Instituto de Ciências Matemáticas e de Computação

ISSN - 0103-2569

Aspectos de Variabilidade em Mecanismos de Comunicação de um Ambiente de Aprendizagem Eletrônica

Edson Alves de Oliveira Junior
Renata Pontin de Mattos Fortes
Rudinei Goularte

Nº 298

RELATÓRIOS TÉCNICOS DO ICMC

São Carlos
Maio/2007

SUMÁRIO

1. INTRODUÇÃO 5

1.1 OBJETIVOS DO TRABALHO 6

2. LINHA DE PRODUTO DE SOFTWARE 8

2.1 DESENVOLVIMENTO DO NÚCLEO DE ARTEFATOS 10

2.2 DESENVOLVIMENTO DO PRODUTO 11

2.3 GERENCIAMENTO DA LINHA DE PRODUTO 12

3. GERENCIAMENTO DE VARIABILIDADE EM LINHA DE PRODUTO 14

3.1 CARACTERIZAÇÃO DE VARIABILIDADE 14

3.2 PROCESSO DE GERENCIAMENTO DE VARIABILIDADE 17

3.2.1 Elaboração do Modelo de Rastreamento de Variabilidades 20

3.2.2 Identificação das Variabilidades 21

3.2.3 Delimitação das Variabilidades 24

3.2.4 Escolha dos Mecanismos de Implementação de Variabilidade 26

3.2.5 Rastreamento e Controle das Variabilidades 26

3.2.6 Análise de Configurações de Produtos Específicos 27

4. VARIABILIDADE NO AMBIENTE TIDIA-AE 28

4.1 VARIABILIDADE NO MODELO CONCEITUAL DO LMS 29

4.2 VARIABILIDADE NA FERRAMENTA DE *CHAT* 31

4.3 VARIABILIDADE NA FERRAMENTA *WHITEBOARD* 39

4.4 VARIABILIDADE NA FERRAMENTA *INSTANT MESSENGER* 47

4.5 ANÁLISE DOS RESULTADOS E DIFICULDADES ENCONTRADAS 57

5. CONSIDERAÇÕES FINAIS 60

REFERÊNCIAS 61

ANEXO I - RELAÇÕES E ESTEREÓTIPOS USADOS NA REPRESENTAÇÃO GRÁFICA DE PONTOS DE VARIAÇÃO E VARIANTES 65

RESUMO

Este relatório apresenta a aplicação das atividades de gerenciamento de variabilidade em linha de produto ao ambiente de aprendizagem eletrônica TIDIA-Ae com o objetivo de identificar, delimitar e representar graficamente as variabilidades em mecanismos de comunicação de tal ambiente. Uma análise acerca das variabilidades identificadas e do domínio em questão é apresentada ao final deste relatório.

ABSTRACT

This technical report presents the product line variability management activities realization in an electronic learning environment, the TIDIA-Ae, regarding its identification, delimitation and graphical representation of variabilities in communication mechanisms of such environment. At the end, an analysis of the identified variabilities and the concerning domain is presented.

1. Introdução

A abordagem de linha de produto de software (LP) tem como objetivo principal promover a geração de produtos específicos de um determinado domínio com base na reutilização de uma infra-estrutura central. Essa infra-estrutura, também conhecida como núcleo de artefatos, contém aspectos comuns e variáveis. A composição e a configuração desses artefatos dão origem a um produto específico da LP.

A abordagem de LP é adequada aos domínios em que existe uma demanda por produtos que possuem características comuns, mas que contêm pontos de variação bem definidos. Atualmente, existem várias abordagens de LP (CZARNECKI; EISENECKER, 2000; CLEMENTS; NORTHROP, 2001; GIMENES; TRAVASSOS, 2002; MANTILASSI, 2004). Dentre os benefícios conseguidos com a abordagem de LP (HEYMAND; TRIGAUX, 2003; CLAMENTS; NORTHROP, 2001), pode-se citar:

- melhor compreensão de domínios de aplicação;
- maior reutilização de artefatos; e
- diminuição do tempo para produtos chegarem ao mercado.

Esses benefícios podem ser evidenciados por meio de relatos de empresas que utilizaram a abordagem de LP (BASS et al., 2003; BUHRDORF, 2003; OMMENING, 2004). No Brasil, empresas de pequeno e médio porte têm se beneficiado desta abordagem, construindo, por exemplo, uma LP com 70 membros em 4 meses no domínio de jogos para celular (MEANTIME, 2006; ALVES et. al, 2005).

Apesar dos benefícios apontados na adoção da abordagem de LP, ainda existem dificuldades, tais como a evolução do núcleo de artefatos, a avaliação de uma linha de produto, a análise de aspectos econômicos, o suporte automatizado, além do gerenciamento de variabilidades.

