.3.1 apresenta as atividades relacionadas aos casos de uso funcionais e os casos de uso transversais do domínio e na seção 4.3.2 serão apresentados e discutidos os artefatos gerados nesta etapa.

4.3.1 Modelo de Casos de Uso Funcionais

Para descrever os casos de uso funcionais do domínio, a orientação é que pelo menos dois produtos distintos da LPS sejam analisados (ver Figura 4.5), estes produtos devem ter características distintas com a finalidade de detectar as similaridades e

variabilidades. Dessa forma, a descrição dos casos de uso transversais deve considerar também esses fatores.

Os requisitos funcionais do domínio devem ser representados em diagramas de casos de uso, procurando, desde a primeira iteração, identificar os casos de uso comuns e opcionais do domínio. Também deve ser elaborada uma tabela relacionando os requisitos funcionais, os casos de uso comuns da LP e os casos de uso opcionais da LP.

Para a LP-SLC foram encontrados três atores: o `Cliente`**,** que aluga e devolve o carro, e liquida a dívida, quando for o caso; o `Funcionário` que opera o sistema e o `Gerente` para quem os relatórios e consultas são enviados e que determina as operações de cadastros.

Os casos de uso encontrados para a LP-SPC e os requisitos que serão cobertos pelo diagrama de casos de uso são mostrados na Tabela 4.1, incorporada ao modelo. A Figura 4.6 representa o diagrama de casos de uso do domínio. A descrição detalhada dos requisitos podem ser vistos no apêndice A.

Foram identificados os seguintes requisitos para o sistema:

Req.1	Gerenciar o controle e manutenção dos carros. Incluindo operações como incluir, alterar, excluir.
Req.2	Gerenciar Clientes. Incluindo operações como incluir, alterar, excluir.
Req.3	Gerenciar tipos de carros disponíveis.
Req.4	Gerenciar usuários. Incluindo operações como incluir, alterar, excluir.
Req.5	Gerenciar a emissão de relatórios diversos.
Req.6	Emitir notas fiscais dos serviços contratados.
Req.7	Emitir contratos por serviço prestado ou por cliente.
Req.8	Manter atualizações para um veículo, podendo haver mudanças quanto as características do veículo, como a inserção de GPS, ar condicionado, vidros elétricos, entre outros.
Req.9	Registrar multas recebidas pelo cliente.
Req.10	Gerenciar a disponibilização de um veículo.
Req.11	Efetuar pagamento. Incluindo operações como escolha do tipo de pagamento, calculo de desconto

	promocionais, entre outros.
Req.12	Gerenciar categorias de carros
Req.13	Gerenciar devolução dos carros. Verificar multas e as condições do veículo.
Req.14	Gerenciar consultas diversas. Incluindo operações de cruzamento de informações.
Req.15	Gerenciar reservas
Req.16	Gerenciar locação de serviços
Req.17	Gerenciar cancelamentos
Req.18	Gerenciar locação
Req.19	Gerenciar disponibilização de serviços, conforme local, veículo e tempo de locação
Req.20	Gerenciar retirada, incluindo opções de locais de retirada e entrega utilizando serviços de motoristas
Req.21	Gerenciar carrinho
Req.22	Gerenciar reservas serviços
Req.23	Gerenciar reserva tipos
Req.24	Gerenciar devolução
Req.25	Gerenciar consultas
Req.26	Autenticar usuário
Req.27	Persistir dados
Req.28	Gerenciar operações
Req.29	Gerenciar acesso

Para o exemplo da LP-SLC os casos de uso identificados são descritos na Tabela 4.1 que mostra também a relação entre os casos de uso e os requisitos funcionais. A terceira coluna da tabela identifica os casos de uso que serão representados no diagrama de casos de uso (DCU).

Requisito Funcional	Casos de Uso	DCU
Req.1	Incluir/alterar/remover Carros	
Req.2	Incluir/alterar/remover Clientes	
Req.3	Incluir/alterar/remover tipo Carro	
Req.4	Incluir/alterar/remover usuários	
Req.5	GerarRelatório	*
Req.5, Req.6, Req.7	GerarVersão Impressa	*
Req.5, Req.6, Req.7	GerarVersãoOnline	*
Req.6	EmitirNotaFiscal	*
Req.7	EmitirContrato	*
Req.8	ManterCarro	*
Req.9	RegistrarMultas	*
Req.10	DisponibilizarCarros	*
Req.11	PagarConta	*
Req.11	EfetuarPgtoCartão	*
Req.11	EfetuarPgtoBalcão	*
Req.11	EfetuarPgtoOnline	*
Requisito Funcional	**Casos de Uso**	**DCU**
Req.11	EfetuarPgtoBoleto	*
Req.12	ListarCarrosCategorias	
Req.13	DesalocarTipoCarro	*
Req.14	ConsultarTipoCarro	
Req.15	listarReservas	
Req.16	LocarServiço	*
Req.18, Req.19, Req.16	IdentificarReserva	*
Req.18	CancelarReserva	*
Req.19	LocarCarro	*
Req.19, Req.20	VerificarDisponibilidadeCarro	*
Req.20	AlocarTipoCarro	*
Req.20, Req.19, Req.16	CalcularDesconto	*
Req.21	ReitrarCarro	*
Req.21	RetirarCarroLoja	*
Req.21	RetirarCarroOutroLocal	*
Req.22	ReservarTipoCarro	*
Req.23	InserirProdutoCarrinho	*
Req.24	DevolverCarro	*
Req.25	GerarContas	*
Req.24	DevolverCarroLoja	*
Req.24	DevolverCarroLocal	*
Req.25	ObterInformaçõesTipoCarro	
Req.26	ConsultarReserva	

Tabela 4.1 - Requisitos X Casos de uso

Depois de identificados os casos de uso comuns e opcionais e suas respectivas relações com os requisitos funcionais, é necessário descrever cada caso de uso, conforme o modelo representado na tabela 4.2 que mostra a descrição do do caso de uso `efetuarPgtoOnline`.

Caso de uso: efetuarPgtoOnline
Tipo: opcional
Objetivo: identificar pagamento efetuado online
Casos de uso: liquidar dívidas
Fluxo obrigatório
1. Cliente loca o carro 2. Cliente devolve o carro 3. Cliente liquida dívidas 4. Cliente efetua pagamento online 5. Sistema registra pagamento 6. Sistema emite recibo
Fluxo alternativo
1. Cliente loca o carro 2. Cliente devolve o carro 3. Cliente liquida dívidas 4. Cliente efetua pagamento Boleto 5. Sistema ativa outro caso de uso

Tabela 4.2 - Descrição do caso de uso efetuarPgtoOnline.

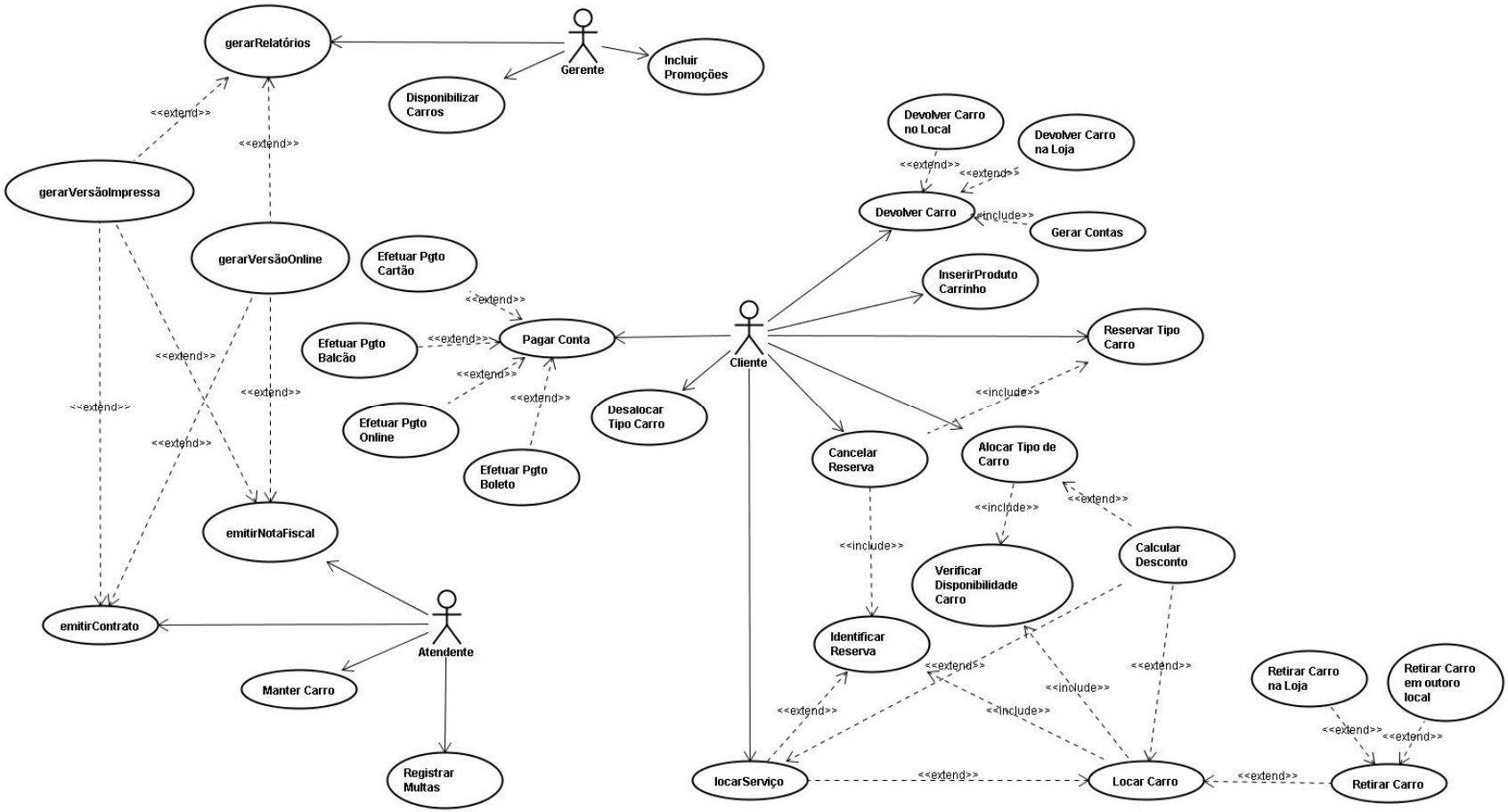


Figura 4.6 -Diagrama de casos de uso da LP-SLC.

4.3.2 Modelo de casos de uso transversais do domínio.

A identificação dos casos de uso transversais começa com a atividade de elaboração do diagrama de casos de uso com os requisitos não funcionais.

A Tabela 4.3 mostra a relação entre os requisitos não funcionais e os casos de uso, o asterisco na terceira coluna da tabela identifica os casos de uso mostrados no diagrama de casos de uso.

Requisito Funcional	**Casos de Uso Comuns da LP**	**D.C.U**
Req.04	gerenciarUsuários	
Req.26	autenticarUsuários	
Req.27	persistirDados	*
Req.28	registrarOperações	*
Req. 29	controlarAcesso	

Tabela 4.3 - Casos de uso não funcionais da LP-SLC

A integração dos casos de uso funcionais com os casos de uso não funcionais em um único diagrama de casos de uso pode gerar dificuldades de legibilidade, dessa forma é recomendado, como proposto por Eler (2005), que sejam criadas visões separadas para cada caso de uso não funcional, ou para pequenos conjuntos de casos de uso relacionados a um mesmo requisito coberto, ou a uma mesma funcionalidade. Nada impede que alguns casos de uso funcionais, os mais relevantes, sejam identificados juntamente com o diagrama de casos de uso não funcionais, como é o caso do caso de uso `calcularDesconto` na Figura 4.6.

A Figura 4.7 mostra um diagrama parcial de casos de uso, ilustrando também alguns casos de uso não funcionais, nota-se na ilustração que o requisito não funcional `registraOperações` é um caso de uso não funcional e é identificado pelo esteriótipo `<<RNF>>`. A relação entre os casos de uso funcionais e os casos de uso não funcionais (JACOBSON NG, 2004) devem ser `<<extend>>`.

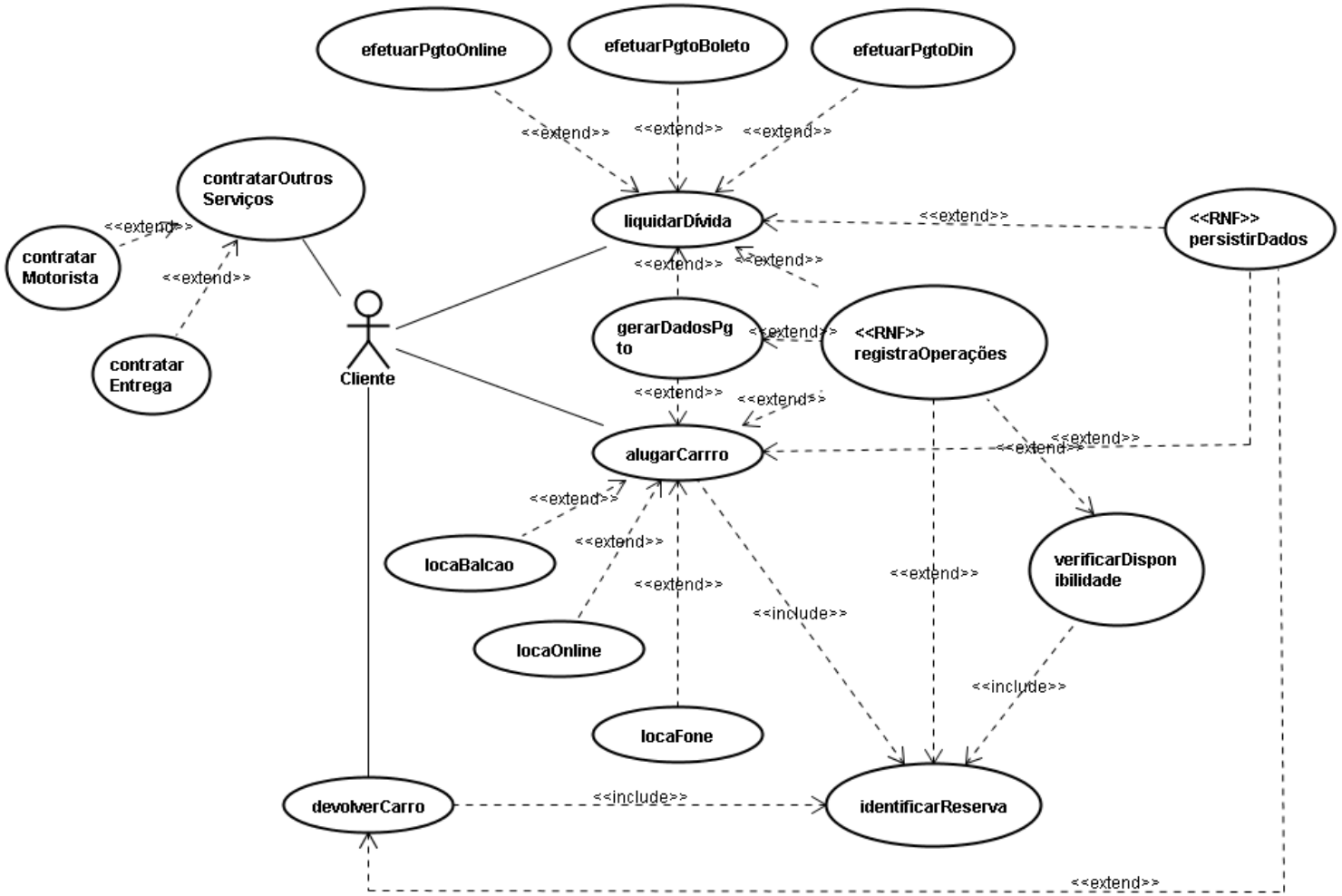


Figura 4.7- Diagrama parcial dos casos de uso dos requisitos não funcionais

4.3.3 Modelo Conceitual de Negócio Funcional e Não Funcional (MCN/F e MCN/NF)

O objetivo do modelo conceitual de negócio (MCN) é representar os conceitos mais importantes do sistema em classes. Este modelo é criado a partir do modelo de casos de uso funcionais. Uma vez que o MCN não funcional será utilizado em momento distinto na abordagem, pode ser construído em uma iteração distinta e posterior no processo de desenvolvimento, em uma atividade intermediária entre as etapas de especificação dos reusisitos e do sistema e projeto arquitetural.

O MCN/F é criado na primeira iteração do processo, mas assim que as variabilidades forem definidas, na próxima iteração deve ser remodelado para representação das variabilidades e pontos de variação. O MPN/F para a LP-SLC pode ser visto na Figura 20.

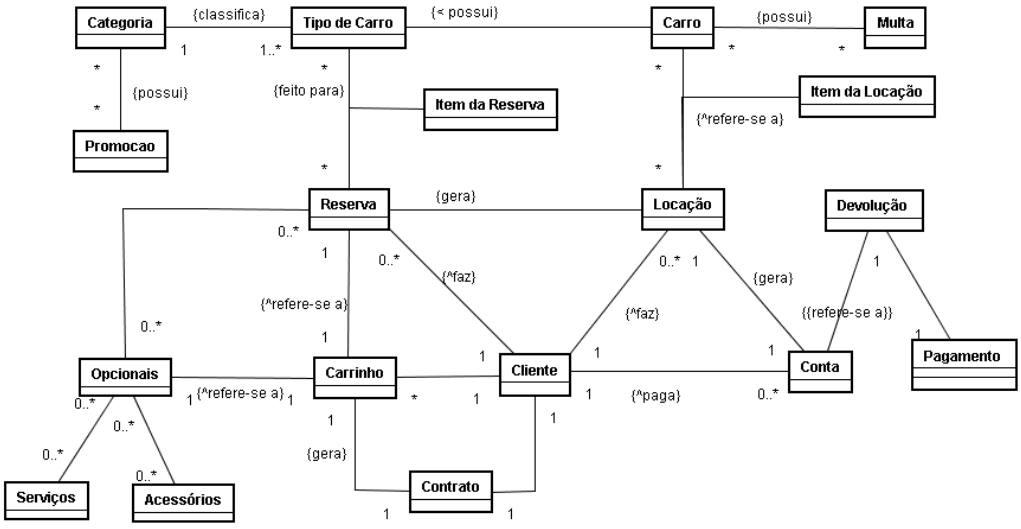


Figura 4.8 – Modelo Conceitual de Negócio Funcional

A criação do MCN/F é recomendada pela UML Components, já a criação do MCN/NF é uma recomendação de Eler (2006) para auxiliar na compreensão e possível implementação dos aspectos transversais. A incorporação desta atividade pela ProLineA é justificada por esses mesmos princípios.

A Figura 4.9, mostra o MCN/NF obtido para a LP, em que os requisitos não funcionais 26, `registrarOperações` gera os conceitos `Carro`, `Pagamento` e

`Papel`, e o requisito não funcional 24, `autenticarUsuário` gera os conceitos `Papel`, `Usuário` e `Operação`. Recomenda-se a criação de um MCN para cada subconjunto de requisitos não funcionais que possuem relações entre si.

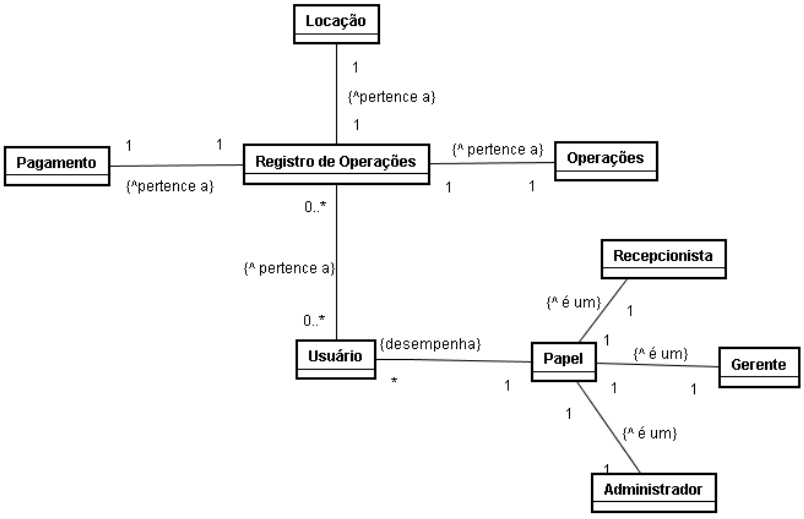


Figura 4.9 – Modelo Conceitual de Negócio Não-Funcional

4.3.4 Identificação dos Casos de Uso Transversais

Os casos de uso transversais são aqueles que implementam aspectos transversais e que provavelmente possuem relação com mais de um caso de uso do sistema. Os critérios recomendados por Eler (2006) são utilizados para identificar os possíveis candidatos a aspectos. Os casos de uso transversais são aqueles que possuem relacionamento (extend ou include) por isso são casos de uso transversais:

- os casos de uso incluídos por mais de um caso de uso;
- os casos de uso que estendem por mais de um caso de uso;
- os casos de uso que restringem mais de um caso de uso.

Os casos de uso transversais encontrados para a LP/SLC são apresentados na Tabela 4.4.

Requisito	Casos de Uso	Tipo
Req.21	calcularDesconto	Funcional
Req.5	locarCarro	Funcional
Req.24	autenticarUsuários	não-funcional
Req.25	persistirDados	não-funcional
Req.26	registrarOperações	não-funcional
Req.27	controlarAcesso	não-funcional

Tabela 4.4 – Identificação dos Casos de Uso Transversais

A recomendação da ProLineA é que neste momento, como uma nova iteração do processo, os diagramas de casos de uso funcionais e não funcionais sejam re-modelados para representar os possíveis aspectos identificados. Os casos de uso identificados como casos de uso transversais devem ser identificados com o esteriótipo <<aspect>>. A Figura 4.10 mostra o diagrama de casos de uso funcionais com os possíveis aspectos (em destaque).

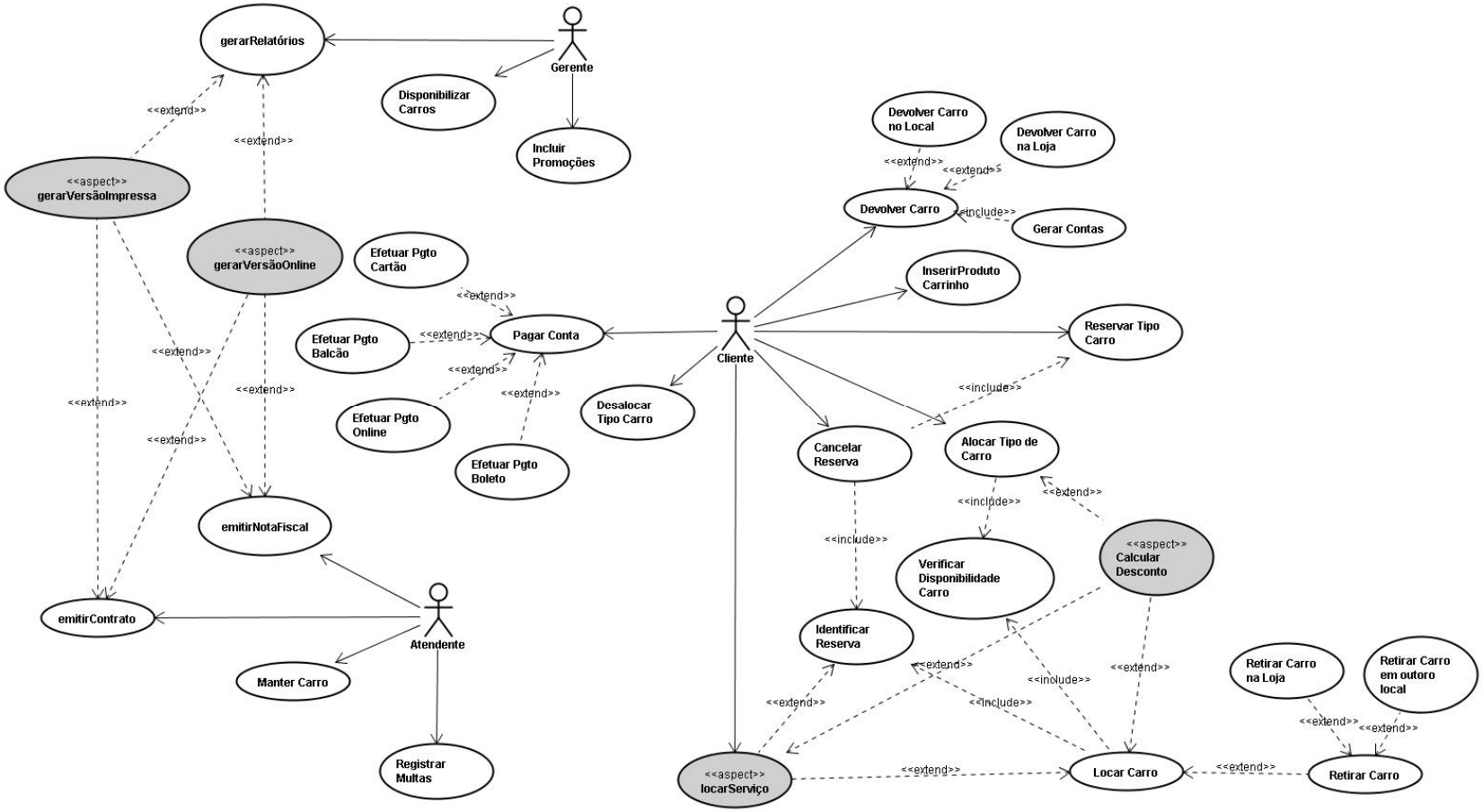


Figura 4.10 – Diagrama de Casos de Uso Funcionais para a LP/SLC com possíveis aspectos identificados.

4.4 Elaboração do Modelo de Características

A elaboração do modelo de características segue, de uma forma geral, as mesmas etapas sugeridas por Oliveira Junior (2005), a diferença é que em vez de utilizar como artefato de entrada apenas o modelo de casos de uso, é usado também o modelo de casos de uso transversais (em ambos os casos os possíveis aspectos já estão identificados) e o MCN.

Nesta etapa será possível identificar quais os casos de uso transversais que constituem também uma característica variável do sistema.. A decisão sobre como os casos de uso serão implementados será tomada posteriormente juntamente com a decisão dos mecanismos de resolução e implementação das variabilidades.

As ações a serem realizadas para a elaboração do modelo de características são:

- identificação das características;
- definição dos tipos das características;
- representação gráfica das características.

4.4.1 Identificação das características

A identificação das características de uma LPS é feita a partir dos requisitos representados no modelo de casos de uso e dos conceitos representados no MCN. Esta identificação depende do conhecimento que o gerente de LPS possui do domínio a ser modelado, mas os procedimentos básicos são: (i) identificar características em grupos de casos de uso funcionais e não funcionais; (ii) identificar características em grupos de conceitos funcionais e não funcionais; e (iii) criar características para cada grupo identificado. Esses procedimentos são descritos a seguir.

(i) Identificar características em grupos de casos de uso funcionais e não funcionais.: devem ser analisados os casos de uso e os atores que tiverem algum relacionamento em comum e que possam representar pontos de variação. Por exemplo: no diagrama de casos de uso funcionais os casos de uso `DevolverCarroLocal`, `DevolverCarroLoja` podem ser agrupados pois representam as opções de tipo de devolução para o caso de uso `DevolverCarro`.

(ii) Identificar características em grupos de conceitos funcionais e não funcionais: para identificar as características em grupos de conceitos funcionais e não funcionais deve ser analisado o MCN, pois grupos de conceitos opcionais também

podem representar uma característica opcional. Por exemplo: no MCN/F os conceitos `Serviços` e `Acessórios` podem ser agrupados pois representam as opções de tipo de serviços ou acessórios opcionais que o cliente pode optar quando alugar um carro. Assim são características opcionais.

(iii) Criar características para cada grupo identificado: para cada grupo de características identificadas devem ser criados uma ou mais características. Por exemplo: para o grupo de casos uso `devolverCarro`, `devolverCarroNo Local`, `devolverCarroNaLoja`, poderia ser gerada a característica `Devolução` que pode ser composta pelas características: `Local` e `Loja`.

4.4.2 Definição do Tipo e das Relações das Características

Nesta etapa as características deverão ser classificadas pelo tipo e também devem ser definidas as relações entre as características. Segundo Oliveira Junior (2005) as características podem ser classificadas em: obrigatórias, opcionais, externas, alternativas inclusivas e alternativas exclusivas. Como as características são identificadas a partir do modelo de casos de uso, a ProLineA inclui um novo tipo de característica denominada "transversal" , relacionado aos casos de uso transversais e representado no diagrama de casos de uso com o estereótipo <<aspect>>.

4.4.3 Representação Gráfica das Características

A representação gráfica do modelo de características utiliza um diagrama de classes da UML, como proposto por Clauß(2001) e utilizado por Oliveira(2005).A Tabela 4.5 apresenta um sumário dos estereótipos utilizados para identificação dos tipos e dos relacionamentos entre as características.

Tipo	**Estereótipo**
Obrigatórias	<<mandatory>>
Opcionais	<<optional>>
Externas	<<external>>
Alternativas inclusivas	<<alternative_OR>>
Alternativas exclusivas	<<alternative_XOR>>
Transversal	<<aspect>>
Relação	**Esteriótipo**
Requires	<<requires>>
Mutex	<<mutex>>

Tabela 4.5 – Estereótipos adotados na representação gráfica dos tipos e das relações entre as características.

O tipo da característica é definida como **transversal** quando ela estiver relacionada com mais de uma característica, podendo ser ainda **alternativa inclusiva** ou **exclusiva**. No modelo de característica da Figura 4.11, a característica `Serviço` é transversal, pois ela é uma característica alternativa inclusiva das características `Locação` ou de `Opcionais.`Quando uma característica for do tipo transversal, a definição da relação *mutex* ou *requires* deve ser feita entre os pontos de variação, uma vez que ela pode ser *mutex* para uma característica e *requires* para outra.

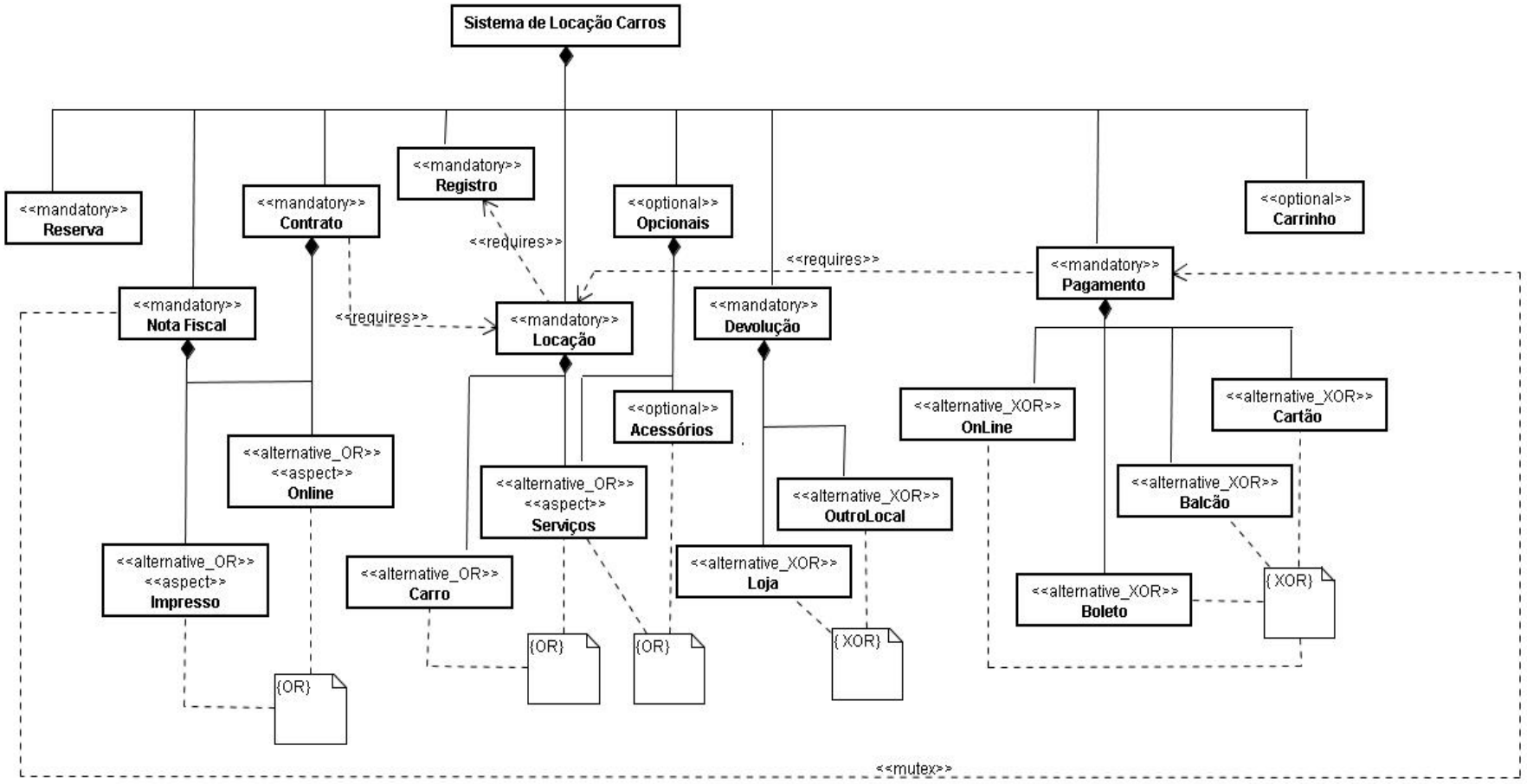


Figura 4.11 – Modelo de Características para a LP – SLC.

4.5 Gerenciamento de Variabilidades: Elaboração do Modelo e Identificação de variabilidades e caracteríticas transversais.

Algumas atividades da etapa de gerenciamento de variabilidades proposta por Oliveira Júnior (2005) foram adaptadas e outras foram incluídas para apoiar a modelagem dos aspectos transversais. Os artefatos de entrada para esta primeira parte da etapa de PGV são o modelo de casos de uso, a descrição dos casos de uso transversais, o modelo de características e o modelo conceitual de negócios, envolvendo as seguintes atividades: 1) Elaboração do modelo de Rastreamento de características variáveis e características transversais e 2) Identificação das variabilidades e características transversais.

Os artefatos de saída para esta atividade são a evolução, com os pontos de variação, multiplicidades, delimitações e características transversais identificadas, dos seguintes artefatos: o modelo de casos de uso do domínio, a evolução do modelo conceitual de negócios em diagrama de classes e o modelo de características.

Todas as atividades sofreram alterações para considerar aspectos em suas ações. Nas subsecções a seguir serão descritos cada uma dessas atividades, mostrando as adaptações, alterações e as atividades internas incluídas no processo.

4.5.1 Elaboração do Modelo de Rastreamento de características variáveis e características transversais

As características transversais podem ter relação com os casos de uso transversais ou não dependendo do requisito que o originou, pois as características são abstrações desses requisitos possibilitando então que a essência transversal desses requisitos (representados nessa abordagem pelos diagramas de casos de uso) seja preservada no modelo de características.

A ProLineA propõe a inserção de novos tipos de características e novas notações para a representação gráfica das características nas atividades propostas por Oliveira Junior(2006).

As Tabelas 4.6 e 4.7 mostram respectivamente o modelo de rastreamento de variabilidades e o modelo de rastreamento de características transversais. A Tabela 4.8 mostra os relacionamentos existentes entre os casos de uso e as características. A Tabela

4.9 identifica os relacionamentos entre as características, mostrando quais características possuem um relacionamento com outras características.

Para construir o modelo de rastreamento de variabilidades, deve-se proceder da seguinte maneira:

1. listar todas as características da LP;

2. listar todos os casos de uso da LP;

3. para cada característica listada, identificar os casos de uso que originam tal característica;

4. marcar com o símbolo * o cruzamento entre as característica e os casos de uso relacionados.

característica **casos de uso**	**...**	**Locação**	**Serviço**	**Carro**	**Devolução**	**Loja**	**Outro Local**	**...**	**Pagto**	**OnLine**	**Balcão**	**Cartão**	**Boleto**
...	...	...	...	...	...	...	...	...	...	...	...	...	...
locarCarro		*		*									
gerarDadosPgto													
devolverCarro					*	*	*						
devolver CarroLoja						*							
devolverCarroLocal							*				*		
cadastrarCarro													
gerarRelatórios													
liquidarDivida									*	*	*	*	*
cadastrarCliente													
efetuarPagamento									*	*	*	*	*
efetuarPgtoOnline											*		
efetuarPgtoCartao												*	
...	...	...	...	...	...	...	...	...	...	...	...	...	...
efetuar PgtoBalcao											*		
efetuarPgtoBoleto													*
identificarLocatário													
contratarServiços		*	*										
contratarMotorista			*										
contratarEntrega			*										
contratarDevolução					*	*	*						
calcularDesconto													
retirarCarroOutroLocal													

Tabela 4.6 – Modelo Parcial de Rastreamento de Variabilidades

característica **característica**		**Nota Fiscal**	**Impresso**	**Online**	**Contrato**	**Registro**	**Locação**	**Serviço**		**Devolução**	**Loja**	**Outro Local**	**Opcionais**
Reserva	...								...				
Nota Fiscal	...	*							...				
Impresso	...	*	*		*				...				
Online		*		*	*								
Contrato					*								
Registro						*							
Locação							*						
Serviço							*	*					*
Carro													
Devolução										*			
Loja											*		
Outro Local												*	
Acessórios													
Opcionais													*
Pagamento													
OnLine													
Balcão													
Cartão													
Boleto													
Carrinho													

Tabela 4.7 – Modelo Parcial de Rastreamento de Variabilidades (Características X Características)

Exemplos de características transversais que podem ser vistos na Tabela 4.7, são: `Impresso`, `OnLine` e `Serviço`.

4.5.2 Identificação das Variabilidades e das Características Transversais.

Os artefatos de entrada para a atividade de identificação das variabilidades e das características transversais são o modelo de casos de uso, a descrição dos casos de uso transversais e o MCN. O objetivo é identificar as variabilidades e as características transversais de uma LPS em diagramas de casos de uso, diagrama de classes e diagramas de componentes. Para atingir esse objetivo são necessários os seguintes passos: 1) a identificação dos pontos de variação, variantes, e seus relacionamentos e a 2) representação gráfica das variabilidades.

4.5.2.1 Identificação de Pontos de Variação, Variantes e de seus Relacionamento.

A identificação dos pontos de variação e variabilidades nos diagrama de casos de uso e diagrama de classes é semelhante ao proposto por Oliveira(2005) com a inclusão das características transversais. No diagrama de casos de uso as características transversais são identificadas através de uma relação `<<extend>>` e com a utilização do estereótipo `<<aspect>>`. Como mostrado na Tabela 4.10, da seção seguinte.

4.5.2.2 Elaboração do Diagrama de Classes e a representação gráfica das variabilidades;

A Figura 4.10 mostra a representação gráfica das variabilidades no diagrama de casos de uso, segundo notação e estereótipos recomendados por Oliveira Junior (2005), descritos no capítulo 2, com exceção da representação das características transversais. A representação gráfica das variabilidades exige que os diagramas de caso de uso sejam remodelados, que o diagrama de classes seja evoluído a partir do MCN e que o diagrama de componentes seja projetado. A Tabela 4.8 mostra a notação utilizada para representação em diagramas de casos de uso, com a inclusão das características transversais.

	Diagramas de Casos de Uso	
	Relação da UML	**Estereótipo do Elemento**
Ponto de Variação	---------	<<variationPoint>>
Variante Obrigatória	Associação, agregação, composição ou dependência	<<mandatory>>
Variante Opcional	Associação, agregação, composição ou dependência	<<optional>>
Variante Alternativa Inclusiva	Especialização/Generalização com o estereótipo <<extend>>	<<alternative_OR>>
Variante Alternativa Exclusiva	Especialização/Generalização com o estereótipo <<extend>>	<<altenative_XOR>>
Variante Mutuamente Exclusiva	Dependência	<<mutex>>
Variante Inclusiva	Dependência	<<requires>>
Variantes Transversais	Especialização/Generalização com <<extend>> ou <include>>	<<aspect>>

Tabela 4.8 – Relação e estereótipos usados na representação gráfica das variabilidades em diagramas de casos de uso.

Oliveira Junior (2005) propôs a utilização das notas da UML para representar informações adicionais sob os pontos de variação. A ProLineA recomenda que as notas sejam utilizadas obrigatoriamente para representar informações adicionais para as variantes transversais. Por exemplo, na Figura 4.12, as variantes `GerarVersãoImpressa` e `GerarVersãoOnLine` utilizam duas notas da UML pois como elas são variantes transversais pertencentes a dois pontos de variação elas possuem informações diferentes, podendo ter uma relação alternativa inclusiva em um ponto de variação e uma relação alternativa exclusiva em outro ponto de variação, como no exemplo citado. A representação dos pontos de variação e variantes no diagrama de classes segue a Tabela 4.9, com a inserção das variantes transversais.

	Diagramas de Classes	
	Relação da UML	**Estereótipo do Elemento**
Ponto de Variação	---------	<<variationPoint>>
Variante Obrigatória	Associação, agregação, composição ou dependência	<<mandatory>>
Variante Opcional	Associação, agregação, composição ou dependência	<<optional>>
Variante Alternativa Inclusiva	Herança	<<alternative_OR>>
Variante Alternativa Exclusiva	Herança	<<altenative_XOR>>
Variante Mutuamente Exclusiva	Dependência	<<mutex>>
Variante Inclusiva	Dependência	<<requires>>
Variantes Transversais	Especialização/Generalização com <<extend>> ou <<include>>	<<aspect>>

Tabela 4.9 – Relação e estereótipos usados na representação gráfica das variabilidades em diagrama de classes

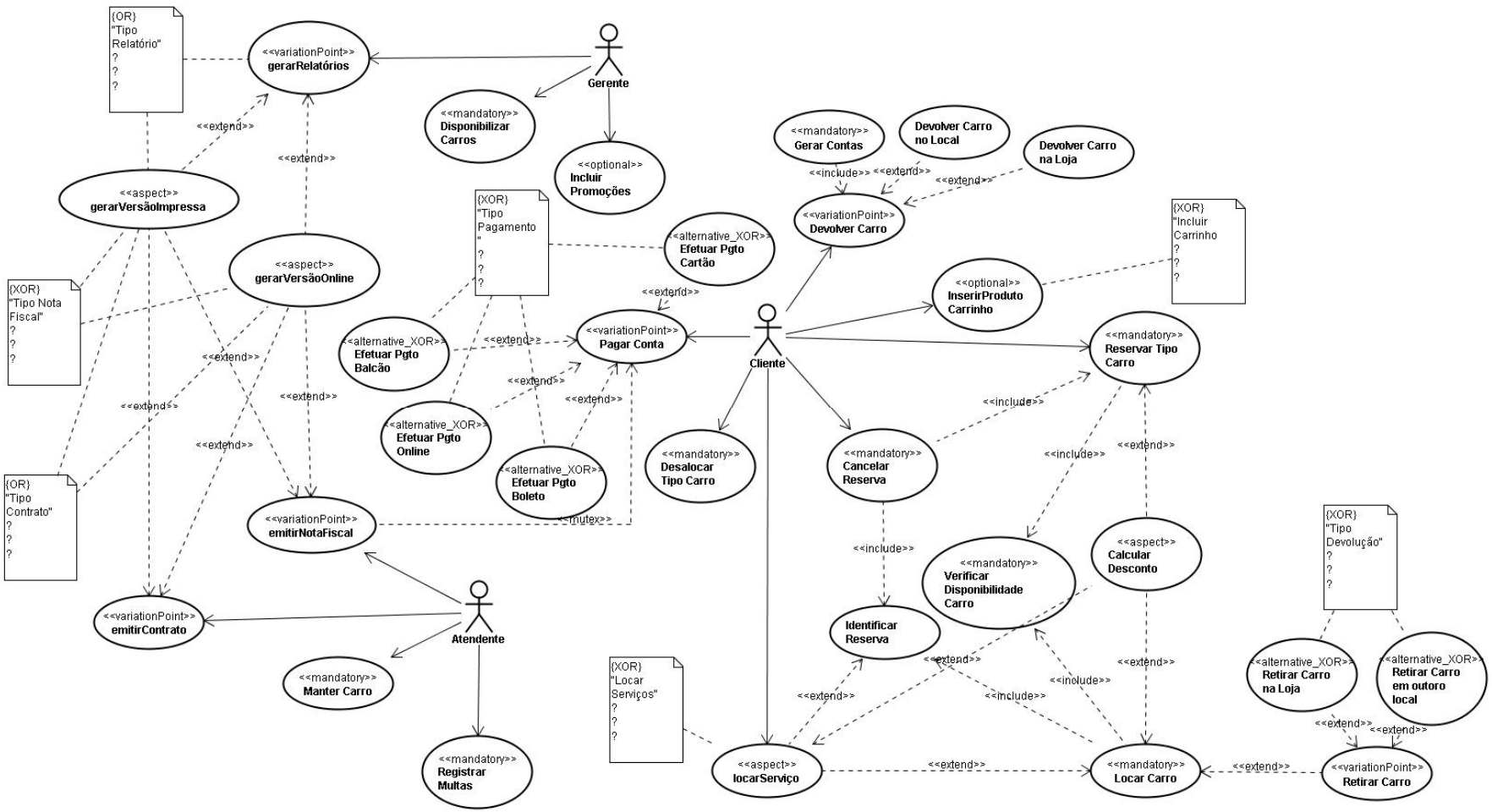


Figura 4.12 – Identificação dos Pontos de Variação e Variantes em Modelo de Casos de Uso

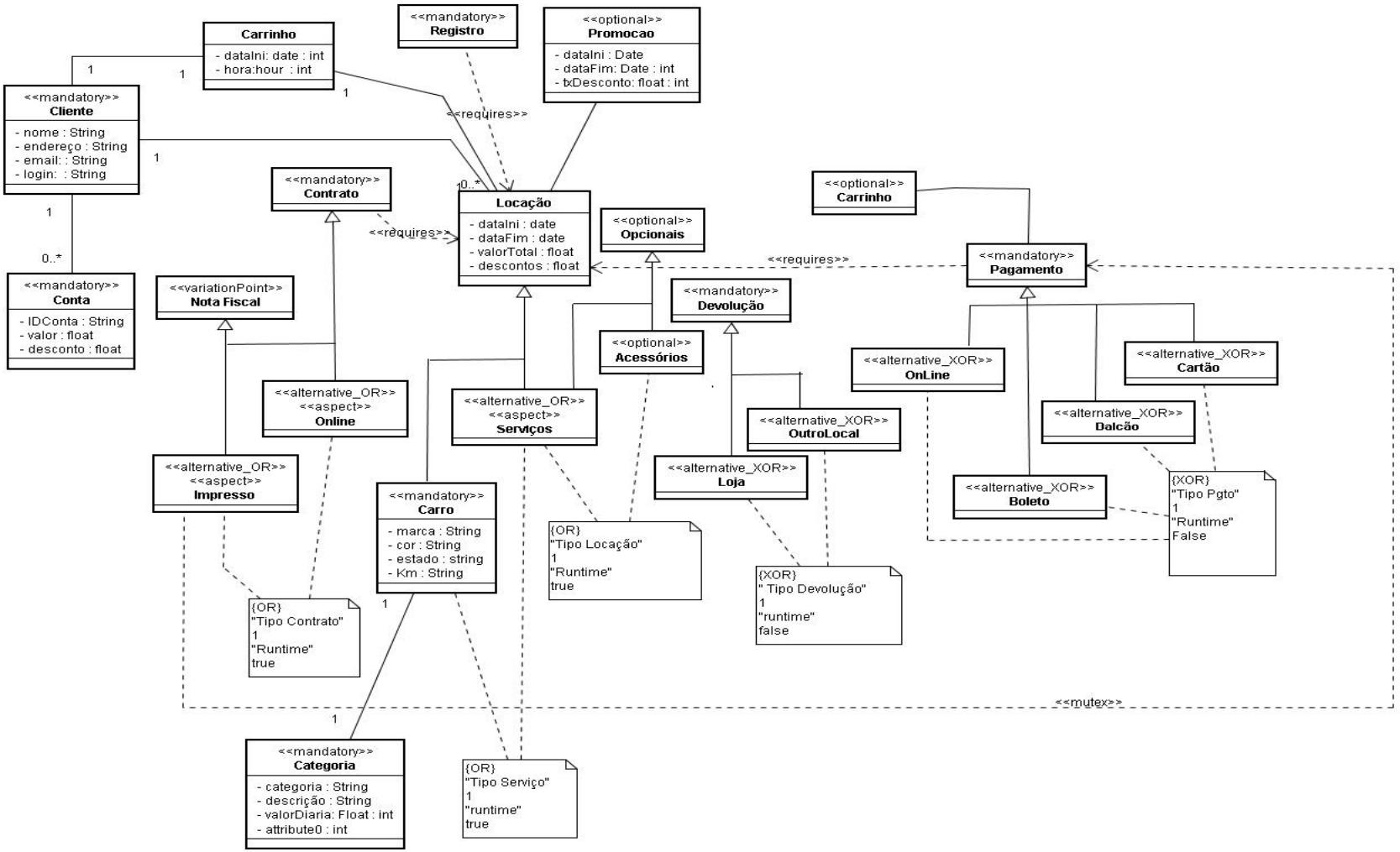


Figura 4.13– Identificação das Variabilidade e Variantes em Diagrama de Classes

4.6 Projeto Arquitetural e Projeto Interno de Componentes Base e Transversais

Para identificação das variabilidades e variantes no diagrama de componentes, a etapa relacionada ao projeto interno dos componentes sofreu algumas modificação que levaram em consideração os trabalhos de Clements at al. (2003) e Eler (2006) para projeto interno dos componentes base e transversais reutilizáveis. Além disso, com base na notação utilizada por Oliveira Junior (2005) foram inseridos os estereótipos para identificação dos componentes base e transversais. Essa notação encontram-se representadas na Tabela 4.10.

	Diagramas de Componentes	
	Relação da UML	**Estereótipo do Elemento**
Ponto de Variação	Depêndencia	`<<variable>>`
Variante Obrigatória	Depêndencia	`<<mandatory>>`
Variante Opcional	Depêndencia	`<<optional>>`
Variante Alternativa Inclusiva	Depêndencia	`<<alternative_OR>>`
Variante Alternativa Exclusiva	Depêndencia	`<<altenative_XOR>>`
Variante Mutuamente Exclusiva	Depêndencia	`<<mutex>>`
Variante Inclusiva	Depêndencia	`<<requires>>`
Variantes Transversais	Depêndencia	`<<aspect>>`
Componentes Base	Depêndencia	`<<compBase>>`
Componentes Transversais	Depêndencia	`<<compAspect>>`

Tabela 4.10 – Relação e estereótipos usados na representação gráfica das variabilidades em diagrama de componentes.

Para o projeto de componentes (base e transversais), a ProLineA propõe as seguintes orientações: 1) Identificar os componentes base a partir do diagrama de classes e do modelo de características; 2) Identificar os componentes transversais; 3) Elaborar o modelo de componentes base e transversais; e 4) Representar as variabilidades e os pontos de variação no diagrama de componentes.

Para a LP-SLC os componentes base identificados por meio das análise do diagrama de classes e do modelo de características, foram, conforme mostra a Figura 4.14: `GerLocadoraAdm`, `GerDevolução`, `GerCarrinho`, `GerConta`, `GerMultas`, `GerRetirada`, `GerCliente`, `GerCarro`, `GerPgto` e `GerLocadoraPub`. Da mesma forma, os componentes transversais identificados foram:

`GerServicos`, `GerGerarDoc`, `GerDesconto`, `GerControleAcesso`. Desses componentes somente o último representa um requisito não funcional.

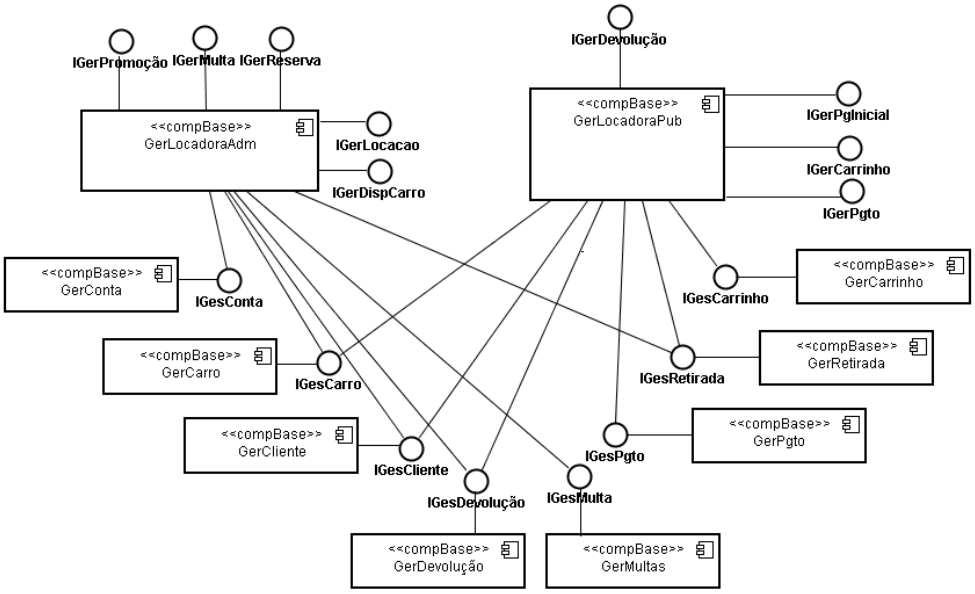


Figura 4.14 – Arquitetura dos componentes base.

Para identificar os componentes transversais foram analisados os critérios propostos por Eler (2006) (ver Capitulo 3): atomicidade, disparo, integridade e variabilidade/volatilidade. Uma vez que o enfoque do PGV são as variabilidades e os pontos de variação, o ultimo cirtério já foi verificado e identificados desde as primeiras etapas da abordagens, quando foram identificados os possíveis casos de uso e características transversais.

A Figura 4.15 apresenta o diagrama dos componentes base e transversais, ou componentes aspectuais como denominado por Eler (2006).

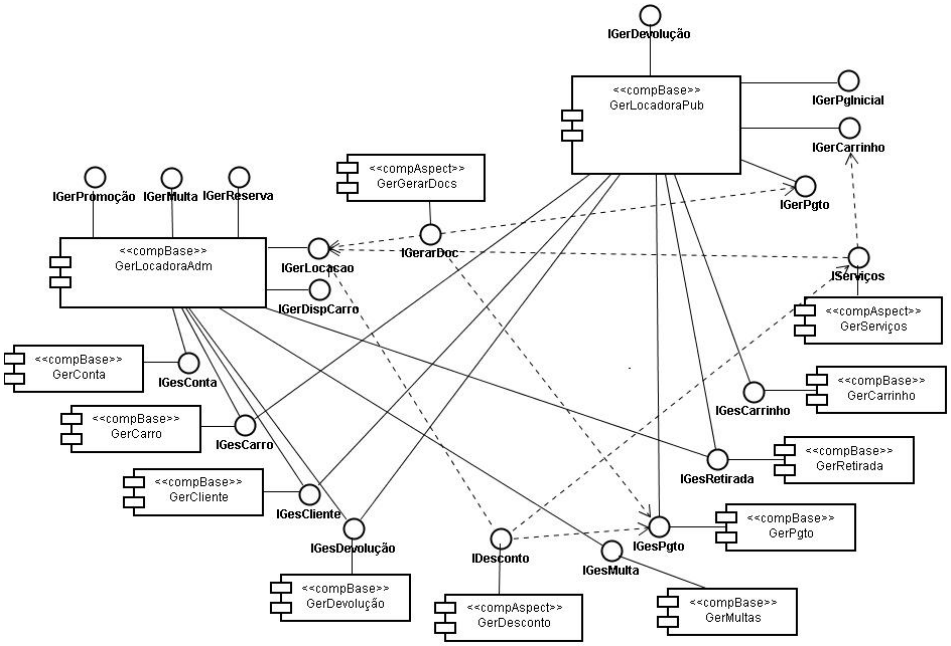


Figura 4.15 – Arquitetura dos componentes base e transversais para a LP-SLC.

4.7 Gerenciamento de Variabilidades: Delimitação das variabilidades e características transversais e Escolha dos mecanismos de implementação.

Nesta segunda parte da etapa de PGV, serão mostrados as atividades de: 1) Delimitação das Variabilidades e das Características transversais;2) Definição do Tempo de Resolução dos Pontos de Variação e Representação Gráfica das Variabiliades de adas características transversais e 3) Escolha dos mecanismos de Implementação

Para representar os pontos de variação e variantes no diagrama de componentes, é necessário que o diagrama seja (re)modelado em um nível de abstração menor, como pode ser visto na Figura 4.16 . Os componentes `GerPagto`, `GerDevolução` e `GerCarrinho` são expandidos e surgem os subcomponentes: `GerDevoluçãoLoja`, `GerDevoluçãoOutroLocal`, `GerPagtoOnline`, `GerPagtoBoleto`, `GerPagtoBalcão` e `GerPagtoCartão` para representar as variantes e os pontos de variação.

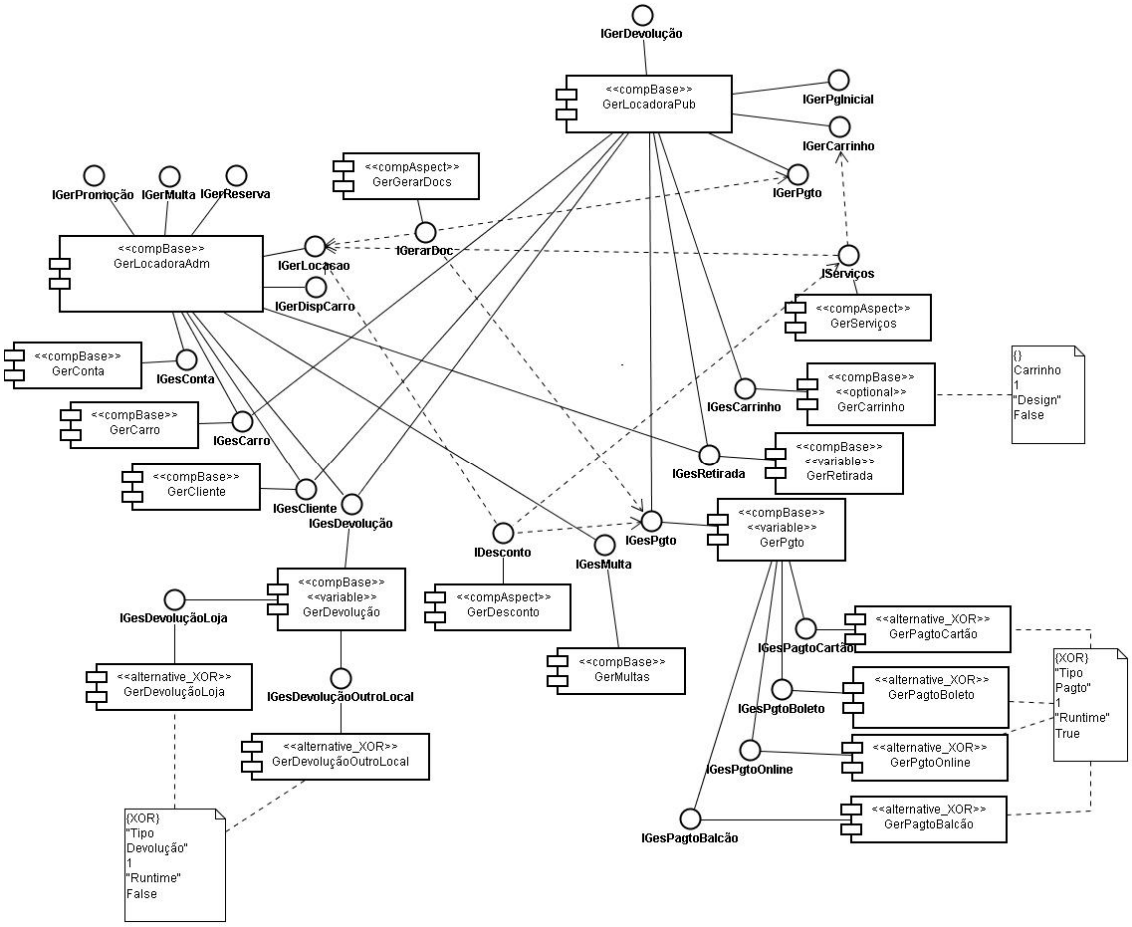


Figura 4.16 – Arquitetura dos componentes base e transversais com pontos de variação e variantes identificadas para a LP-SLC.

4.7.1 Delimitação das Variabilidades e das Características transversais

O objetivo da atividade de delimitação das variabilidades e das características transversais é definir a multiplicidade e o tempo de resolução dos pontos de variação e identificar quais pontos de variação podem aceitar a adição de mais variantes. A relação e a multiplicidade entre os pontos de variação e variantes serão representados conforme a Tabela 4.11.

	Obrigatória	**Opcional**	**Alternativa Inclusiva**	**Alternativa Exclusiva**
Relação entre Ponto de Variação e Variante(s).	1:1	1:1	1:N	1:N
Multiplicidade	1	0	0..N	1

Tabela 4.11 – Relação entre pontos de variação e variantes.

Assim, no diagrama de classes representado na Figura 4.17, pode-se notar que nas notas UML, os três “?”, da figura 4.12, serão substituídos respectivamente pela

multiplicidade do ponto de variação definidos na Tabela 4.13; pelo tempo de resolução do ponto de variação (projeto, implementação, compilação, ligação, execução ou atualização) e por último por um *true* se o ponto de variação permite a adição de mais variantes ou com um *false* se ele não permitir mais vairantes.

Para tanto, deve-se considerar os seguintes tempos de resolução:

• **projeto:** o ponto de variação é resolvido durante o desenvolvimento da LP ou dos produtos;

• **implementação:** o ponto de variação é resolvido em tempo de programação, antes da compilação;

• **compilação:** o ponto de variação é resolvido durante a compilação;

• **ligação:** o ponto de variação é resolvido durante a ligação de módulos ou de bibliotecas;

• **execução:** o ponto de variação é resolvido durante a execução do programa;

• **atualização:** o ponto de variação é resolvido durante a atualização do programa, o que acontece durante a sua execução.

A representação gráfica da delimitação das variabilidades e das características transversais realizadas nos diagramas de classes, diagrama de casos de uso e diagramas de componentes são mostrados respectivamente pelas Figuras 4.17, 4.18 e 4.19.

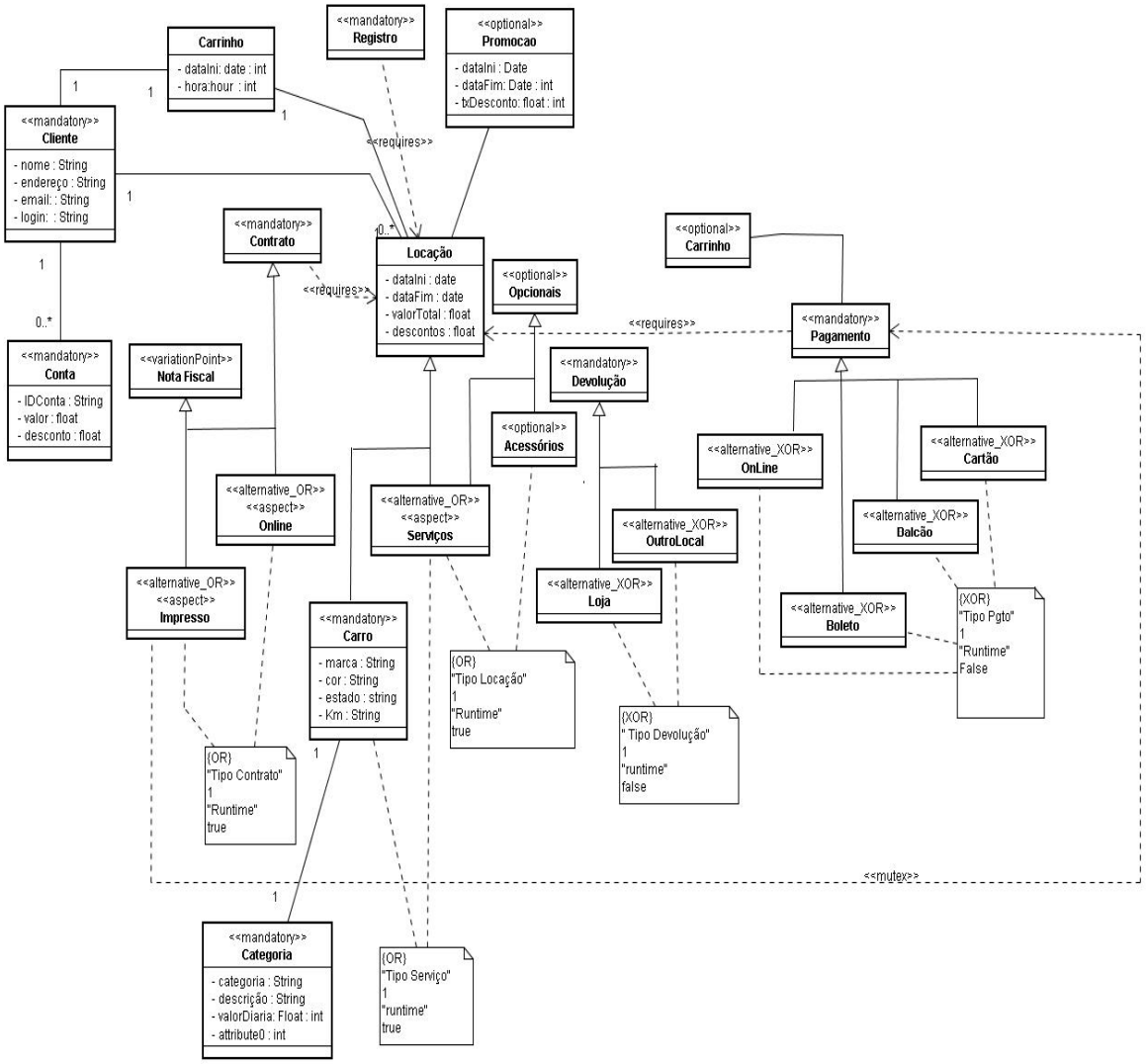


Figura 4.17 – Diagrama de classes com multiplicidade e tempo de resolução para a LP-SLC.

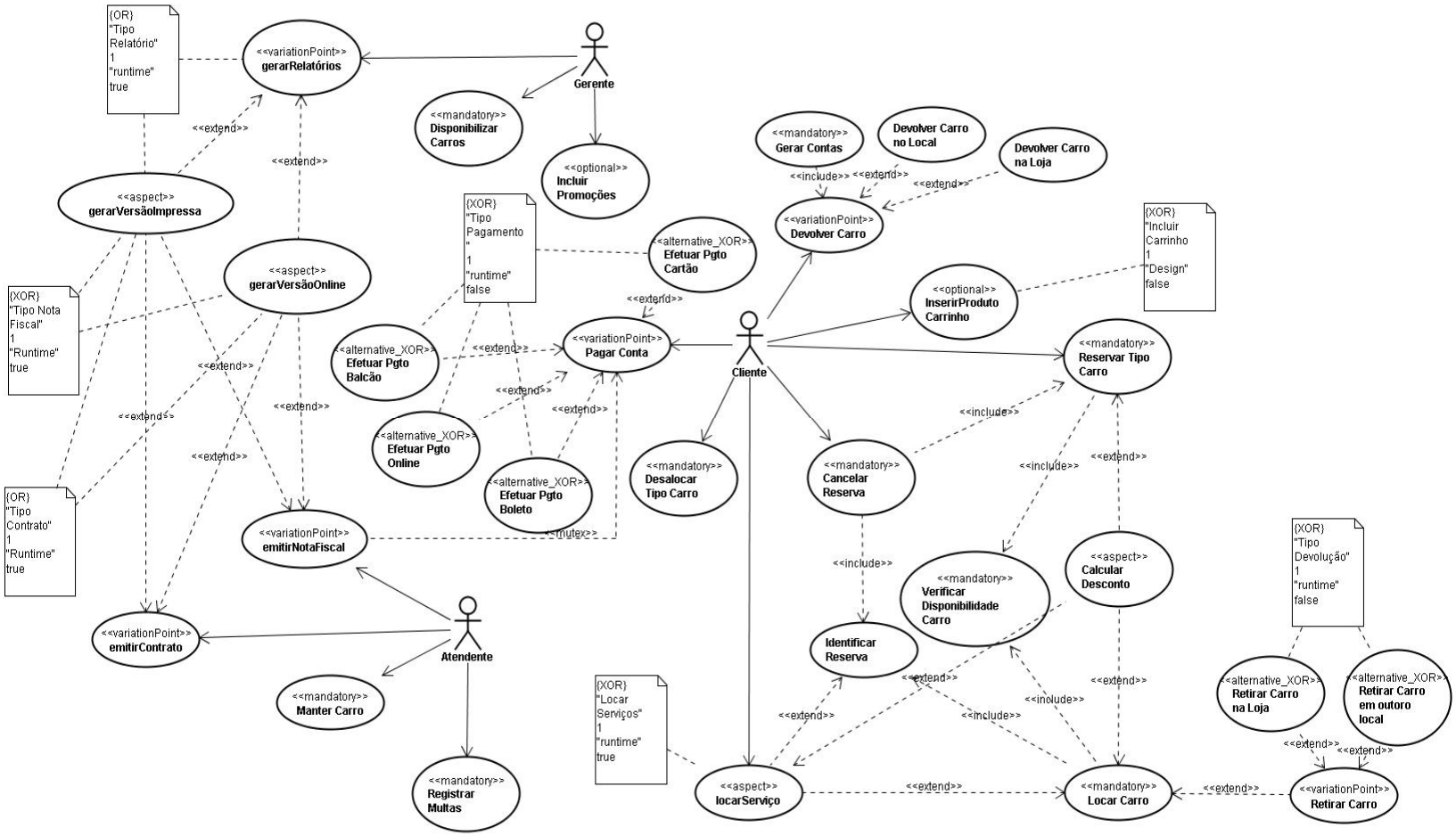


Figura 4.18 – Diagrama de Casos de Uso com Multiplicidade e tempo de resolução para a LP-SLC

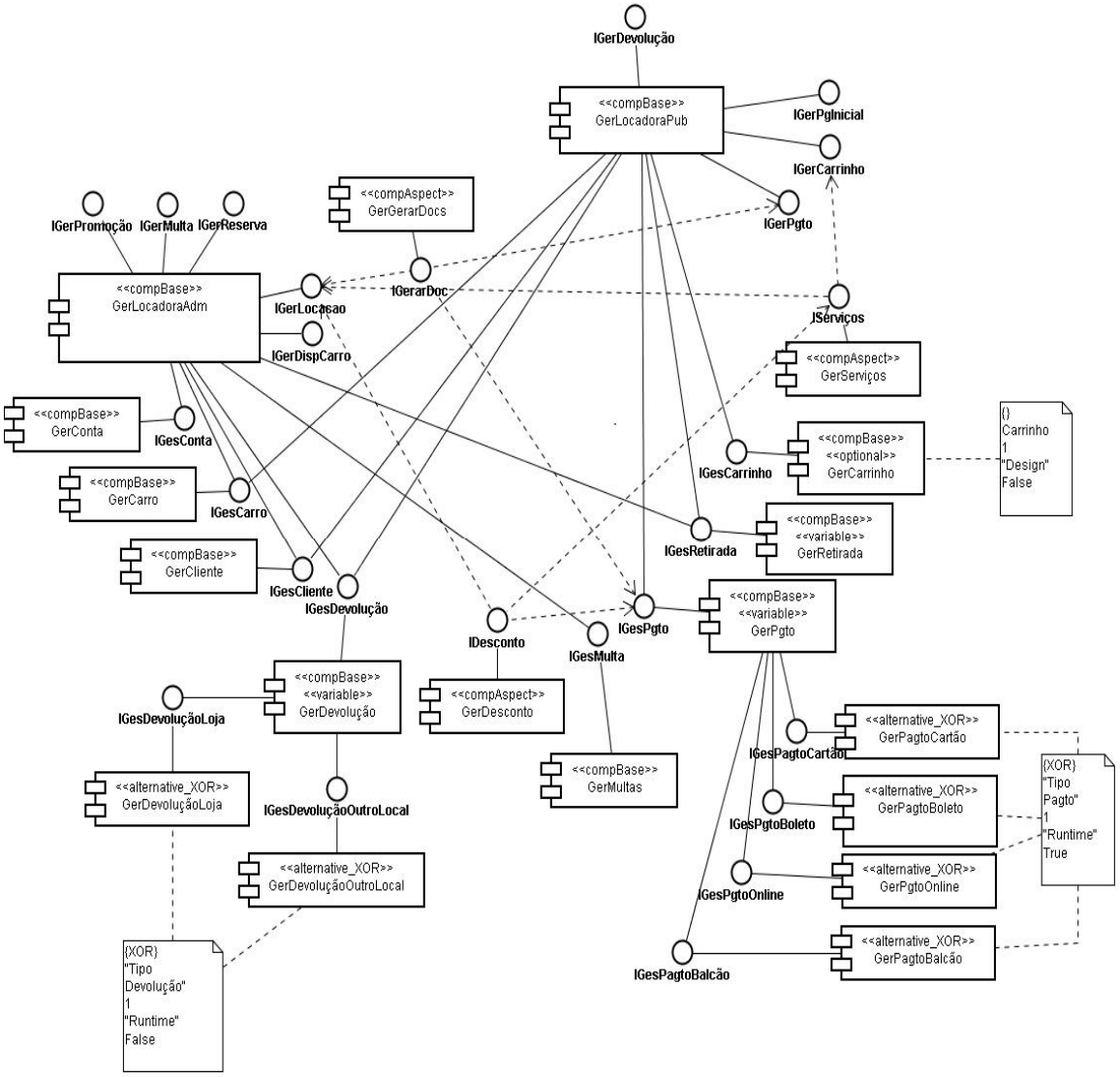


Figura 4.19 – Diagrama de Componente com Multiplicidade e tempo de resolução para a LP-SLC

4.7.2 Escolha dos mecanismos de Implementação.

O objetivo da atividade de escolhas dos mecanismos de implementação é escolher os mecanismos mais adequados para implementar as variabilidades, levando em consideração as características variáveis, as características transversais, as multiplicidades e também o tempo de resolução das variantes. A implementação das características transversais utilizando a programação orientada a aspectos (POA) só é possível se o tempo de resolução da variante for em tempo de compilação. Isso se deve as características do paradigma orientado a ascpetos, pois estes são combinados em tempo de compilação.

Para a escolha dos mecanismos de implementação serão considerados as técnicas identificadas por Van Gurp e Bosch (2002), com algumas adaptações para explorar mais a fundo a POA. Com base na Tabela 4.12 e considerando o diagrama de classe, e o diagrama de componentes base e transversais, deve ser construído o modelo de implementação das variabilidades.

<table>
<tr><th rowspan="2">Entidades de Softwares Envolvidas</th><th colspan="4">Tempo de Resolução</th></tr>
<tr><th>Desenvolvimento da Arquitetura do Produto</th><th>Compilação</th><th>Ligação</th><th>Execução</th></tr>
<tr><td rowspan="4">Componentes

Frameworks</td><td>Reorganização Arquitetural</td><td rowspan="4">Programação Orientada a Aspectos</td><td>Substituição Binária – Diretivas de Ligação</td><td rowspan="4">Arquitetura Orientada a Infra Estrutura</td></tr>
<tr><td>Componente Variante de Arquitetura</td><td rowspan="3">Substituição Binária - Física</td></tr>
<tr><td>Componente Opcional de Arquitetura</td></tr>
<tr><td>Componente Transversal de Arquitetura</td></tr>
<tr><td rowspan="3">Implementação de Frameworks

Classes</td><td>Especialização de Componentes Variantes</td><td>Superposição de Fragmento de Código</td><td>Substituição Binária – Diretivas de Ligação</td><td>Especializações de Componentes Variantes de Tempo de Execução</td></tr>
<tr><td>Especialização de Componentes Opcionais</td><td rowspan="2">Programação Orientada a Aspectos</td><td rowspan="2">Substituição Binária - Física</td><td>Implementação de Componentes Variantes</td></tr>
<tr><td>Especialização de Componentes Transversais</td><td>Especialiazações de Componentes Transversais de Tempo de Execução</td></tr>
<tr><td rowspan="3">Linha de Código</td><td rowspan="3">---</td><td>Condição em Constante</td><td rowspan="3">--------</td><td rowspan="3">Condição em Variável</td></tr>
<tr><td>Superposição de Fragmento de Código</td></tr>
<tr><td>Programação Orientada a Aspectos</td></tr>
</table>

Tabela 4.12 – Técnicas de implementação de variabilidade de acordo com o tempo de resolução. Svahnberg,Van Gurp e Bosch (2002), com adaptações.

O resultado desse modelo para a LP-SLC pode ser visto na Tabela 4.13, abaixo. Esta Tabela mostra a relação entre as variabilidades, o nível e o tempo de implementação e a sugestão dos mecanismos e estratégias de implementação.

Variação	Nível	Tempo de Resolução	Mecanismos de Implementação	Estratégia de Implementação
Tipo Devolução	Classe	Execução	Especialização de Componentes Variantes em Tempo de Execução	Utilizar os padrões de projeto Strategy e Template
Tipo Pagamento	Classe / Componente	Execução	Especialização de Componentes Variantes em Tempo de Execução	Utilizar os padrões de projeto Strategy e Template
Tipo Locação	Componente	Execução	Componente Opcional de Arquitetura	Componente que chama: delegar ao mecanismo em fases posteriores Componente chamado: criar um componente nulo que responde com valores que o componente que chama deve ignorar.
Tipo Contrato	Classe	Compilação	Especialização de Componentes Transversais em Tempo de Compilação	Utilizar POA
Tipo Nota Fiscal	Classe	Compilação	Especialização de Componentes Transversais em Tempo de Compilação	Utilizar POA
Locar Serviços	Componente	Projeto	Componente Transversal de Arquitetura	Componente que chamado: criar um componente transversal que responde com valores que o componente que chama deve ignorar
Incluir Carrinho	Componente	Projeto	Componente Opcional de Arquitetura	Utilizar os padrões de projeto Strategy e Template

Tabela 4.13 – Modelo de Implementação de Variabilidades para LP-SLC.

O objetivo alcançado utilizando os padrões de projeto descrito na Tabela 4.15, foi a reutilização do código. Por exemplo, o padrão de projeto *Strategy* (GAMMA, 2000) é utilizado quando o objetivo é representar uma operação a ser realizada sobre os elementos de uma estrutura de objetos, por exemplo, ele pode ser utilizado quando um objeto deve ser parametrizado com um de vários programas, os quais podem ser encapsulados e representados por uma única interface. Outro exemplo é o padrão de projeto Template Method (GAMMA,

2000) é utilizado quando o comportamento desejado é a inversão do controle da aplicação, ou seja, a classe-pai chama os métodos da classe-filha e não o inverso.

A escolha de mecanismos de implementação de variabilidade seguem duas atividades: Rastreamento e Controle das Variabilidades e da Análise de Configuração de Produtos Específicos. Essas atividades não apresentaram modificações que justificasse a (re)apresentação dos conceitos vistos no capítulo 2.

4.8 Considerações Finais

Neste capítulo foi apresentado a abordagem ProLineA para o desenvolvimento de LPS orientadas a aspectos. A abordagem é uma adaptação do processo de desenvolvimento de linha produto PGV (OLIVEIRA JUNIOR, 2005) e do método DBC/A (ELER, 2006), com inclusões e modificações de algumas atividades para considerar aspectos transversais no desenvolvimento.

Foram utilizados os conceitos da UML para representação de artefatos. A utilização da abordagem permite desenvolver linhas de produtos de softwares gerenciando suas variabilidades com suporte dos conceitos do paradigma orientado a aspectos.

A apresentação da ProLineA foi ilustrada através de uma LPS para o domínio de locação de carros. O proximo capitulo apresenta um exemplo de aplicação de LPS para gestão de processos seletivos (LP–GPS). Primeiramente será apresentada a LP-GPS utilizando o método existente no grupo de pesquisa em engenharia de software da UEM e o PGV proposto por Oliveira Junior(2005). Visando um estudo comparativo, o projeto para LP-GPS foi desenvolvido utilizando a abordagem ProLineA.

CAPÍTULO

5

Exemplo de Aplicação: uso e comparação da abordagem ProLineA

5.1 Considerações Iniciais

No Capítulo 4 foi apresentado a abordagem ProLineA para o desenvolvimento de LPS orientado a aspectos. A avaliação desta abordagem foi realizada observando-se alguns casos e identificando características agregadas à área de estudo, por meio da aplicação da abordagem em um exemplo.

A interpretação dos dados é baseada na observação dos resultados obtidos quando confrontados com critérios desejáveis pré-estabelecidos.

O presente trabalho configura-se como um estudo de caso observacional (BOGDAN e BIKLEN, 1997) global único (YIN, 2001), já que objetiva a compreensão de um caso específico de interesse do pesquisador. Na área da computação, principalmente relacionadas à educação, esse tipo de estudo é amplamente utilizado (MUSTARO e SILVEIRA, 2008; FORBECK, 2008; STRADA;UMANN e ALBANO, 2007). Em engenharia de software alguns trabalhos podem ser encontrados na literatura mostrando a relevância desse tipo de estratégia de pesquisa (CAGNIN ET. AL., 2004; DIAS, 2006; BARROS, WERNER e TRAVASSOS, 2002).

Segundo Yin (2001), alguns componentes são especialmente importantes para o adequado desenvolvimento de uma pesquisa empírica: uma questão de estudo pertinente; um

objetivo preciso; um caso relevante; uma vinculação lógica entre os dados apresentados e o propósito do estudo; e critérios objetivos para a interpretação do material coletado.

Kitchenham et al.(2002), estabeleceram algumas orientações para metodologias empíricas na área de engenharia de software. Para os estudos observacionais a recomendação é que todos os dados referentes ao objeto de estudos sejam minuciosamente documentados e analisados, como modelos e artefatos gerados em todas as etapas do processo. Para interpretar e validar os dados coletados estes devem ser comparados com parêmetros de referências de estudos preliminares.

A partir desses princípios, no contexto desse trabalho, foi utilizada uma aplicação que serviu como instrumento para fornecer dados para a investigação comparativa e para análise observacional e foi elaborado um plano para o estudo observacional envolvendo os seguintes passos:

1. Estabelecer os critérios desejáveis com a aplicação da abordagem.
2. Conceber uma LP para gestão de processos seletivos (LP-GPS) utilizando o PGV.
3. Conceber uma LP-GPS com a abordagem ProLineA
4. Comparar a LP-GPS sem aspectos e a LP-GPS obtida pela abordagem ProLineA.
5. Realizar uma análise comparando os resultados obtidos com os critérios estabelecidos.

A concepção da LP-GPS foi desenvolvido inteiramente pela autora, explorando as limitações da abordagem.. Porém, antes de iniciar a concepção foram estabelecidos alguns parâmetros de avaliação desejáveis:

1. esperava-se representar as características transversais que não eram levadas em consideração pelo método utilizado por Oliveira Junior (2005).
2. esperava-se que essas características pudessem ser encapsuladas em componentes transversais.

A natureza do estudo, seu delineamento como metodologia de investigação e sua aplicação são apresentadas no apêndice B. Nessa seção serão descritos apenas os resultados mais relevantes observados. Na seção 5.1 é apresentada a LP-GPS desenvolvida com PGV

(OLIVEIRA JUNIOR, 2005). A seção 5.4 mostra o projeto para LP-GPS utilizando a abordagem ProLineA. Apresenta também algumas considerações conclusivas a partir da análise comparativa entre os artefatos e as atividades modificados e gerados. As considerações finais são descritas na seção 5.4 e 5.5.

5.2 Projeto da Linha de Produto para Gestão de Processos Seletivos (LP-GPS)

A LP-GPS é uma linha de produto para sistemas web que controla e gerencia processos seletivos como concursos públicos, vestibulares, entre outros. Por um lado, os candidatos acessam o sistema via Web e podem acompanhar o processo de seleção desde a inscrição até os resultados. Por outro lado, o administrador pode gerenciar o processo de inscrição, homologação, correção e disponibilização dos resultados. As características desses sistemas podem mudar de acordo com as necessidades relacionadas a cada tipo de processo seletivo ou às necessidades dos usuários (ex. pessoas físicas ou jurídicas, instituições públicas ou privadas, entre outros).

Este primeiro exemplo de aplicação foi desenvolvido utilizando o PGV (OLIVEIRA JUNIOR, 2005).

Foram analisados três tipos de processos seletivos diferentes: 1. processo seletivo para ingresso nos cursos oferecidos pela Univesidade do Norte do Paraná (UENP/FALM), 2. concurso do Conselho Regional de Engenharia do Paraná – CREA/PR e 3. concurso público para contração de docentes e técnicos administrativos da (UENP/FALM).

A análise desses diferentes tipos de processos seletivos originou a LP–GPS a qual é descrita pelo: modelo de casos de uso, modelo de características e diagrama de componentes conforme apresentados nas Figuras 5.1, 5.2 e 5.3. A documentação detalhada para a LP-GPS encontra-se no Anexo B.

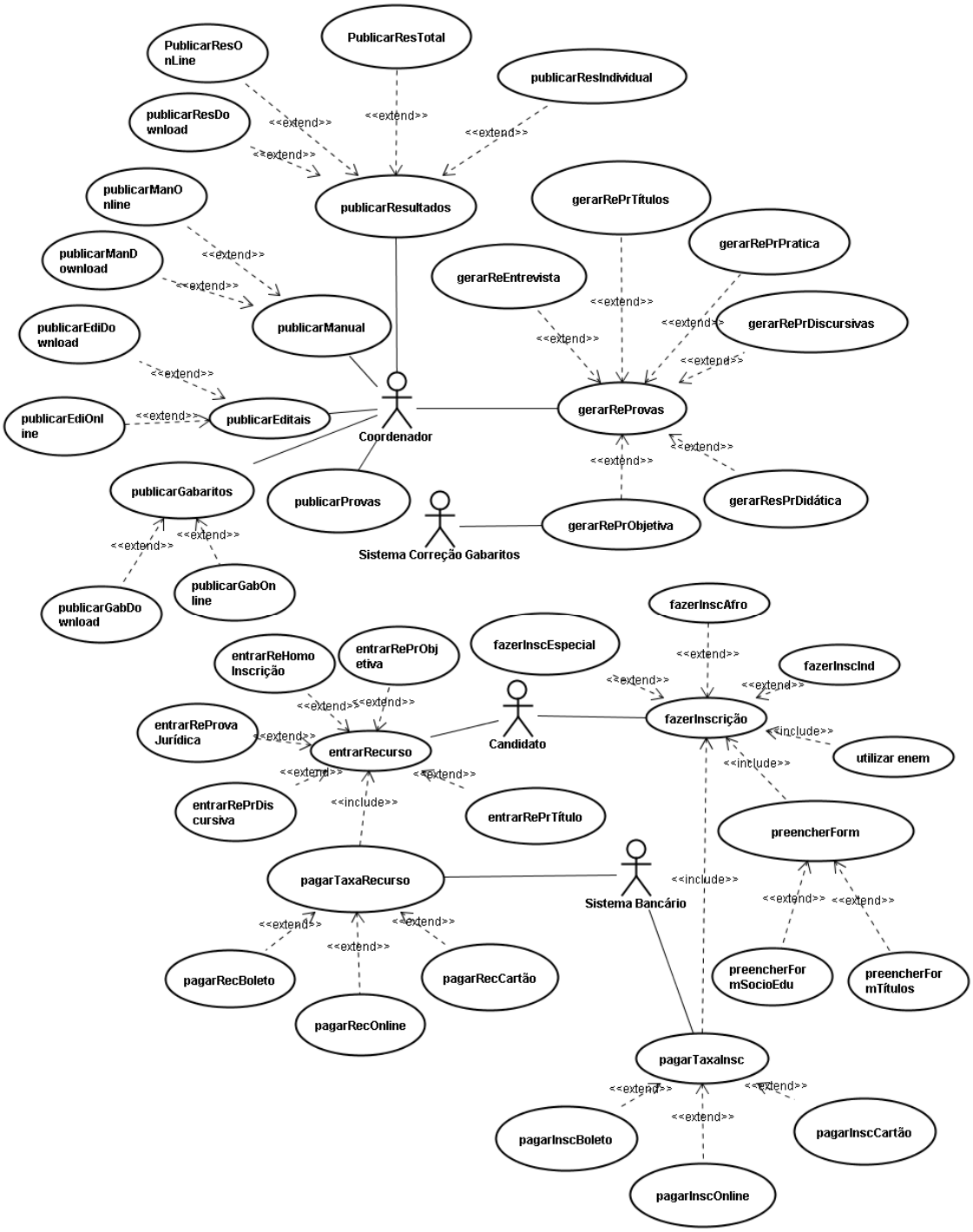


Figura 5.1 – Modelo de casos de uso para LP-GPS sem aspectos.

A análise dos requisitos de cada sistema originou a especificação dos requisitos do domínio e a identificação das características comuns e variáveis da LP-GPS. Os casos de uso identificados foram:

1 fazerInscrição
2 pagarTaxaIsncrição
3 pagarInscBoleto
4 pagarIsncOnline
5 pagarInscCartão
6 preencherForm
7 preencherFormSocioEdu
8 preencherFormTítulos
9 utilizarEnem
10 fazerInscInd
11 fazerInscAfro
12 fazerInscEspecial
13 entrarRecurso
14 entrarRePrObjetiva
15 entrarReHomoInscrição
16 entrarReProvaJurídica
17 entrarRePrDiscursiva
18 pagarTaxaRecurso
19 pagarRecBoleto
20 pagarRecOnline
21 pagarRecCartão
22 gerarRePrObjetiva
23 gerarReProvas
24 gerarReEntrevista
25 gerarRePrTítulos
26 gerarRePrPratica
27 gerarRePrDiscursiva
28 gerarResPrDidática
29 gerarRePrObjetiva
30 publicarProvas
31 publicarGabaritos
32 publicarGabDownload
33 publicarGabOnline
34 publicarEditais
35 publicarEdiOnline
36 publicarEdiDownload
37 publicarManual
38 publicarManOnline
39 publicarManDownload
40 publicarResultados
41 publicarResDownload
42 publicarResOnline
43 publicarResTotal
44 publicarResIndividual

As características que compõem o modelo de características da Figura 5.2, são:

1 publicação
2 PubEditais
3 PubResultados
4 PubManual do Candidatos
5 PubEdiOnline
6 PubEdiDown
7 PubResTotal
8 PubResIndividual
9 PubResDown
10 PubResOnline
11 PubManOline
12 PubManDownloa
13 Pagamento
14 PagtoRecurso
15 PagtoInscrições
16 PagtoOnLine
17 PagtoCartão
18 PagtoBoleto
19 Inscrição
20 Enem
21 Inscrição Geral
22 Inscrição Afro
23 Inscrição Ind
24 Inscrição Especial
25 Recursos
26 RecTitulos
27 RecPrática
28 RecDidática
29 RecObjetiva
30 RecEntrevista
31 RecDiscursiva
32 Provas
33 Prova Títulos
34 Prova Discursiva
35 Prova Prática
36 Prova Objetiva
37 Prova Entrevista
38 Prova Didática

A Tabela 5.1 mostra o modelo de rastreamento para a LP-GPS. A análise do modelo de casos de uso, do modelo de características, do modelo de rastreamento de variabilidades e do diagrama de componentes permitiram tomar decisões sobre os mecanismos de implementação.

características casos de uso	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	...
1																			*		*									
2													*		*															
3																		*												
4																*														
5													*				*													
...																														
9																				*										
10																			*				*							
11																			*			*								
...																														
19													*																	
20													*					*												
21													*				*													
22																									*					
23																									*					
24																									*					
25																									*	*				
26																									*		*			
27																									*			*		
28																									*					
29																									*					
30	*																													
31	*																													
32	*																													
33	*																													
34	*	*																												
35	*	*			*																									
36	*	*																												
37	*			*																										
38	*			*							*																			
39	*			*								*																		
40	*																													
41	*		*				*		*	*																				
42	*		*																											
43	*		*			*																								
44	*		*					*																						

Tabela 5.1 - Modelo de Rastreamento das variabilidades para a LP-GPS.(cont)

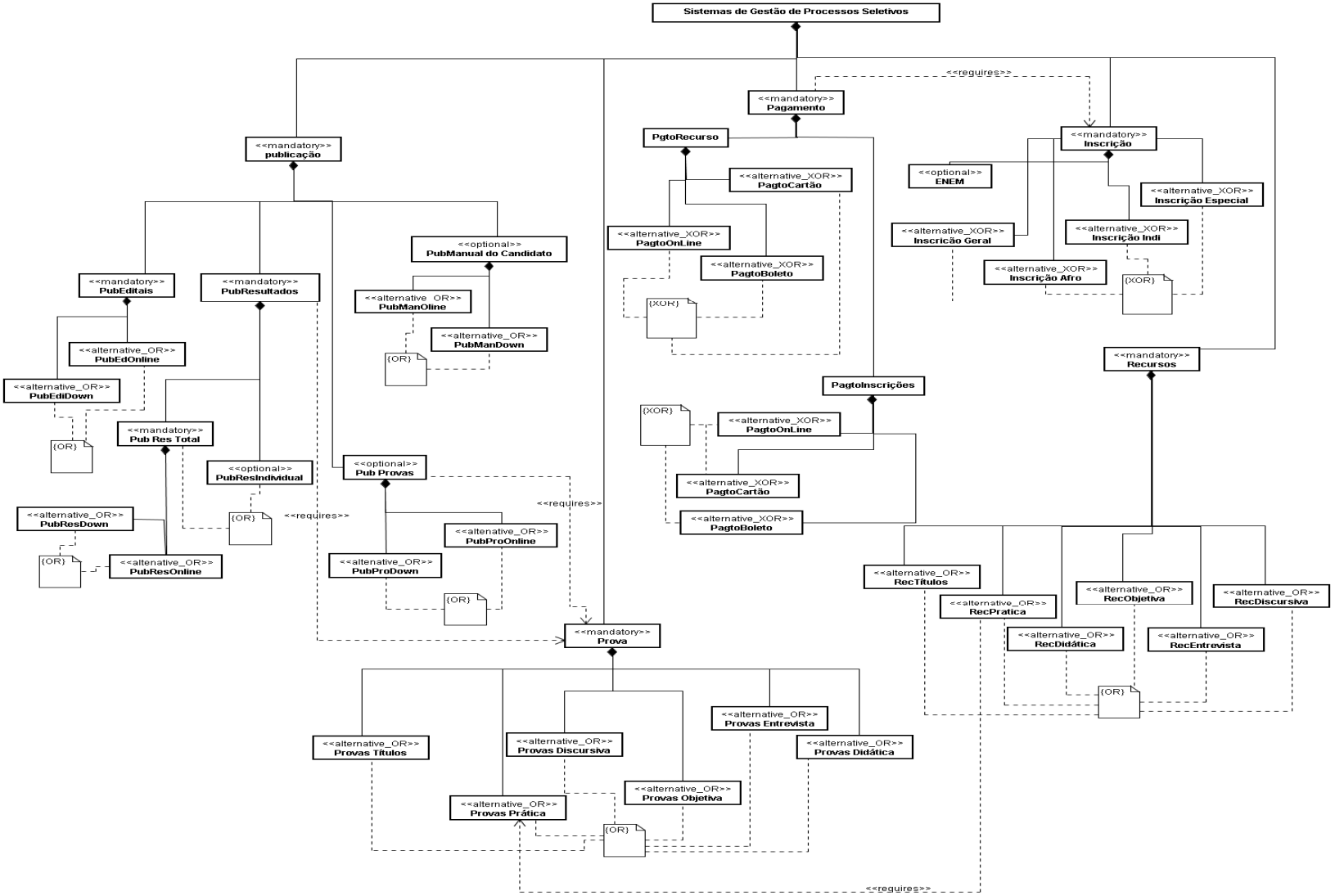


Figura 5.2 – Modelo de características para a LP-GPS sem aspectos

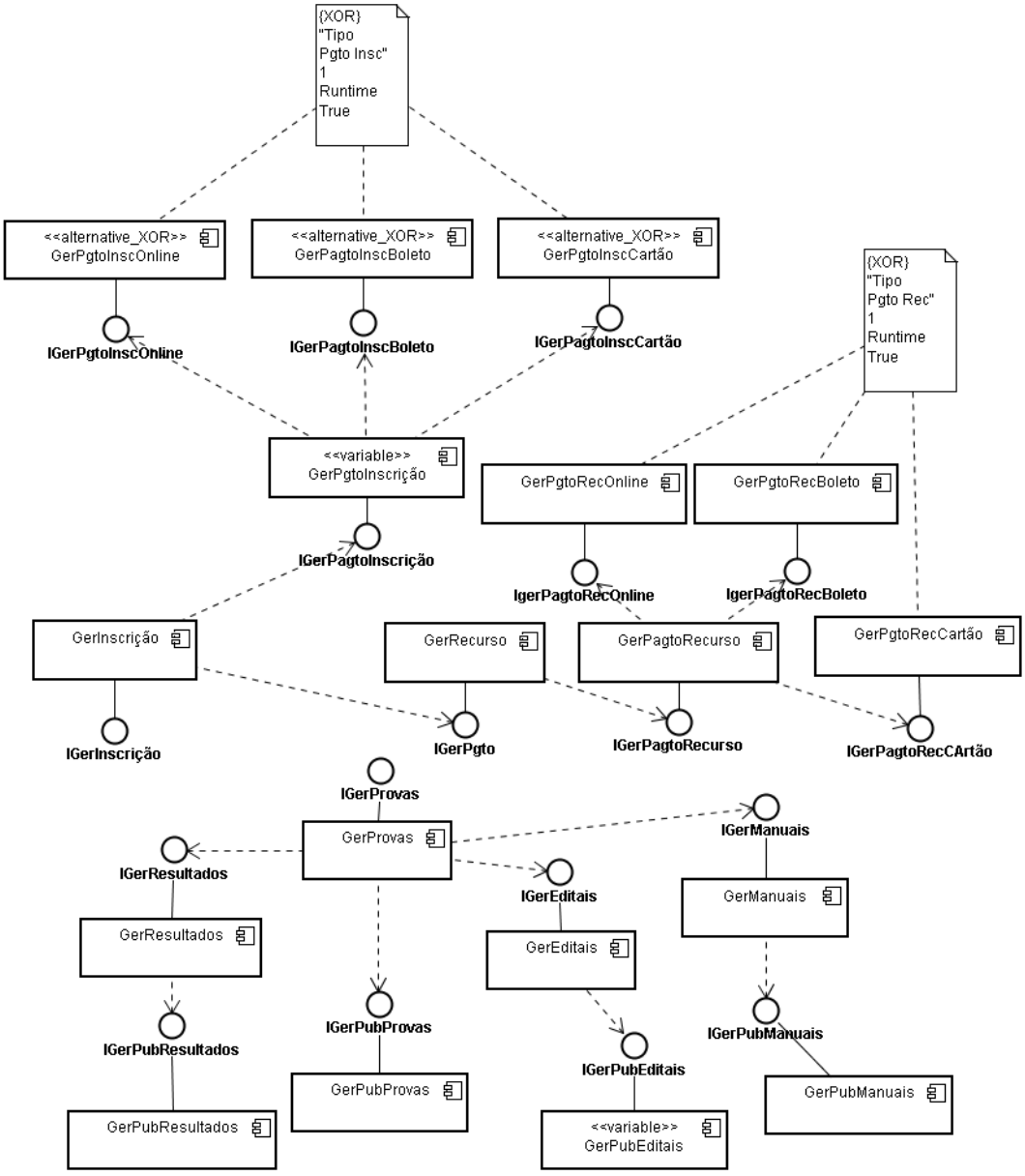


Figura 5.3 – Diagrama de Componentes

Variação	Nível	Tempo de Resolução	Mecanismos de Implementação	Estratégia de Implementação
Tipo Aplicação	Componente	Compilação	Especialização de Componentes Variantes em Tempo de Execução	Utilizar os padrões de projeto Strategy e Template
Tipo Pagamento	Classe / Componente	Execução	Especialização de Componentes Variantes em Tempo de Execução	Utilizar os padrões de projeto Strategy e Template
Tipo Resultados	Componente	Execução	Componente Opcional de Arquitetura	Componente que chama: delegar ao mecanismo em fases posteriores Componente chamado: criar um componente nulo que responde com valores que o componente que chama deve ignorar.
Tipo Correção	Classe	Compilação	Especialização de Componentes em Tempo de Compilação	POA
Tipo Prova	Componente	Compilação	Especialização de Componentes em Tempo de Compilação	POA

Tabela 5.2 - Modelo de implementação das variabilidades para a LP-GPS sem aspectos.

5.3 Projeto da Linha de Produto para Gestão de Processos Seletivos pela abordagem ProLineA.

Nesta seção são mostrados alguns artefatos do projeto LP-GPS pela abordagem ProLineA. O documento de requisitos completo pode ser visto no Apêndice B.

5.3.1 Especificação dos Requisitos e Especificação do Sistema

Para o levantamento de requisitos e a construção do DCU/F, foram utilizados os mesmos artefatos e documentos do projeto da LP-GPS sem aspectos. A Tabela 5.1 mostra uma relação entre os requisitos funcionais do domínio (apêndice C), os casos de usos e os casos de uso cobertos pelo DCUF. A Tabela 5.2 mostra a relação entre os requisitos não funcionais do domínio, os casos de uso e os casos de uso cobertos pelo DCU(F/NF).

A Figura 5.4 mostra um diagrama parcial para integração dos casos de uso funcionais e não funcionais do domínio.

Requisito	NCtrl	Caso de uso	DCU/F
Req.1, Req.4, Req.5, Req.6.	1	alterarInscrição	
Req. 1, Req.4, Req.5, Req.6.	2	cancelarInscrição	
Req. 1, Req.4, Req.5, Req.6	3	fazerInscrição	*
Req.5, Req.4, Req.6	4	fazerInscEspecial	*
Req.5	5	fazerInscAfro	*
Req.4	6	fazerInscEsp	*
Req.6	7	fazerInscInd	*
Req. 1, Req.4, Req.5, Req.6	8	utilizarEnem	*
Req. 1, Req.4, Req.5, Req.6	9	preencherFormTitulos	*
Req. 1, Req.4, Req.5, Req.6	10	preencherFormSocioEdu	*
Req.2	11	pagarTaxaInsc	*
Req.2	12	pagarOnline	*
Req.2	13	pagarBoleto	*
Req.2	14	pagarCartão	*
Req.3, Req.1, Req.4, Req.5, Req.6	15	solicitarIsenção	
Req.8	16	controlarHomo	
Req.8	17	imprimirCartãoHomo	*
Req.8	18	segundaViaCartão	
Req.9	19	publicarEditais	*
Req.9	20	publicarResultados	*
Req.9	21	publicarResTotal	*
Req.9	23	publicarResInd	*
Req.9	24	publicarManual	*
Req.9	25	publicarManual	
Req.9	27	publicarGabaritos	*
Req.9	28	publicarProvas	*
Req.10, Req. 11	29	gerarRePrObjetivas	*
Req.10, Req. 11	30	gerarRePrDiscursivas	*
Req.10, Req. 11	31	gerarRePrPráticas	*
Req.10, Req. 11	32	gerarRePrDidática	*
Req.10, Req. 11	33	gerarReEntrevista	*
Req.10, Req. 11	34	gerarRePrTitulos	*
Req.12	35	convocarCandExame	
Req.12	36	convocarCandNomear	
Req.12	37	convocarCandMatrícula	
Req.12	38	convocarCandFase	
Req.12	39	imprimirCartãoFase	
Req.13	40	entrarRecurso	*
Req.13	41	pagarTaxaRecurso	*
Req.13	42	entrarRePrTitulo	*
Req.13	43	entrarRePrObjetiva	*
Req.13	44	entrarRePrJurídica	*
Req.13	45	entrarRePrDiscursiva	*
Req.13	46	entrarReHomoInscrição	*
Req.10, Req.11	47	gerarResProvas	*
Req.9	48	publicarOnline	*
Req.9	49	publicarDownload	*

Tabela 5.3 – Requisitos x casos de uso x casos de usos cobertos pelo DCUF

Requisito	NCtrl	Caso de uso	DCU/NF
Req.16	40	gerUsuário	
Req.17	41	auntentUsuário	*
Req.18	42	criptSenhas	
Req.19	43	registrarOperUser	*
Req.20	44	permitirAcesso	*
Req.21	45	persistirDados	*
Req.22	46	controlarTempInscrição	*
Req.23	47	controlarTempFichaSócio	*
Req.24	48	controlarTempImpressão	*

Tabela 5.4 – Requisitos não funcionais x casos de uso não funcionais x casos de usos cobertos pelo DCU/NF

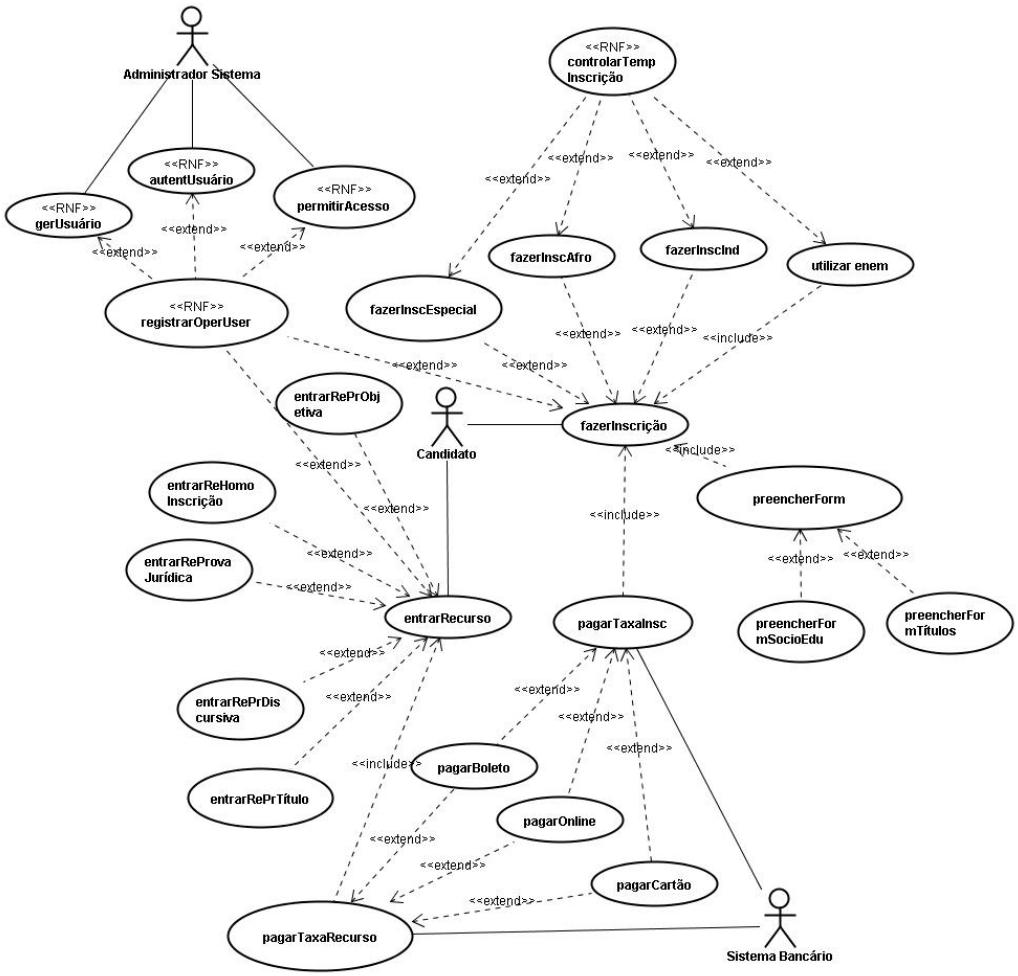


Figura 5.4 – Diagrama parcial integrando os casos de usos funcionais e não funcionais

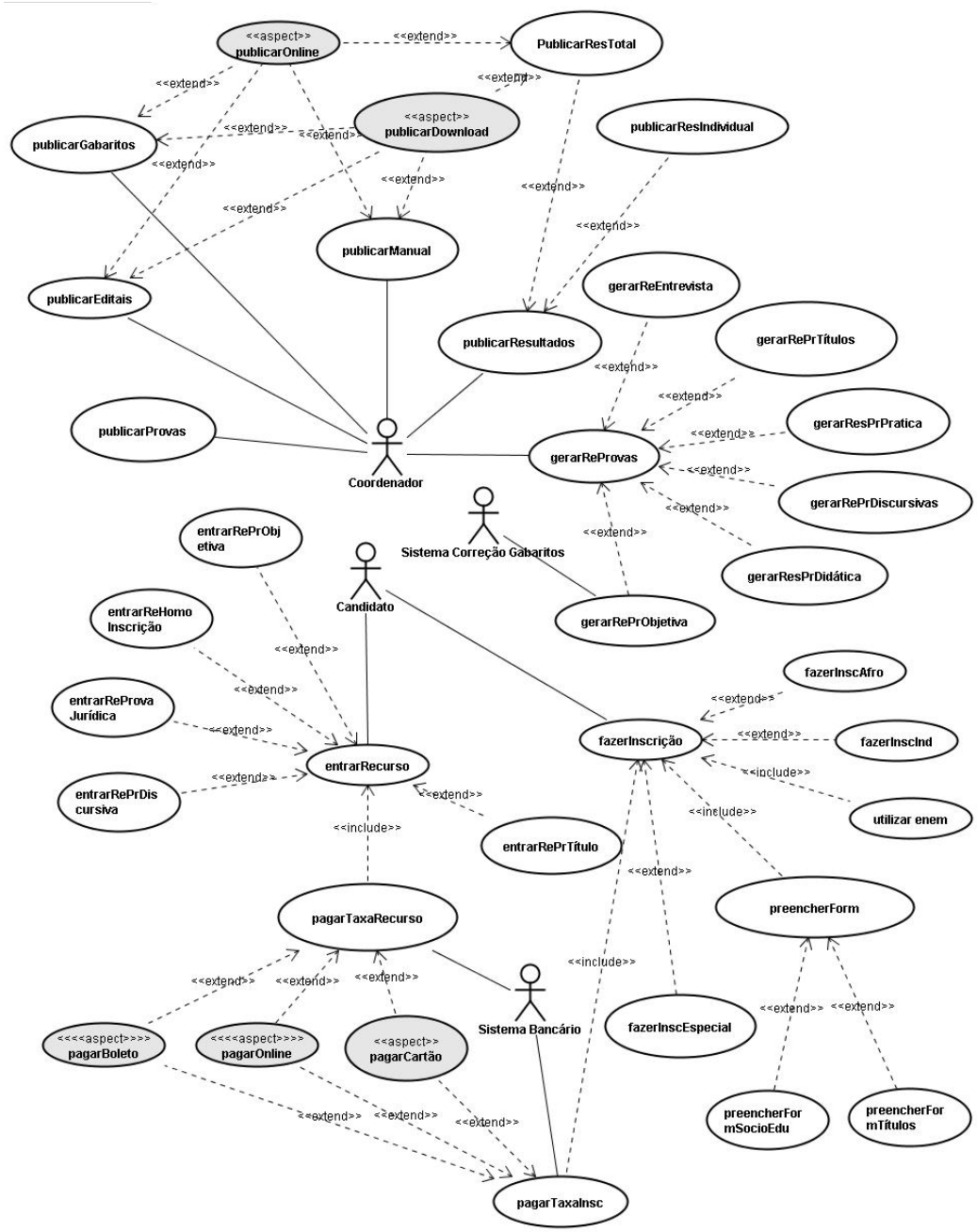


Figura 5.5 – Diagrama de casos de uso com possíveis aspectos identificados.

Os possíveis aspectos identificados através do modelo de casos de uso e da especificação dos requisitos foram:

Requisito	Casos de Uso	Tipo
Req. 9	publicarOnline	Funcional
Req. 9	publicarDownload	Funcional
Req. 2	pagarBoleto	Funcional
Req. 2	pagarOnline	Funcional
Req. 2	pagarCartão	Funcional
Req.19	registrarOperações	Não-funcional
Req. 22	controlarTempInscrição	Não-funcional
Req. 21	persistirDados	Não-funcional

Tabela 5.5 – Possíveis aspectos x Requisitos funcionais e não funcionais

5.3.2 Elaboração do Modelo de Características

Os artefatos de entrada para a atividade de elaboração do modelo de características foram o modelo de casos de uso funcionais e o modelo de casos de uso não funcionais. A Figura 5.7 mostra o modelo de características produzido nessa atividade e a Figura 5.6 mostra o modelo conceitual de negócios.

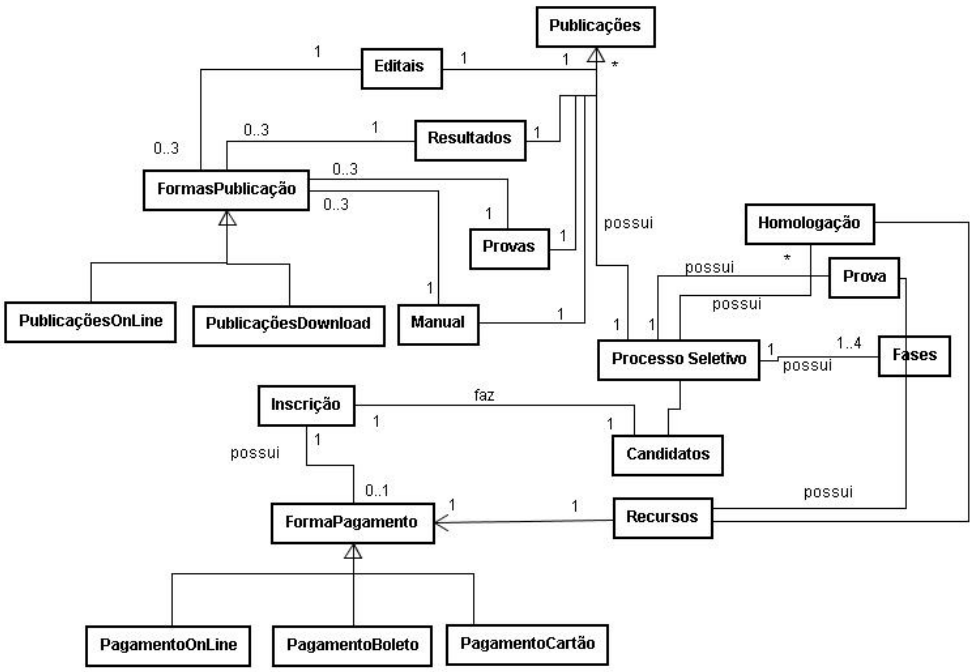


Figura 5.6 – Modelo Conceitual de Negócio

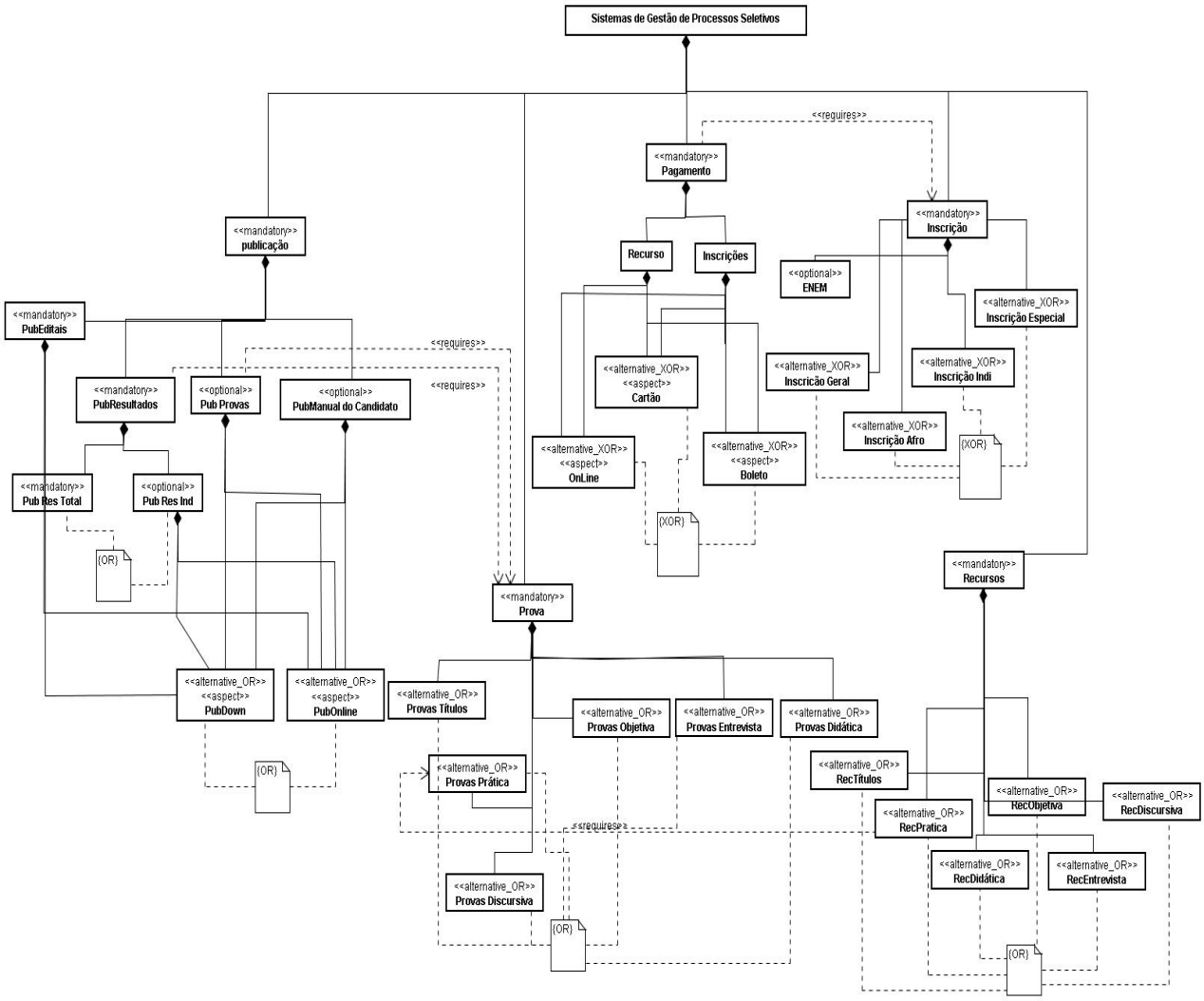


Figura 5.7 – Modelo de Características

5.3.2 Processo de Gerenciamento de Variabilidades (PGV)

Os artefatos de entrada para as atividades do PGV são: o modelo de características, o diagrama de casos de uso e o modelo conceitual do negócio. Os artefatos de saída do PGV são: o modelo de rastreamento de variabilidades, e a evolução do diagrama de casos de uso, diagrama de classes com multiplicidade e resolução dos pontos de variação identificados.

A Figura 5.6 mostra a evolução do diagrama de casos de uso evoluído, com as multiplicidades e tempo de resolução dos pontos de variação.

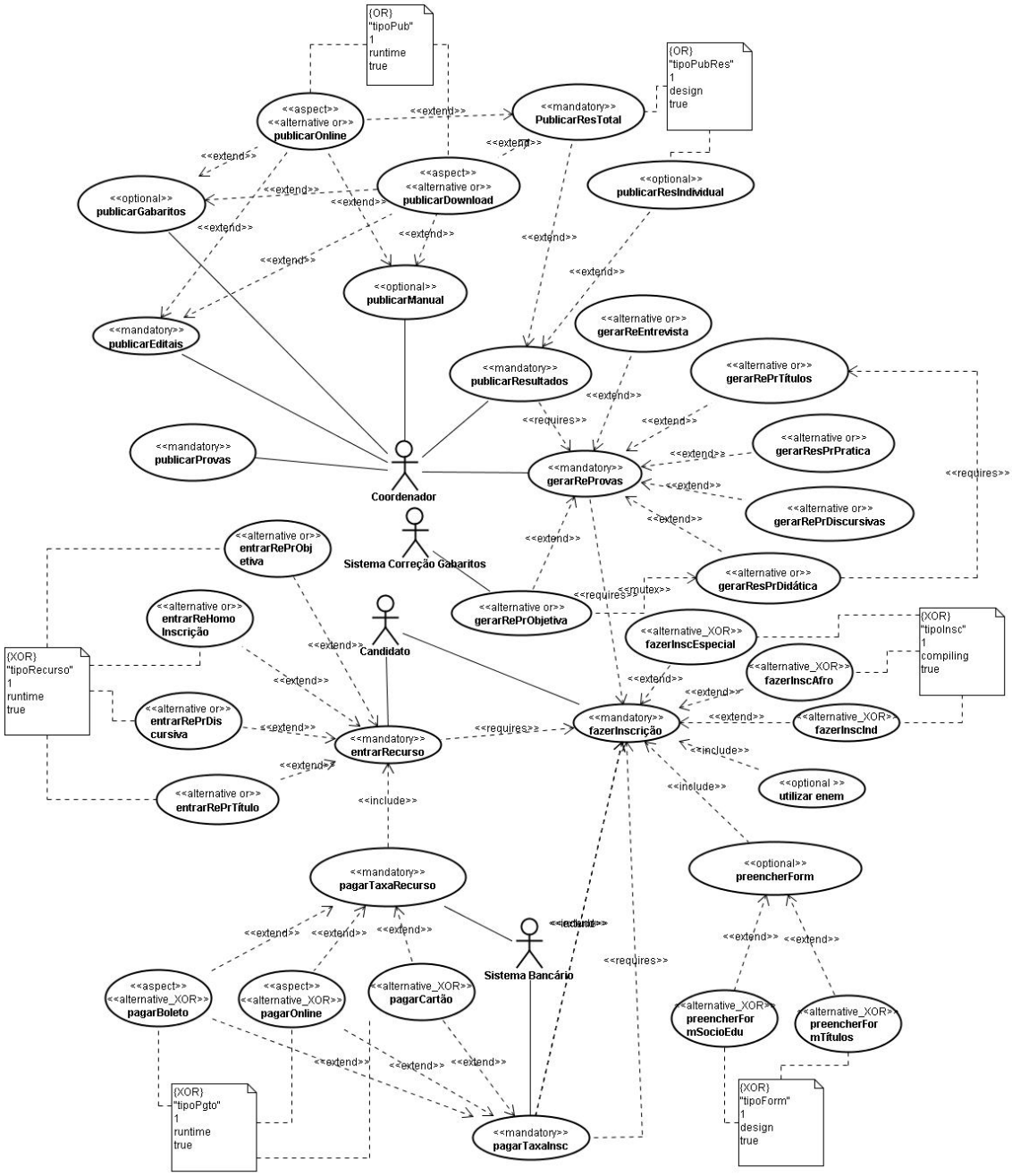


Figura 5.8 – Representação de multiplicidade e tempo de resolução de variação em diagrama de casos de uso para a LP GPS com aspectos.

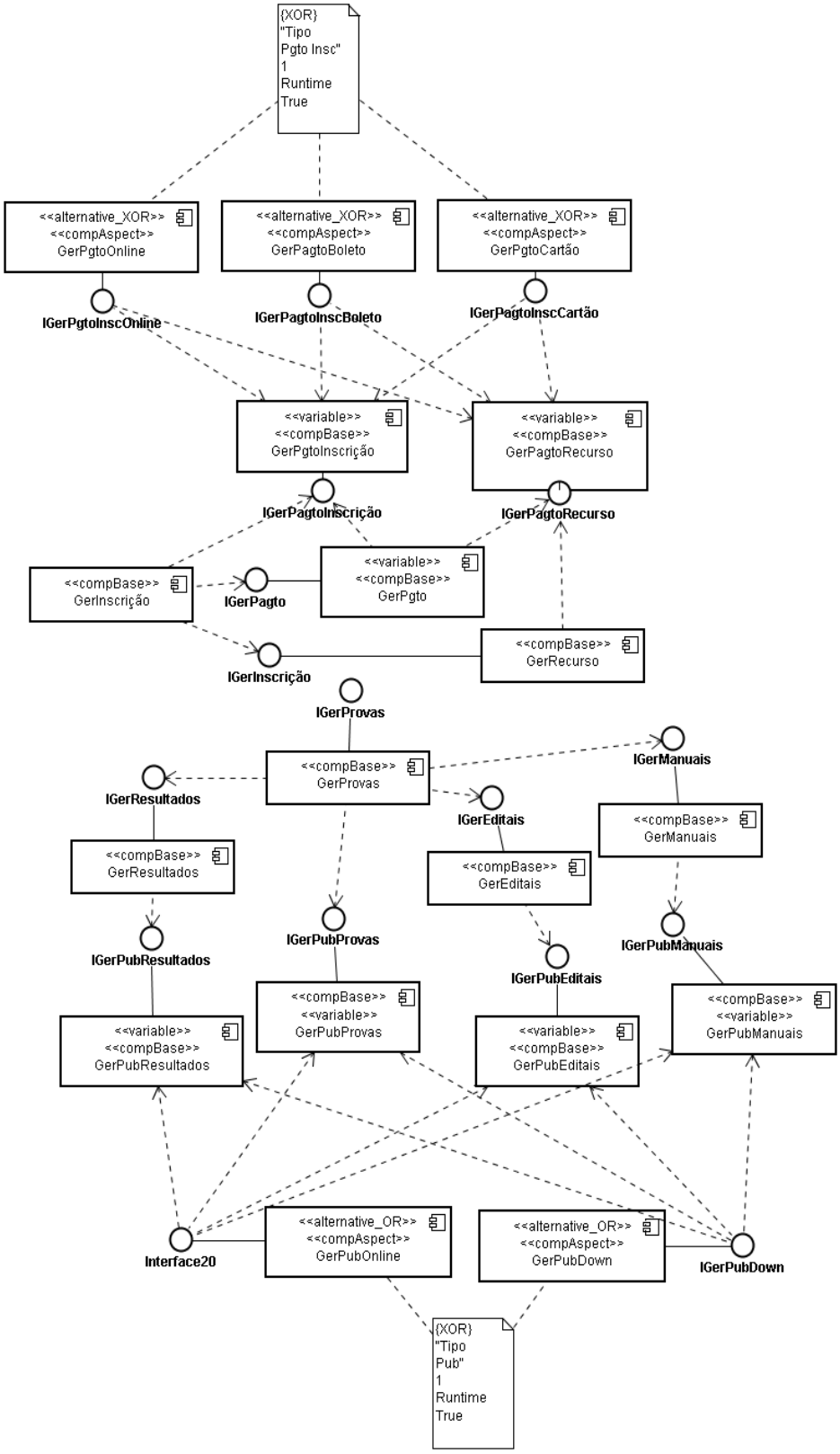


Figura 5.9 – Representação de multiplicidade e tempo de resolução de variação em diagrama de componentes para a LP GPS com aspectos.

5.4 Discussões e Resultados

Os parâmetros desejáveis foram avaliados por meio dos resultados obtidos com os projetos apresentados nas seções anteriores. Os resultados obtidos são discutidos em dois momentos distintos nas seções 5.4.1 e 5.4.2 respectivamente e de acordo com os parâmetros estabelecidos: Representação de Características Transversais e Componentes Transversais.

5.4.1 Representação de Características Transversais

As Figuras 5.1 e 5.4 mostram que é possível representar as características transversais a partir da identificação dos casos de uso. Isso foi possível por meio da análise do diagrama de casos de uso da Figura 5.1 e do modelo de rastreamento de variabilidades da Tabela 5.1, que possibilitou a identificação dos casos de uso transversais.

A tabela 5.6 compara alguns grupos de características, pelo PGV, que foram encapsuladas em características tranversais (ProLinea).

Característica (PGV)	**Características (ProLineA)**
PubEdOnline	PubOnline
PubManOnline	
PubResOnline	
PubProOnline	
PubEdDown	PubDown
PubResDown	
PubProDown	
PubManDown	
PagtoInscOnline	PagtoOnline
PagtoRecOnline	
PagtoInscCartão	PagtoCartão
PagtoRecCartão	
PagtoInscBoleto	PagtoBoleto
PagtoRecBoleto	

Tabela 5.6 – Possíveis aspectos x Requisitos funcionais e não funcionais

A descrição dos requisitos, dos casos de uso (funcionais e não funcionais) permitiu identificar quais características possuíam comportamentos transversais.

5.4.2 Componentes Transversais

A comparação entre a Figura 5.3 e 5.9 mostram que é possível encapsular grupos de características que possuem o mesmo comportamento, em componentes transversais. A Figura 5.8 mostra o digrama de componentes base e transversais, em que o componentes `GerPgtoInscOnline` e `GerPagtoRecOnline` da figura 5.3 são encapsulados no componente transversal `GerPgtoOnline` e assim respectivamento para os componentes transversais `GerPgtoBoleto` e `GerPgtoCartão`.

Outro exemplo verificado analisando os diagramas de componentes, são os componentes: `GerPubResultados, GerPubProvas, GerPubEditais e GerpubManuais` da Figura 5.3, que por possuírem operações de interface (obter informação da publicação, gerar relatórios, gerar relatório on line, gerar realtório para download) deram origem a dois componentes transversais: `GerPubOnline` e `GerPubDown`.

5.5 Considerações Finais

Na seção 5.1 foram apresentados alguns parâmetros de avaliação desejáveis. Primeiro, esperava-se representar todas as características transversais que não eram levados em consideração pelo método utilizado por (OLIVEIRA JUNIOR, 2005). Segundo, esperava-se que essas características pudessem ser encapsuladas em componentes transversais e finalmente esperava-se ainda identificar um menor número de variabilidades, uma vez que são que são consideradas as características transversais a pontos de variação diferentes.

O projeto da LP–SLC realizado ao longo da concepção da ProLineA, permitiu não apenas o refinamento da abordagem, mas também que esses parâmetros pudessem ser estabelecidos.

A seção 5.2 apresentou um projeto preliminar de uma LPS-GPS sem considerar aspectos com o objetivo de realizar um estudo comparativo com o projeto apresentado na seção 5.3 para a LP-GPS com aspectos através da ProLineA.

Desta forma, em linhas gerais, os parâmetros desejáveis foram avaliados por meio dos resultados obtidos e as contribuições da abordagem são :

- a representação de características transversais, obtidas através do rastreamento dos casos de uso transversais (funcionais e não funcionais) e do modelo de características.
- o encapsulamento de grupos de características em componentes bases e componentes transversais.

CAPÍTULO

6

Conclusão

6.1 Considerações Iniciais

Neste trabalho foi apresentado uma abordagem para o desenvolvimento de LPS orientado a aspectos, denominada ProLineA. Desta forma, foram apresentados inicialmente os conceitos, princípios e objetivos de linha de produto de software e do paradigma orientado a aspectos. Foram apresentados também alguns métodos de desenvolvimento de linhas de produtos e do desenvolvimento orientado a aspectos.

Observou-se por meio do desenvolvimento de uma LPS para o domínio de gestão de processos seletivos que as características transversais encontravam-se espalhadas e entrelaçadas pelos modelos e artefatos produzidos nas atividades do desenvolvimento do núcleo de artefatos e do gerenciamento de variabilidades quando a LPS foi desenvolvida somente com base no PGV. Isso causa problemas no entendimento, manutenção, evolução e reuso dos artefatos e também dos componentes a serem implementados e reutilizados (ELER, 2006). Assim, como os conceitos do paradigma orientado a aspectos já comprovou que é possível resolver e amenizar os problemas causados pelo espalhamento e entrelaçamento de código e também nas fases premilinares a implementação, uma proposta de solução viável foi a utilização desses conceitos combinados ao desenvolvimento de LPS. A partir desta definição, foram discutidos e apresentados abordagens de linha de produto com aspectos. As abordagens apresentadas vão desde a utilização de conceitos do paradigma orientada a aspectos apenas para representação dos requisitos não funcionais nas fases de especificação

de requisitos até abordagens mais requintadas que envolvem desde a especificação de resquisitos à programação oirentada a aspectos.

A partir dos estudos e pesquisas sobre o desenvolvimento de LPS e do paradigmas orientado a aspectos, e de várias abordagens que combinam esses dois cenários, foi apresentado uma abordagem para o desenvolvimento de linhas de produtos de software orientado a aspectos (ProLineA).

A abordagem é uma adaptação do processo de gerenciamento de variabilidades (PGV), proposto por Oliveira Junior (2005) e do método para desenvolvimento baseado em componentes e aspectos (DBC/A), proposto por Eler (2006). O exemplo que ilustra a abordagem é uma LPS para sistemas de locação de carros que serviu também para o refinamento da abordagem e a especificação dos modelos de documentação gerados pela abordagem.

A ProLineA é uma abordagem iterativa e incremental, possuindo 5 etapas: especificação dos requsitos e do sistema, elaboração do modelo de caracteríticas, processo de gerenciamento de variabilidades, projeto arquitetural e projeto interno de componentes e finalmente, teste e avaliação da arquitetura. As atividades da etapa de gerenciamento de variabilidades iniciam antes da elaboração do modelo de caracteríticas e segue paralelamente às outras estapas, consumindo e produzindo artefatos no decorrer de todo processo de desnvolvimento.

A avaliação da abordagem foi realizada por meio de um segundo exemplo de aplicação, o mesmo exemplo produzido no inicio dos estudos de LPS, foi desenvolvido novamente com a ProLineA. Alguns parâmetros desejáveis foram estabelecidos antes da aplicação da abordagem e os resultados foram comparados o exemplo inicial.

6.2 Contribuições

A principal contribuição deste trabalho para a área de engenharia de software, para o desenvolvimento de linhas de produto com aspectos, é a abordagem ProLineA. A abordagem possui etapas que vão desde a especificação de requisitos, passando pelo gerenciamento de variabilidades e terminando com os teste e a avaliação da arquitetura, cobrindo assim as atividades do desenvolvimento do núcleo de artefato e do gerenciamento propostos pela *Product Line Pratice* (PLP) propostas pelo SEI (2006).

A abordagem possui um conjunto de notações, artefatos e atividades internas que garantem a rastreabilidade dos aspectos durante todas as fases de desenvolvimento.

Outra contribuição deste trabalho é que a abordagem dá enfoque ao processo gerenciamento de varibialidades, essa é uma das características que não são encontradas na literatura em abordagens propostas para desenvolvimento de linhas de produto orientados a aspectos.

6.3 Trabalhos Futuros

O trabalho apresentado nesta dissertação de mestrado descreveu propostas para apoiar o desenvolvimento de linha de produtos de software e aspectos, no entanto, o trabalho não está completo e deve ser aperfeiçoado por meio de propostas de trabalhos futuros.

O método poderia se aplicado em outros domínios de aplicação, como sistemas distribuídos e sistemas de tempo real. Seria muito interessante também estudos quantitativos e testes para verificar a vantagem de desenvolver linhas de produtos orientados a aspectos.

Seria muito importante também executar testes na implementação e reutilização dos componentes transversais.

Referências Bibliográficas

AKSIT M., WAKITA K., BOSCH J., BERGMANS L., YONEZAWA A. *Abstracting Object Interactions using Composition-Filters*. Springer Verlag: 1994.

ALFERT, Klaus; GMBH, Zühlke. *Requirements, Características and Aspects for Software Product Lines.* Aspects and Software Product Lines: An Early Aspects Workshop at SPLC-Europe 2005. Rennes, France: 2005.

ANTKIEWICZ, M., CZARNECKI, K., *Feature Plugin: Feature Modeling Plug-In for Eclipse.* In: OOPSLA'04 Eclipse Technology eXchange (ETX) Workshop, Oct. 24-28, 2004, Vancouver. Procedings…Vancouver,2004

ATKINSON, C., et al.*Component-based product line engineering with UML*. Addison Wesley, 2000.

ARAGÓN, R.C. *Processo de Desenvolvimento de uma Linha de Produto para Sistemas de Gestão em Bibliotecas*. Dissertação de Mestrado, Mato Grosso: UFMS, 2004.

ARAÚJO, J. F. B; MOREIRA, F. Uma abordagem para instanciação de Métricas para medir documentos de resquisitos orientados a aspectos. In: III Workshop Brasileiro de Desenvolvimento de Software, 2006.

BARROS, M.O.; WERNER, C.M.L; TRAVASSOS, G.H. . *Gerenciamento de Projetos Baseado em Cenários: uma Abordagem de Modelagem Dinâmica e Simulação* . In: I Simpósio de qualidade de software. Rio de Janeiro, 2002.

BATORY, D.; O'MALLEY S.*The design and implementation of hierarchical software systems with reusable components*. ACM Transactions on Software Engineering and Methodology (TOSEM), 1992. 355-398 p.

BATORY, D. , et al. *Design Wizards and Visual Programming Environments for GenVoca Generators*. IEEE Transactions on Software Engineering, 2000. 441-452 p.

BAYER, J.; FLEGE, O.; KNAUBER, P.; LAQUA, R.; MUTHIG, D.; SCHMID, K.; WIDEN, T. *Pulse: A methodology to develop software product lines*. In: Proceedings of the 1999 symposium on Software reusability. Los Angeles, California, USA: ACM Press, 1999. 122–131p.

BOGDAN, R.C.; BIKLEN, S. (1997). *Investigação qualitativa em educação*. Porto: Editora Porto.

BROWN, A. *Large Scale Component Based Development.* Prentice Hall PTR: 2000.

CAGNIN, M. I., BRAGA, R. T. V., PENTEADO, R. D., MALDONADO, J. C., MASIEIRO, P., and GERMANO,F. S. *Empirical studies on framework-based (re)-engineering*. In Workshop LatinoAmericano de Engenharia de Software Experimental, Simp´osio Brasileiro de Engenharia de Software, 2004.

CLARKE, S; BANIASSAD, E.,.*Theme: An approach for aspect-oriented analysis and design.* 26th International Conference on Software Engineering (ICSE'04), Scotland: 2004. 158-167 p.

CHAVEZ, C.; GARCIA, A.; KULESZA, U. *Programação Orientada a Aspectos*. Anais do XVII Simpósio Brasileiro de Engenharia de Software. Manaus: Universidade Federal do Amazonas, 2003.

CHAVEZ, C. *A Model-Driven Approach to Aspect-Oriented Design*.Tese de Doutorado, Rio de Janeiro: PUC-Rio, 2004.

CHEESMAN, J.; DANIELS, J.*UML Components: A Simple Process for Specifying Component-Based Software*. Addison-Wesley 2001.

CLEMENTS, P. C.; NORTHROP, L. *Software Product Lines: Practice and Patterns.* Addison-Wesley: 2001.

CZARNECK, K.;HELSEN, S; EISENECKER, U. *Staged conFiguration using feature models*.In R. L. Nord (ed). Software Product Lines: Third International Conference. Boston, MA, USA: SPLC, 2004.

DIAS, J.M. *Utilizando estudos observacionais para testar e aperfeiçoar um pacote de laboratório para avaliação de ferramentas de mineração visual de dados*. Dissertação de Mestrado, 2006. Universidade Salvador. Disponível em: http://tede.unifacs.br/tde_busca/arquivo.php?codArquivo=69. Acessado em 29 de outubro de 2008.

D'SOUZA, D.; WILLS, A. *Objects, Components, and Frameworks with UML - The Catalysis Approach*. Addison-Wesley:1999.

ELER, M. *Um método para o desenvolvimento baseado em components e aspectos*. Dissertação de Mestrado. São Carlos, SP : ICMC/USP, 2006.

ELRAD, T.; KICZALES, G.; AKSIT, M.; LIEBERHER, k.; Ossher, H. *Discussing Aspects of AOP*. . Communications of the ACM, v. 44, n. 10, p. 33-38, 2001.

ESTEARBOOK, S ; ARANDA, J. *Case Studies for Software Engineers*. In: 28th International Conference on Software Engineering – 28th ICSE. Shangai: 2006.

FILMAN,R.; FRIEDMAN, D. *Aspect-oriented programming is quantication and obliviousness*. In OOPSLA 2000 Workshop on Advanced Separation of Concerns, Minneapolis, MN: 2000.

FILMAN, R., ELRAD,T.; CLARKE, T.; AKSIT, S. *Aspect-Oriented Software Development*. Addison-Wesley, 2005.

FORBECK, V.L.A. *Novas tecnologias e comunicação e sua apropriação por parte de alunos e professores.* Dissertação de mestrado. Unicaieiras. Disponível em: www.unicaieiras.com.br/ProducaoADM/Docente/artigo_mestrado_em_educacao.pdf. Acessado em 30 de outrubro de 2008.

GAMMA, E.; HELM, R.; JOHNSON, R.; VLISSIDES, J. *Padrões de Projeto: Soluções Reutilizáveis de Software Orientado a Objetos*. Porto Alegre: Ed. Bookman, 2000.

GARCIA, V.; LUCRÉDIO, D.; PINTO L.;ÁLVARO A.; ALMEIDA E.; PRADO F. A. *Uma Ferramenta Case para o Processo Orientado a Aspectos*. Brasília: Simpósio Brasileiro de Enegenharia de Sdoftware, 2004. 11-12p.

GARCIA, A.; SANT'ANNA, C.; CHAVEZ, C.; SILVA, V.; LUCENA, C.; STAA, A. *Separation of Concerns in Multi-Agent Systems: An Empirical Study*. In: C. Lucena et al (Eds)., Software Engineering for Multi-Agent Systems II. Springer-Verlag, LNCS 2940: 2004. 49-72 p.

GIMENES, I. M. S.; TRAVASSOS, G. H. *O enfoque de linha de produto para desenvolvimento de software*. In: Jornada de atualização em informática da SBC. Florianópolis: 2002. 34 p.

GIMENES, I. M. S.; HUZITA, E. H. M. *Desenvolvimento Baseado em Componente*.Maringá: Editora Ciência Moderna, 2005.

GMVISIT. GMVisit - *Algorithms, Algorithm Modeling,Software, & Software Architecture.* Disponível em: http://www.eecs.umich.edu/courses/eecs486/win03/notes/GMVisit.pdf. Acessado em novembro de 2008.

GOMAA, H. *Reusable software engineering: Concepts and research directions; tutorial in software reusability*. In: Workshop on Reusable Software. Newport, EUA, Proceeding Newport: ITT, 1993.

GOMAA, H. *Designing software product lines with UML: from use cases to pattern-based software architecture*s. Boston: Addison-Wesley, 2005.

GRISS, M; FAVARO, J. D´ALESSANDRO, M. *Integrating Feature Modeling with rseb.I* In: International Conference on Software Reuse. Canada: ACM/IEEE, 1998. 76-85p.

HALL OF FAME. Disponível em http://www.sei.cmu.edu/productlines/plp_hof.html. Acessado em novembro de 2008.

HARRISON W. OSSHER H. *Subject-oriented programming (a critique of pure objects)*. In proc. Object-Oriented Programming Systems Languages and Applications (OOPSLA), 1993. 411- 428 p.

HEINEMAN, G. T.; COUNCILL, W. T. *Definition of a Software Component and its Elements*, Boston, MA: Addison-Wesley, 2001.

HETRICK, W., Moore, J. and Krueger, C. *Incremental Return on Incremental Investment: Engenio's Transition to Software Product Line Practice*. OOPSLA Proceedings 2006. Portland, Oregon, 2006. 798-804 p.

HEYMANS, P.; TRIGAUX, J. C. Software *product line: state of the art*. Technical report for PLENTY project, Institut d'Informatique FUNDP, Namur, 2003.

HERZUM, P.; SIMS, O. *Business Component Factory: A Comprehensive Overview of Component-Based Development for the Enterprise*. OMG Press: 2000.

JACOBSON, L. ; NG, P.W.*Aspect Oriented Software Development with Use Case (Addison-Wesley object technology series).* Addison Wesley Professional: 2004.

JUDE. http://jude.change-vision.com/jude-web/product/community.html. Acessado em 10 de agosto de 2008.

KANG, K., Feature-oriented domain analysis (FODA) - feasibility study. Technical Report CMU/SEI-90-TR-21, SEI/CMU, Pittsburgh, 1990.

KANG, K.; KIM, S.; KIM, K.; KIM, G.; SHIN, E. FORM: A Feature-Oriented Reuse Method with Domain-Specific Refrence Architecture, SEI Technical Report, 1998.

KITCHENHAM, B.A.; PFLEEGER, S.L.; PICKARD, L.M.; JONES, P.W.; HOAGLIN, D.C.; EL EMAN, K.; ROSENBERG, J. *Preliminary guidelines for empirical research in software engineering*. Software Engineering, IEEE Transactions on Volume 28, Issue 8, 2002.

KIZALES,G., LAMPING, J., MENDHEKAR, A., MAEDA,C., LOPES, C., LOINGTIER, J. e IRWIN, J. *Aspect-Oriented Programming*. Local : 1997.

KRUEGER, C. W. *Software Reuse*. ACM Computing Surveys, Vol. 24, No. 02, 1992. 131-183 p.

LIEBERHERR,K.J.;SILVA, I.; XIAU, C. *Adaptive object-oriented programming using graph-based customization*. Communications of the ACM, 37(5):94:101. Local: 1994.

LOBO, A.E.C, BRITO, P. H. S., RUBIRA C. E. F.*Gerenciamento de Variabilidade em Linha de Produto de Software Baseado em Componente*. Recife: VI Workshop de Desenvolvimento Baseado em Componentes, 2006.

MUSTARO, P.N; SILVEIRA, I.F. *A formação profissional em computação por meio da alfabetização científica: um estudo de caso*. In: WEI – Workshop sobre educação em computação. Belém do Pará: Anais do XXVIII Congresso da Sociedade Brasileira de Computação. Belém do Pará, 2008.

NYBEN, A.; TYSZBEROWICZ, S.; WEILLER, T. *Are Aspects useful for Managing Variability in Software Product Lines? A Case Study*. Aspects and Software Product Lines: An Early Aspects Workshop. Rennes, France: Software Product Line Conference, 2005.

OLIVEIRA JUNIOR, E. *Um processo de gerenciamento de variabilidade para linha de produto de software*. Dissertação de Mestrado. Maringá: UEM, 2005.

OSSHER, H., TARR, P. *Using subject-oriented programming to overcome common problems in object-oriented software development/evolution*. In: International Conference on Software Engineering: 688-698. IEEE Computer Society Press. 1999.

PACIOS, S.F. *Uma abordagem Orientada a Aspectos para o desenvolvimento de linhas de produto de software*. Dissertação de Mestrado. Instituto de Ciências Matemática e Computação : USP, 2006.

PAWLAK, L.; DUCHIEN, G.; FLORIN, F.; LEGOND-AUBRY, L.; SEINTURIER, L.; MARTELLI, L. *A UML notation for aspect-oriented software design*. In Proceedings of the 1st International Conference on Aspect-Oriented Software Development (AOSD'2002). ACM Press, 2002.

TRIGAUX, J. C.; HEYMANS, P. *Modelling variability requirements in software product lines: a comparative survey*. Technical report PLENTY project. Namur: Institut d'Informatique/ FUNDP, 2003.

SEI - SOFTWARE ENGINEERING INSTITUTE (2004). *A framework for software product line practice 4.2*. Disponível em <http://www.sei.cmu.edu/plp/framework.html>. Acesso em: 10 de setembro de 2006.

STRADA, F; UMANN, F; ALBANO, C.S. *A utilização do teleeduc como recurso de tecnologia para apoio às atividades de ensino. Revista linha virtual*. Disponível em: http://www.nead.uncnet.br/2007/revistas/ead/5/5.pdf. Acessado em 29 de outubro de 2008.

STEGER, M.; TISCHER, C.; BOSS, B.; MULLER, A.; PERTLER, O.; STOLZ, W.; FERBER, S. *Introducing PLA at Bosch Gasoline Systems: Experiences and Practices*. Nord, R. (ed.), *Proceedings SPLC3*, Lecture Notes in Computer Science 0302-9743, vol. 3154. Boston: Springer, 2004. 34-50p.

SVEN, A. *Aspect Refinement.* Technical Report 10. Germany : University of Magdeburg/Department of Computer Science, 2006.

SOARES, S.; LAUREANO, E.; BORBA, P. *Implementing distribution and persistence aspects with AspectJ*. In: 17th ACM Conference on Object-Oriented programming systems, languages, and applications. Washington: ACM Press, 2002. 174-190p.

SOCHOS, P.; PHILIPPOW, I.; RIEBISCH, M. Feature-oriented development of software product lines: mapping feature models to the architecture. Springer, LNCS 3263, p. 138-152. , 2004

SUZUKI,Y.; YAMAMOTO, Y. *Extending UML with aspects: Aspect support in the design phase*. In Proceedings of 13rd the European Conference Object-Oriented Programming (ECOOP'99). Londres: International Workshop on Aspect-Oriented Programming, 1999. 299-300p.

YIN, R. *Estudo de caso: planejamento e métodos*. 2a ed. Porto Alegre: Bookman; 2001.

VAN GURP, J.; BOSCH, J. : SVAHNBERG, M. *On the Notion of Variability in Software Product Lines.* Washington: Working IEEE/IFIP Conference on Software Architecture (WISCA'01), 2001. 45p.

WEISS, D.; CHI TAU, R. L. *Software product-line engineering: a family-based softwaredevelopment process.* Boston: Addison-Wesley, 1999.

Apêndice A

Documento de Requisitos da Linha de Produto para Sistemas de Gestão de Processos Seletivos (LP-GPS)

Este documento de requisitos é a especificação oficial dos requisitos da Linha de Produto para Gestão de Processos Seletivos

As seções seguintes descrevem todos serviços e funcionalidades que o LP-GPS devem prover; restrições; informações sobre o domínio da aplicação, bem como restrições no processo usado para desenvolver o sistema".

A1. Descrição do Domínio

A LP-GPS e uma linha de produto de sistemas especializados em gestão de concursos e testes seletivos. O objetivo desses tipos de sistemas é atender com qualidade as demandas de instituições públicas e privadas nessa área, oferecendo uma forma sistemática de avaliação diferenciada, centrada no gerenciamento de inscrições, provas, resultados, convocações, entre outras atividades relacionadas. Outro objetivo é atender às necessidades de seleção de profissionais do setor público e privado.

Os produtos gerados por essa LPS devem oferecer serviços personalizado para aplicação, elaboração e realização das provas.

A2. Requisitos do Domínio.

1. Gerenciar o processo de inscrição. Incluindo operações como incluir, alterar, e cancelar uma inscrição.

2. Controlar pagamento, descontos, e formas de pagamento. Pode ser realizado em conjunto com sistemas bancários e online.
3. candidato poderá solicitar isenção de taxas de inscrição, para isso dever ser disponibilizado um formulário online, o sistema deve ser capaz de processar as informações sobre as documentações entregues pelo usuário.
4. Gerenciar o processo de inscrição para portadores de necessidades especiais.
5. Gerenciar o processo de inscrição para afro-descendentes.
6. Gerenciar o processo de inscrição para indígenas.
7. No ato da inscrição o usuário poderá optar por utilizar o resultado do ENEM.
8. Gerenciar processo de homologação. O sistema deve verificar se pagamento foi efetuado e se a documentação necessária foi entregue.
9. sistema deve permitir que materiais, resultados, editais e comunicados sejam disponibilizados online e impressa.
10. sistema deve permitir o gerenciamento das provas e resultados por fase.
11. sistema de gerenciar os resultados, de todas as fases do processo seletivo.
12. Gerenciar Convocação de Candidatos
13. Gerenciar Recursos

A3. Requisitos do Domínio X Casos de uso

A Tabela A.1 mostra a relação entre os requisitos do domínio e os casos de uso identificados, a terceira coluna mostra quais casos de uso serão cobertos pelo diagrama de casos de uso.

Requisito	NCtrl	Caso de uso	DCU
Req.1, Req.4, Req.5, Req.6.	1	alterarInscrição	
Req. 1, Req.4, Req.5, Req.6.	2	cancelarInscrição	
Req. 1, Req.4, Req.5, Req.6	3	fazerInscrição	*
Req.5, Req.4, Req.6	4	fazerInscEspecial	*
Req.5	5	fazerInscAfro	*
Req.4	6	fazerInscEsp	*
Req.6	7	fazerInscInd	*
Req. 1, Req.4, Req.5, Req.6	8	utilizarEnem	*
Req. 1, Req.4, Req.5, Req.6	9	preencherFormTitulos	*
Req. 1, Req.4, Req.5, Req.6	10	preencherFormSocioEdu	*
Req.2	11	pagarTaxaInsc	*
Req.2	12	pagarOnline	*
Req.2	13	pagarBoleto	*
Req.2	14	pagarCartão	*
Req.3, Req.1, Req.4, Req.5, Req.6	15	solicitarIsenção	
Req.8	16	controlarHomo	
Req.8	17	imprimirCartãoHomo	*
Req.8	18	segundaViaCartão	
Req.9	19	publicarEditais	*
Req.9	20	publicarResultados	*
Req.9	21	publicarResTotal	*
Req.9	22	publicarResTotalDow	
Req.9	23	publicarResInd	*
Req.9	24	publicarManual	*
Req.9	25	publicarManualOn	
Req.9	26	publicarManualDow	*
Req.9	27	publicarGabaritos	*
Req.9	28	publicarProvas	*
Req.10, Req. 11	29	gerarRePrObjetivas	*
Req.10, Req. 11	30	gerarRePrDiscursivas	*
Req.10, Req. 11	31	gerarRePrPráticas	*
Req.10, Req. 11	32	gerarRePrDidática	*
Req.10, Req. 11	33	gerarReEntrevista	*
Req.10, Req. 11	34	gerarRePrTitulos	*
Req.12	35	convocarCandExame	
Req.12	36	convocarCandNomear	
Req.12	37	convocarCandMatrícula	
Req.12	38	convocarCandFase	
Req.12	39	imprimirCartãoFase	
Req.13	40	entrarRecurso	*
Req.13	41	pagarTaxaRecurso	*
Req.13	42	entrarRePrTitulo	*
Req.13	43	entrarRePrObjetiva	*
Req.13	44	entrarRePrJurídica	*
Req.13	45	entrarRePrDiscursiva	*
Req.13	46	entrarReHomoInscrição	*
Req.10, Req.11	47	gerarResProvas	*
Req.9	48	publicarOnline	*
Req.9	49	publicarDownload	*

Tabela A.1 – Requisitos do sistema x Casos de uso

A4. Atores e papéis

Ator	Descrição	Responsabilidades
Administrador	È o administrador do sistema e o responsável pela manutenção e gerenciamento de usuários	Gerenciar os usuários
Candidatos	Candidatos que utilizarão o sistema para fazer inscrição e acompanhar o processo seletivo	Prove informações necessárias para que o sistema efetue a inscrição do candidato.
Gerente do processo seletivo	É o represente da comissão responsável pelo processo seletivo.	Manter o sistema atualizado. Disponibilizar materiais e resultados. Disponibilizar dados para relatórios.
Sistema de correção	É o sistema, ou pessoas responsáveis por fornecer os resultados de todas as etapas e fases do processo seletivo..	Fornecer resultados; Emitir dados estatísticos.
Sistema Bancário	É o sistema que disponibilizará as informações sobre pagamentos efetuados.	Fornecer pagamentos efetuados.

Tabela A.2 – Requisitos do sistema x Casos de uso

A5. Descrição dos casos de uso

Caso de uso 1: alterarInscrição
Tipo: opcional
Objetivo: alterar dados da inscrição
Fluxo obrigatório
1. Candidato preenche formulário inscrição 2. Candidato altera dados já cadastrados 3. Sistema efetua alteração 4. Candidato confirma alteração 5. Sistema registra alteração no BD
Fluxo alternativo
1. Candidato preenche formulário inscrição. 2. Candidato altera dados já cadastrados 3. Sistema efetua alteração 4. Candidato não confirma alteração 5. Sistema não registra alteração no BD 6. Sistema ativa outro caso de uso

Caso de uso 2 : cancelarInscrição
Tipo: opcional
Objetivo: cancelar inscrição
Fluxo obrigatório
1. Candidato efetua inscrição 2. Candidato solicita cancelamento de inscrição. 3. Candidato preenche formulário 4. Sistema solicita confirmação 5. Candidato confirma cancelamento
6. Sistema efetuar cancelamento e atualiza BD.
Fluxo alternativo
1. Candidato preenche formulário inscrição. 2. Candidato altera dados já cadastrados 3. Sistema efetua alteração 4. Candidato não confirma alteração 5. Sistema não registra alteração no BD 6. Sistema ativa outro caso de uso

Caso de uso 3: fazerInscrição
Tipo: obrigatorio
Objetivo: fazer inscrição do candidato para um processo seletivo
Fluxo obrigatório
1. Candidato inicia processo de inscrição. 2. Candidato preenche formulário de incrição. 3. Sistema atualiza BD. 4. Sistema ativa outro caso de uso.
Fluxo alternativo
1. Candidato inicia processo de inscrição. 2. Candidato preenche formulário de incrição. 3. Candidato opta por utilizar resultado do Enem. 4. Sistema atualiza BD. 5. Sistema ativa outro caso de uso.
1. Candidato inicia processo de inscrição 2. Candidato solicita inscrição Especial 3. Sistema atualiza BD

4. Sistema ativa outro caso de uso.
Caso de uso 4: fazerInscEspecial
Tipo: opcional
Objetivo: fazer inscrição com atendimento especial para candidatos portadores de necessidades especial, afro-descendentes ou indígenas.
Fluxo obrigatório
1. Candidato inicia processo de inscrição. 2. Candidato opta pelo tipo de inscrição especial. 3. Sistema ativa outro caso de uso

Caso de uso 5: fazerInscAfro
Tipo: obrigatorio
Objetivo: fazer inscrição do candidato para um processo seletivo
Fluxo obrigatório
1. Candidato inicia processo de inscrição. 2. Candidato preenche formulário de incrição. 3. Sistema atualiza BD. 4. Sistema ativa outro caso de uso.
Fluxo alternativo
1. Candidato inicia processo de inscrição. 2. Candidato preenche formulário de incrição. 3. Candidato opta por utilizar resultado do Enem. 4. Sistema atualiza BD. 5. Sistema ativa outro caso de uso.
1. Candidato inicia processo de inscrição 2. Candidato solicita inscrição Especial 3. Sistema atualiza BD 4. Sistema ativa outro caso de uso.

Caso de uso 6: fazerInscEsp
Tipo: obrigatório
Objetivo: fazer inscrição do candidato para um processo seletivo
Fluxo obrigatório
1. Candidato inicia processo de inscrição. 2. Candidato preenche formulário de incrição. 3. Sistema atualiza BD. 4. Sistema ativa outro caso de uso.
Fluxo alternativo
1. Candidato inicia processo de inscrição. 2. Candidato preenche formulário de incrição. 3. Candidato opta por utilizar resultado do Enem. 4. Sistema atualiza BD. 5. Sistema ativa outro caso de uso.
1. Candidato inicia processo de inscrição 2. Candidato solicita inscrição Especial 3. Sistema atualiza BD 4. Sistema ativa outro caso de uso.

Caso de uso 7: fazerInscInd
Tipo: obrigatório
Objetivo: fazer inscrição do candidato para um processo seletivo
Fluxo obrigatório
1. Candidato inicia processo de inscrição.
2. Candidato preenche formulário de incrição. 3. Sistema atualiza BD. 4. Sistema ativa outro caso de uso.
Fluxo alternativo
1. Candidato inicia processo de inscrição. 2. Candidato preenche formulário de incrição. 3. Candidato opta por utilizar resultado do Enem. 4. Sistema atualiza BD. 5. Sistema ativa outro caso de uso.
1. Candidato inicia processo de inscrição 2. Candidato solicita inscrição Especial 3. Sistema atualiza BD 4. Sistema ativa outro caso de uso.

Caso de uso 8: utilizarEnem
Tipo: obrigatório
Objetivo: fazer inscrição do candidato permitindo que ele utilize os resultados do Enem para uma primeira seleção
Fluxo obrigatório
1. Candidato faz opção por utilizar resultado do ENEM 2. Candidato entra com os dados do ENEM 3. Sistema atualiza BD. 4. Sistema ativa outro caso de uso.
Fluxo alternativo
1. Candidato faz opção por utilizar resultado do ENEM 2. Candidato cancela operação 3. Sistema atualiza BD. 4. Sistema ativa outro caso de uso.

Caso de uso 9: preencherFormTítulos
Tipo: opcional
Objetivo: candidato preenche o formulário para entrega do material da prova de títulos.
Fluxo obrigatório
1. Candidato preenche formulário. 2. Candidato imprime formulário preenchido. 3. Sistema atualiza BD. 4. Sistema ativa outro caso de uso.
Fluxo alternativo
1. Candidato preenche formulário. 2. Candidato cancela operação 3. Sistema atualiza BD. 4. Sistema ativa outro caso de uso.

Caso de uso10: preencherFichaSócioEdu
Tipo: obrigatória
Objetivo: candidato preenche o formulário para entrega do material da prova de títulos.
Fluxo obrigatório
1. Candidato preenche ficha sócio educacional. 2. Candidato confirma envio dos dados. 3. Sistema atualiza BD. 4. Sistema ativa outro caso de uso.
Fluxo alternativo
1. Candidato preenche ficha sócio educacional.

2. Candidato cancela operação 3. Sistema atualiza BD. 4. Sistema ativa outro caso de uso.
1. Candidato preenche ficha sócio educacional. 2. Candidato deseja alterar dados da ficha. 3. Sistema recupera dados. 4. Candidato altera dados. 5. Candidato confirma alteração. 6. Sistemas atualiza BD. 7. Sistema ativa outro caso de uso.

Caso de uso 11: pagarTaxaInsc
Tipo: opcional
Objetivo: candidato seleciona forma de pagamento.
Fluxo obrigatório
1. Candidato opta pelo tipo de pagamento. 2. Candidato confirma envio dos dados. 3. Sistema atualiza BD. 4. Sistema ativa outro caso de uso.
Fluxo alternativo
1. Candidato opta pela forma de pagamento. 2. Candidato cancela operação 3. Sistema atualiza BD. 4. Sistema ativa outro caso de uso.

Caso de uso 12: pagarOnline
Tipo: opcional
Objetivo: candidato efetua pagamento online.
Fluxo obrigatório
1. Candidato entra com os dados do pagamento. 2. Sistema ativa interface com sistema bancário. 3. Sistema bancário envia confirmação de pagamento. 4. Sistema atualiza BD. 5. Sistema ativa outro caso de uso.
Fluxo alternativo
1. Candidato entra com os dados do pagamento. 2. Sistema ativa interface com sistema bancário. 3. Comunicação com sistema bancário falha. 4. Sistema envia mensagem para candidato tentar mais tarde. 5. Sistema ativa outro caso de uso.
1. Candidato entra com os dados do pagamento. 2. Sistema ativa interface com sistema bancário. 3. Comunicação com sistema bancário falha. 4. Sistema envia mensagem para candidato tentar mais tarde. 5. Candidato opta por outra forma de pagamento. 6. Sistema ativa outro caso de uso.

Caso de uso 13: pagarBoleto
Tipo: opcional
Objetivo: candidato imprime boleto para pagamento.
Fluxo obrigatório
1. Candidato entra com os dados do pagamento. 2. Sistema gera boleto. 3. Candidato imprime o boleto.
4. Sistema atualiza BD. 5. Sistema ativa outro caso de uso.
Fluxo alternativo
1. Candidato entra com os dados do pagamento. 2. Candidato cancela operação. 3. Sistema atualiza BD. 4. Sistema ativa outro caso de uso.

Caso de uso 14: pagarOnline
Tipo: opcional
Objetivo: candidato efetua pagamento online.
Fluxo obrigatório
1. Candidato entra com os dados do pagamento. 2. Sistema ativa interface com sistema bancário. 3. Sistema bancário envia confirmação de pagamento. 4. Sistema atualiza BD. 5. Sistema ativa outro caso de uso.
Fluxo alternativo
1. Candidato entra com os dados do pagamento. 2. Sistema ativa interface com sistema bancário. 3. Comunicação com sistema bancário falha. 4. Sistema envia mensagem para candidato tentar mais tarde. 5. Sistema ativa outro caso de uso.
6. Candidato entra com os dados do pagamento. 7. Sistema ativa interface com sistema bancário. 8. Comunicação com sistema bancário falha. 9. Sistema envia mensagem para candidato tentar mais tarde. 10. Candidato opta por outra forma de pagamento. 11. Sistema ativa outro caso de uso.

Caso de uso 15: solicitarIsenção
Tipo: opcional
Objetivo: candidato solicita isenção de taxas de inscrição.
Fluxo obrigatório
1. Candidato preenche formulário de isenção de taxa de inscrição. 2. Sistema atualiza BD. 3. Candidato preenche formulário preenchido, juntamente com as instruções dos materiais a serem anexados. 4. Sistema ativa outro caso de uso.
Fluxo alternativo
1. Candidato preenche formulário de isenção de taxa de inscrição. 2. Candidato cancela operação. 3. Sistema ativa outro caso de uso.

Caso de uso 16: controlarHomo
Tipo: obrigatório
Objetivo: coordenador do processo seletivo, gera homologação dos candidatos
Fluxo obrigatório
1. Candidato entra com os dados de pagamento por boleto.

2. Sistema ativa interface com sistema bancário. 3. Sistema verifica pagamento por cartão e online. 4. Sistema gera homologação 5. Sistema atualiza BD. 6. Sistema ativa outro caso de uso.
Fluxo alternativo
1. Candidato entra com os dados de pagamento por boleto. 2. Sistema ativa interface com sistema bancário. 3. Sistema verifica pagamento por cartão e online. 4. Coordenador cancela operação 5. Sistema ativa outro caso de uso.

Caso de uso 17: imprimirCartãoHomo
Tipo: obrigatório
Objetivo: candidato imprime cartão de homologação.
Fluxo obrigatório
7. Candidato entra com os dados para imprimir cartão de homologação. 8. Sistema confirma homologação e gera o cartão a ser impresso. 9. Sistema atualiza BD. 10. Sistema ativa outro caso de uso.
Fluxo alternativo
1. Candidato entra com os dados para imprimir cartão de homologação. 2. Sistema não confirma homologação. 3. Sistema envia mensagem para usuário. 4. Sistema atualiza BD. 5. Sistema ativa outro caso de uso.

Caso de uso 18: segundaViaCartão
Tipo: opcional
Objetivo: candidato imprime segunda via do cartão
Fluxo obrigatório
1. Candidato entra com os dados para imprimir segunda via do cartão. 2. Sistema verifica consistência dos dados. 3. Sistema atualiza BD. 4. Sistema ativa outro caso de uso.
Fluxo alternativo
1. Candidato entra com os dados para imprimir segunda via do cartão. 2. Candidato cancela operação. 3. Sistema atualiza BD. 4. Sistema ativa outro caso de uso.

Caso de uso 19: publicarEditais
Tipo: opcional
Objetivo: coordenador publica editais do processo seletivo
Fluxo obrigatório
1. coordenador publica edital online do processo seletivo 2. Sistema verifica consistência dos dados. 3. Sistema atualiza BD. 4. Sistema ativa outro caso de uso.

Fluxo alternativo
1. Coordenador publica edital online do processo seletivo 2. Coordenador cancela operação. 3. Sistema atualiza BD. 4. Sistema ativa outro caso de uso.

Caso de uso 20: publicarResultados
Tipo: opcional
Objetivo: coordenador publica Resultados do processo seletivo
Fluxo obrigatório
1. coordenador publica resultados. 2. Sistema verifica consistência dos dados. 3. Sistema atualiza BD. 4. Sistema ativa outro caso de uso.
Fluxo alternativo
1. Coordenador publica resultados online do processo seletivo 2. Coordenador cancela operação. 3. Sistema atualiza BD. 4. Sistema ativa outro caso de uso.

Caso de uso 21: publicarResTotal
Tipo: opcional
Objetivo: coordenador publica Resultados do processo seletivo
Fluxo obrigatório
1. coordenador publica resultados online do processo seletivo 2. Sistema verifica consistência dos dados. 3. Sistema atualiza BD. 4. Sistema ativa outro caso de uso.
Fluxo alternativo
1. Coordenador publica resultados online do processo seletivo 2. Coordenador cancela operação. 3. Sistema atualiza BD. 4. Sistema ativa outro caso de uso.

Caso de uso 22: publicarResTotalDown
Tipo: opcional
Objetivo: coordenador publica Resultados do processo seletivo
Fluxo obrigatório
1. coordenador publica resultados total para download do processo seletivo 2. Sistema verifica consistência dos dados. 3. Sistema atualiza BD. 4. Sistema ativa outro caso de uso.
Fluxo alternativo
1. Coordenador publica resultados total para download do processo seletivo 2. Coordenador cancela operação. 3. Sistema atualiza BD. 4. Sistema ativa outro caso de uso.

Caso de uso 23: publicarResInd
Tipo: opcional

Objetivo: coordenador publica Resultados do processo seletivo
Fluxo obrigatório
1. coordenador publica resultados online individuais do processo seletivo 2. Sistema verifica consistência dos dados. 3. Sistema atualiza BD. 4. Sistema ativa outro caso de uso.
Fluxo alternativo
1. Coordenador publica resultados online individuais do processo seletivo 2. Coordenador cancela operação. 3. Sistema atualiza BD. 4. Sistema ativa outro caso de uso.

Caso de uso 24: publicarManual
Tipo: opcional
Objetivo: coordenador publica Manual do Candidato
Fluxo obrigatório
1. coordenador publica manual do candidato 2. Sistema verifica consistência dos dados. 3. Sistema atualiza BD. 4. Sistema ativa outro caso de uso.
Fluxo alternativo
1. Coordenador publica manual do candidato 2. Coordenador cancela operação. 3. Sistema atualiza BD. 4. Sistema ativa outro caso de uso.

Caso de uso 25: publicarManualOn
Tipo: opcional
Objetivo: coordenador publica Manual do Candidato Online
Fluxo obrigatório
1. coordenador publica manual do candidato Online 2. Sistema verifica consistência dos dados. 3. Sistema atualiza BD. 4. Sistema ativa outro caso de uso.
Fluxo alternativo
1. Coordenador publica manual do candidato Online 2. Coordenador cancela operação. 3. Sistema atualiza BD. 4. Sistema ativa outro caso de uso.

Caso de uso 26: publicarManualDow
Tipo: opcional
Objetivo: coordenador publica Manual do Candidato para download
Fluxo obrigatório
1. coordenador publica manual do candidato para dowload 2. Sistema verifica consistência dos dados. 3. Sistema atualiza BD. 4. Sistema ativa outro caso de uso.
Fluxo alternativo
1. Coordenador publica manual do candidato para download 2. Coordenador cancela operação. 3. Sistema atualiza BD. 4. Sistema ativa outro caso de uso.

Caso de uso 27: publicarGabaritos
Tipo: opcional
Objetivo: coordenador publica gabaritos das provas
Fluxo obrigatório
1. coordenador publica gabarito das provas 2. Sistema verifica consistência dos dados. 3. Sistema atualiza BD. 4. Sistema ativa outro caso de uso.
Fluxo alternativo
1. Coordenador publica gabarito provas 2. Coordenador cancela operação. 3. Sistema atualiza BD. 4. Sistema ativa outro caso de uso.

Caso de uso 28: publicarProvas
Tipo: opcional
Objetivo: coordenador publica provas
Fluxo obrigatório
1. coordenador publica provas 2. Sistema verifica consistência dos dados. 3. Sistema atualiza BD. 4. Sistema ativa outro caso de uso.
Fluxo alternativo
1. Coordenador publica provas 2. Coordenador cancela operação. 3. Sistema atualiza BD. 4. Sistema ativa outro caso de uso.

Caso de uso 29: gerarResPrObjetivas
Tipo: opcional
Objetivo: coordenador gera resultados das provas objetivas
Fluxo obrigatório
1. Coordenador entra com os dados da prova objetiva. 2. Sistema ativa sistema de correção de gabaritos. 3. Sistema verifica consistência dos dados. 4. Sistema atualiza BD. 5. Sistema gera resultados. 6. Sistema ativa outro caso de uso.
Fluxo alternativo
1. Coordenador entra com os dados da prova objetiva. 2. Sistema ativa sistema de correção de gabaritos. 3. Sistema verifica consistência dos dados. 4. Sistema encontra dados inconsistentes. 5. Sistema envia mensagens para o usuário. 6. Sistema ativa outro caso de uso.
Caso de uso 30: gerarResPrDiscursivas
Tipo: opcional
Objetivo: coordenador gera resultados das provas discursivas

Fluxo obrigatório
1. Coordenador entra com os dados da prova discursiva 2. Sistema atualiza BD. 3. Sistema gera resultados. 4. Sistema ativa outro caso de uso.
Fluxo alternativo
1. Coordenador entra com os dados da prova discursiva 2. Sistema atualiza BD. 3. Coordenador cancela operação. 4. Sistema gera resultados. 5. Sistema ativa outro caso de uso.

Caso de uso 31: gerarResPrPrática
Tipo: opcional
Objetivo: coordenador gera resultados das provas práticas
Fluxo obrigatório
1. Coordenador entra com os dados da prova prática. 2. Sistema atualiza BD. 3. Sistema gera resultados. 4. Sistema ativa outro caso de uso.
Fluxo alternativo
1. Coordenador entra com os dados da prova prática. 2. Sistema atualiza BD. 3. Coordenador cancela operação. 4. Sistema gera resultados. 5. Sistema ativa outro caso de uso.

Caso de uso 32: gerarResPrDidática
Tipo: opcional
Objetivo: coordenador gera resultados das provas didática.
Fluxo obrigatório
1. Coordenador entra com os dados da prova didática. 2. Sistema atualiza BD. 3. Sistema gera resultados. 4. Sistema ativa outro caso de uso.
Fluxo alternativo
1. Coordenador entra com os dados da prova didática 2. Sistema atualiza BD. 3. Coordenador cancela operação. 4. Sistema gera resultados. 5. Sistema ativa outro caso de uso.

Caso de uso 33: gerarResEntrevista
Tipo: opcional
Objetivo: coordenador gera resultados da entrevista
Fluxo obrigatório
1. Coordenador entra com os dados da entrevista. 2. Sistema atualiza BD. 3. Sistema gera resultados. 4. Sistema ativa outro caso de uso.

Fluxo alternativo
1. Coordenador entra com os dados da entrevista 2. Sistema atualiza BD. 3. Coordenador cancela operação. 4. Sistema gera resultados. 5. Sistema ativa outro caso de uso.

Caso de uso 34: gerarResPrTítulos
Tipo: opcional
Objetivo: coordenador gera resultados das provas de títulos
Fluxo obrigatório
1. Coordenador entra com os dados da prova de títulos 2. Sistema atualiza BD. 3. Sistema gera resultados. 4. Sistema ativa outro caso de uso.
Fluxo alternativo
1. Coordenador entra com os dados da prova de títulos 2. Sistema atualiza BD. 3. Coordenador cancela operação. 4. Sistema gera resultados. 5. Sistema ativa outro caso de uso.

Caso de uso 35: convocarCandExames
Tipo: opcional
Objetivo: coordenador convoca candidatos para exame médico
Fluxo obrigatório
1. Coordenador entra com os dados da convocação 2. Sistema atualiza BD. 3. Sistema gera resultados. 4. Sistema ativa outro caso de uso.
Fluxo alternativo
1. Coordenador entra com os dados da convocação. 2. Sistema atualiza BD. 3. Coordenador cancela operação. 4. Sistema gera resultados. 5. Sistema ativa outro caso de uso.

Caso de uso 36: convocarCandNomear
Tipo: opcional
Objetivo: coordenador convoca candidatos para exame médico
Fluxo obrigatório
1. Coordenador entra com os dados da convocação 2. Sistema atualiza BD. 3. Sistema gera resultados. 4. Sistema ativa outro caso de uso.
Fluxo alternativo
1. Coordenador entra com os dados da convocação. 2. Sistema atualiza BD. 3. Coordenador cancela operação. 4. Sistema gera resultados.

Caso de uso 40: entrarRecurso
Tipo: opcional
Objetivo: candidato escolhe tipo de recurso
Fluxo obrigatório
1. Candidatos escolhe tipo de recurso 2. Sistema atualiza BD. 3. Sistema ativa outro caso de uso.
Fluxo alternativo
1. Candidatos escolhe tipo de recurso 2. Candidato cancela operação 3. Sistema ativa outro caso de uso.
5. Sistema ativa outro caso de uso.

Caso de uso 37: convocarCandMatrícula
Tipo: opcional
Objetivo: coordenador convoca candidatos para matrícula
Fluxo obrigatório
1. Coordenador entra com os dados da convocação 2. Sistema atualiza BD. 3. Sistema gera resultados. 4. Sistema ativa outro caso de uso.
Fluxo alternativo
1. Coordenador entra com os dados da convocação. 2. Sistema atualiza BD. 3. Coordenador cancela operação. 4. Sistema gera resultados. 5. Sistema ativa outro caso de uso.

Caso de uso 38: convocarCandFase
Tipo: opcional
Objetivo: coordenador convoca candidatos para matrícula
Fluxo obrigatório
1. Coordenador entra com os dados da convocação 2. Sistema atualiza BD. 3. Sistema gera resultados. 4. Sistema ativa outro caso de uso.
Fluxo alternativo
1. Coordenador entra com os dados da convocação. 2. Sistema atualiza BD. 3. Coordenador cancela operação. 4. Sistema gera resultados. 5. Sistema ativa outro caso de uso.

Caso de uso 39: imprimirCartaoFase
Tipo: opcional
Objetivo: candidato imprime cartão para fase seguinte
Fluxo obrigatório
1. Coordenador entra com os dados da inscrição. 2. Sistema confirma desempenho do candidato 3. Sistema atualiza BD. 4. Sistema gera cartão.
5. Candidato imprime o cartão. 6. Sistema ativa outro caso de uso.
Fluxo alternativo
1. Coordenador entra com os dados da inscrição. 2. Sistema não confirma desempenho do candidato 3. Sistema envia mensagem ao candidato. 4. Sistema ativa outro caso de uso.

Caso de uso 41: pagarTaxaRecurso
Tipo: opcional
Objetivo: candidato seleciona forma de pagamento.
Fluxo obrigatório
1. Candidato opta pelo tipo de pagamento. 2. Candidato confirma envio dos dados. 3. Sistema atualiza BD. 4. Sistema ativa outro caso de uso.
Fluxo alternativo
1. Candidato opta pela forma de pagamento. 2. Candidato cancela operação 3. Sistema atualiza BD. 4. Sistema ativa outro caso de uso.

Caso de uso 42: entrarRePrTitulos
Tipo: opcional
Objetivo: candidato entra com recurso para prova de títulos.
Fluxo obrigatório
1. Candidatos preenche formulário de recurso. 2. Candidato seleciona forma de pagamento. 3. Sistema ativa outro caso de uso efetuarPgto. 4. Sistema atualiza BD.
Fluxo alternativo
1. Candidatos preenche formulário de recurso. 2. Candidato cancela operação. 3. Sistema ativa outro caso de uso.

Caso de uso 43: entrarRePrObjetiva
Tipo: opcional
Objetivo: candidato entra com recurso para prova objetiva.
Fluxo obrigatório
1. Candidatos preenche formulário de recurso. 2. Candidato seleciona forma de pagamento. 3. Sistema ativa outro caso de uso efetuarPgto. 4. Sistema atualiza BD.
luxo alternativo
1. Candidatos preenche formulário de recurso. 2. Candidato cancela operação. 3. Sistema ativa outro caso de uso.

Caso de uso 44: entrarRePrJurídica
Tipo: opcional
Objetivo: candidato entra com recurso para prova de títulos.
Fluxo obrigatório
1. Candidatos preenche formulário de recurso.

2. Candidato seleciona forma de pagamento. 3. Sistema ativa outro caso de uso efetuarPgto. 4. Sistema atualiza BD.
Fluxo alternativo
1. Candidatos preenche formulário de recurso. 2. Candidato cancela operação. 3. Sistema ativa outro caso de uso.

Caso de uso 45: entrarRePrDiscursiva
Tipo: opcional
Objetivo: candidato entra com recurso para prova de títulos.
Fluxo obrigatório
1. Candidatos preenche formulário de recurso. 2. Candidato seleciona forma de pagamento. 3. Sistema ativa outro caso de uso efetuarPgto. 4. Sistema atualiza BD.
Fluxo alternativo
1. Candidatos preenche formulário de recurso. 2. Candidato cancela operação. 3. Sistema ativa outro caso de uso.

Caso de uso 46: entrarReHomoInscrição
Tipo: opcional
Objetivo: candidato entra com recurso para prova de títulos.
Fluxo obrigatório
1. Candidatos preenche formulário de recurso. 2. Candidato seleciona forma de pagamento. 3. Sistema ativa outro caso de uso efetuarPgto. 4. Sistema atualiza BD.
Fluxo alternativo
1. Candidatos preenche formulário de recurso. 2. Candidato cancela operação. 3. Sistema ativa outro caso de uso.

Caso de uso 47: gerResProvas
Tipo: obrigatório
Objetivo: coordernador faz opção pelo tipo da prova relacionado ao resultado que deverá ser gerado.
Fluxo obrigatório
1. Coordenador seleciona prova relacionado ao resultado. 2. Coordenador insere os dados referentes a opção escolhida 3. Sistema ativa outro caso de uso. 4. Sistema atualiza BD.

Fluxo alternativo
1. Coordenador seleciona prova relacionado ao resultado. 2. Coordenador cancela operação 3. Sistema atualiza BD. 4. Candidato cancela operação. 5. Sistema ativa outro caso de uso.

Caso de uso 48: pubOnline
Tipo: obrigatório
Objetivo: coordenador disponibiliza material online
Fluxo obrigatório
1. Coordenador disponibiliza material onLine 2. Coordenador insere os dados referentes a publicação. 3. Sistema ativa outro caso de uso. 4. Sistema atualiza BD.
Fluxo alternativo
1. Coordenador disponibiliza material onLine 2. Coordenador cancela operação 3. Sistema atualiza BD. 4. Candidato cancela operação. 5. Sistema ativa outro caso de uso.

Caso de uso 49: pubDownload
Tipo: obrigatório
Objetivo: coordenador disponibiliza material para download.
Fluxo obrigatório
1. Coordenador disponibiliza material para download 2. Coordenador insere os dados referentes a publicação. 3. Sistema ativa outro caso de uso. 4. Sistema atualiza BD.
Fluxo alternativo
1. Coordenador disponibiliza material para download 2. Coordenador cancela operação 3. Sistema atualiza BD. 4. Candidato cancela operação. 5. Sistema ativa outro caso de uso.

Apêndice B

Documento de Requisitos da Linha de Produto para Gestão de Processos Seletivos (LP-GPS)

Este documento de requisitos é a especificação dos requisitos da Linha de Produto para Gestão de Processos Seletivos obtida através da modelagem utilizando a abordagem ProLineA.

Foram analisados 3 tipos de processos seletivos: Concurso Vestibular, Concurso Público para Contratação Docente, Concurso Publico para Contratação de Profissionais

B1. Descrição do domínio

A linha de produto para gestão de processos seletivos visa o desenvolvimento de produtos de softwares para controle de processos de inscrição, provas, homologação e resultados de diferentes tipos de processos seletivos.

O administrador será responsável pela administração do sistema e gerenciamento de usuários da comissão de concursos ou o administrador.

O administrador será o responsável por controlar o sistema, cadastrando os dados do vestibular, disponibilizando informações e realimentando o sistema para os processos de homologação e ensalamento.

B2. Modelo de Processo de Negócio

Modelo de Processo de Negócios (Atividade Inscrição

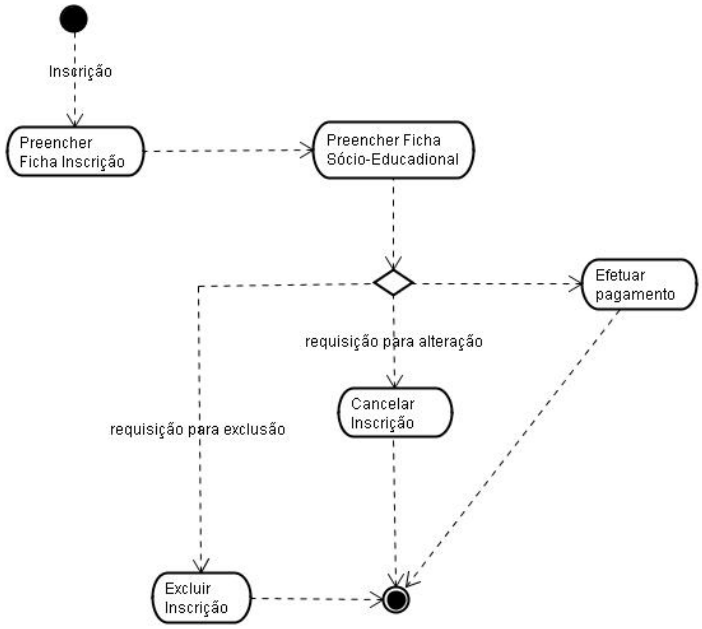


Figura B.1 – Diagrama de atividades para Inscrição

Modelo de Processo de Negócios (Atividade Homologação)

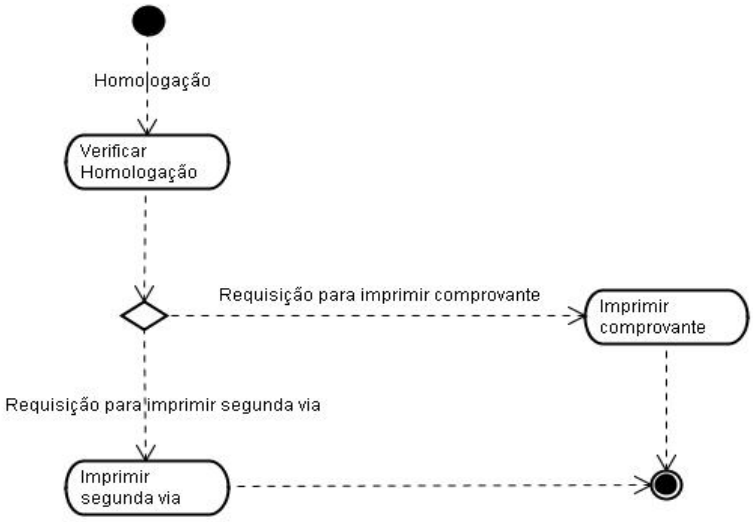


Figura B.2 – Diagrama de atividades para Homologação

B3. Atores de Papéis

ATOR	PAPEL
Administrador	- administrar o sistema - gerenciar os usuários do processo seletivo
Coordenadores do Processo	- controlar o sistema geral - cadastrar os dados do processo seletivo - disponibilizar resultados do processo - verificar pagamento do candidato - gerenciar homologação - gerenciar provas - gerenciar locais de prova - gerenciar ensalamento
Candidato	- preencher a ficha de inscrição on-line - preencher ficha sócio-educacional - efetuar pagamento referente à ficha de inscrição - na data prevista verificar sua homologação e imprimir o comprovante de inscrição
Sistema Bancário	- gerenciar pagamento on-line - gerenciar pagamento boleto - gerenciar pagamento depósito - fornecer arquivo com candidatos que efetuaram o pagamento para o coordenador do processo
Sistema de Correção	- disponibilizar a correção da prova para o sistema

Tabela B.1 – Atores x papéis

B4. Requisitos Funcionais do Domínio

1 O sistema deve permitir a alteração, cancelamento, e a inscrição de candidatos com os seguintes atributos: nome, RG, CPF, telefone, e-mail, endereço, curso.

2. O sistema deve permitir a inclusão dos dados da ficha sócio educacional on line, com todos os atributos conforme constam no apêndice A1.

3. O sistema deve permitir a inclusão, alteração e cancelamento de processos seletivos, com os seguintes atributos: código do processo, datas de inscrição, datas de provas, locais de prova, ensalamento.

4. O sistema deve permitir o preenchimento de anexação de documentação da prova de títulos/currículos.

5. O sistema deve permitir que o gerenciamento da forma de pagamento.

6. O sistema deve permitir inclusão, alteração e cancelamento de solicitação de taxa isenção de pagamento.

7. O sistema deve gerar e gerenciar a homologação dos candidatos, com os seguintes atributos: nome do candidato, RG, CPF, número de inscrição, data da prova, curso, local de prova (bloco) e local de prova (sala).

8. O sistema deve permitir publicação de editais, resultados, manuais, resultados individuais e relatórios.

9. O sistema deve permitir disponibilização de material para download. Em formato pdf, contendo os seguintes atributos: nome do arquivo, tamanho do arquivo, data de publicação.

10. O sistema deve permitir a inclusão, alteração e cancelamentos de solicitação de recursos. Com os seguintes atributos: código do recurso, tipo de prova, data do recurso, prazo para resposta.

11. O sistema deve gerar os resultados das provas.

12. O sistema deve permitir a inserção, cancelamento e alteração de chamadas e convocações. Com os seguintes atributos: nome do candidato, tipo convocação

13. O sistema deve enviar ao candidato dados referentes à efetivação de sua inscrição. Com os seguintes atributos: data inscrição, flag do pagamento (local e data da efetivação do pagamento), número de incrição, nome candidato, cpf, rg, curso/vaga/função.

Requisito	**Caso de uso**	**DCU/F**
Req.1, Req.4, Req.5, Req.6.	alterarInscrição	
Req. 1, Req.4, Req.5, Req.6.	cancelarInscrição	
Req. 1, Req.4, Req.5, Req.6	fazerInscrição	*
Req.5, Req.4, Req.6	fazerInscEspecial	*
Req.5	fazerInscAfro	*
Req.4	fazerInscEsp	*
Req.6	fazerInscInd	*
Req. 1, Req.4, Req.5, Req.6	utilizarEnem	*
Req. 1, Req.4, Req.5, Req.6	preencherFormTitulos	*
Req. 1, Req.4, Req.5, Req.6	preencherFormSocioEdu	*
Req.2	pagarTaxaInsc	*
Req.2	pagarOnline	*
Req.2	pagarBoleto	*
Req.2	pagarCartão	*
Req.3, Req.1, Req.4, Req.5, Req.6	solicitarIsenção	
Req.8	controlarHomo	
Req.8	imprimirCartãoHomo	*
Req.8	segundaViaCartão	
Req.9	publicarEditais	*

Req.9	publicarResultados	*
Req.9	publicarResTotal	*
Req.9	publicarResTotalDow	
Req.9	publicarResInd	*
Req.9	publicarManual	*
Req.9	publicarManualOn	
Req.9	publicarManualDow	*
Req.9	publicarGabaritos	*
Req.9	publicarProvas	*
Req.10, Req. 11	gerarRePrObjetivas	*
Req.10, Req. 11	gerarRePrDiscursivas	*
Req.10, Req. 11	gerarRePrPráticas	*
Req.10, Req. 11	gerarRePrDidática	*
Req.10, Req. 11	gerarReEntrevista	*
Req.10, Req. 11	gerarRePrTitulos	*
Req.12	convocarCandExame	
Req.12	convocarCandNomear	
Req.12	convocarCandMatrícula	
Req.12	convocarCandFase	
Req.12	imprimirCartãoFase	
Req.13	entrarRecurso	*
Req.13	pagarTaxaRecurso	*
Req.13	entrarRePrTitulo	*
Req.13	entrarRePrObjetiva	*
Req.13	entrarRePrJurídica	*
Req.13	entrarRePrDiscursiva	*
Req.13	entrarReHomoInscrição	*
Req.10, Req.11	gerarResProvas	*
Req.9	publicarOnline	*
Req.9	publicarDownload	*

Tabela B2 – Requisitos x casos de uso x representação em diagrama

B5. Requisitos não funcionais do domínio

Segurança

16. O sistema deve gerenciar a inclusão, alteração e remoção de usuários.

17. O sistema deve permitir dos autenticação dos usuários cadastrados.

18. O sistema deve efetuar a criptografia das senhas de acesso dos usuários para maior confiabilidade e segurança no acesso as informações.

19. O sistema deve efetuar log de acessos dos usuários, para registro de acesso do usuário

Controle de acesso

20. O sistema deve controlar o acesso a determinadas funções e serviços do sistema conforme o usuário.

Persistência

21. O sistema deve permitir que as informações necessárias sejam armazenadas (persistidas) adequadamente.

Desempenho

22. O sistema deve permitir a inscrição do usuário em 10 minutos (máximo).

23. O sistema deve permitir o preenchimento da ficha sócio educacional em 10 minutos máximo)

24. O sistema deve permitir a impressão do cartão de homologação, fases, comprovante de inscrição, resultados, entre outros em 3 minutos (máximo).

Requisito	Caso de uso	DCU/NF
Req.16	gerUsuário	
Req.17	auntentUsuário	*
Req.18	criptSenhas	
Req.19	registrarOperUser	*
Req.20	permitirAcesso	*
Req.21	persistirDados	*
Req.22	controlarTempInscrição	*
Req.23	controlarTempFichaSócio	*
Req.24	controlarTempImpressão	*

Tabela B.3 – Requisito não funcionais x caso de uso x representação em diagrama de casos de uso