A avaliação de linha de produto é uma questão importante tanto do ponto de vista industrial quanto acadêmico (VAN DER LINDEN et al., 2004). Do ponto de vista acadêmico existem temas em aberto de pesquisa como teste de LP (MUCCINI;

VAN DER HEOK, 2003) e análise econômica de LP, bem como questões relativas à disponibilidade de dados para avaliação de LP.

O gerenciamento de variabilidades é um tema que vem sendo ultimamente muito discutido pela comunidade de LP. Vários trabalhos vêm sendo realizados nesse sentido, sendo os mais importantes à identificação, definição e representação gráfica de variabilidade (OLIVEIRA JUNIOR et al., 2005).

Nesse sentido, este relatório apresenta a aplicação do processo de gerenciamento de variabilidade em LP proposto por Oliveira Junior (2005) ao ambiente de aprendizagem eletrônica TIDIA-Ae com o objetivo de identificar, delimitar e representar suas variabilidades. Uma análise é realizada por meio das variabilidades identificadas para mostrar se o domínio de aprendizagem eletrônica é propício à abordagem de LP e, com isso, realizar experimentos formais para a avaliação de LP.

Este relatório está estruturado da seguinte maneira: a seção 2 apresenta os conceitos fundamentais sobre linha de produto de software; a seção 3 apresenta uma caracterização de variabilidade em LP, bem como o processo de gerenciamento de variabilidade; a seção 4 apresenta a aplicação das atividades do processo de gerenciamento de variabilidade em mecanismos de comunicação do ambiente TIDIA-Ae e uma análise dos resultados obtidos; e, finalmente, a seção 5 apresenta as considerações finais acerca do trabalho realizado.

1.1 Objetivos do Trabalho

O objetivo deste trabalho é apresentar a aplicação do processo de gerenciamento de variabilidade proposto por Oliveira Junior (2005) com o intuito de identificar, delimitar e representar graficamente as variabildiades nos mecanismos de comunicação do ambiente TIDIA-Ae. Ao final, uma análise acerca do domínio de aprendizagem eletrônica é apresentada com relação à sua aplicabilidade no desenvolvimento de linha de produto e a realização de experimentos formais para apoiar a avaliação de LP. Dentre os objetivos específicos, podemos citar:

- estudo do ambiente TIDIA-Ae e as ferramentas que o compõem;
- aplicação do processo de gerenciamento de variabilidade nas ferramentas: *chat*, *whiteboard* e *instant messenger*;

- análise acerca das variabilidades encontradas;
- análise acerca do domínio em questão e a sua aplicabilidade no desenvolvimento de linhas de produto e a realização de experimentos formais para avaliação de LP.

2. Linha de Produto de Software

Uma LP representa um conjunto de sistemas que compartilham características comuns e gerenciáveis que satisfazem as necessidades de um segmento particular do mercado ou de uma missão (CLEMENTS; NORTHROP, 2001; NORTHROP, 2002). Os produtos específicos de uma LP são desenvolvidos de maneira sistemática a partir de um conjunto comum de artefatos da LP.

Nos últimos anos, tem-se percebido uma crescente adoção da abordagem de LP por causa dos seus benefícios. Essa abordagem possibilita às organizações explorar as semelhanças entre seus produtos, aumentando, assim, a reutilização de artefatos e, como conseqüência, tem-se uma diminuição dos custos e do tempo no desenvolvimento (HEYMANS; TRIGAUX, 2003).

Segundo Clements e Northrop (2001) e Heymans e Trigaux (2003), os benefícios conseguidos com a adoção da abordagem de LP podem ser classificados em:

- **organizacionais:** melhor compreensão do domínio e aumento da qualidade dos produtos e da confiança do cliente;
- **engenharia de software:** melhor análise e aumento da reutilização dos requisitos e dos artefatos, melhor controle da qualidade dos produtos, estabelecimento de padrões e documentação reutilizável;
- **negócio:** redução dos gastos com teste e manutenção.

A adoção de uma abordagem de LP leva em consideração fatores organizacionais (CLEMENTS; NORTHROP, 2001; NORTHROP, 2002; SEI, 2006), a saber: a natureza dos produtos, o mercado ou missão, as metas de negócio, a estrutura organizacional, a cultura e as políticas organizacionais, as disciplinas de processo de software, a maturidade dos artefatos legados e a distribuição geográfica da equipe de trabalho.

A obtenção dos benefícios acima indicados implica em contrapartida das organizações. Um exemplo disso é a mudança da visão de desenvolvimento de produtos específicos para uma abordagem de LP que exige treinamento e adaptação de comportamento da equipe de desenvolvimento e gerenciamento.

O desenvolvimento seguindo uma abordagem de LP requer gerenciamento em longo prazo, pois os benefícios não são imediatamente visíveis. Assim, uma grande quantidade de produtos deve ser produzida para que a adoção desta abordagem possa ser realmente consolidada.

O SEI (*Software Engineering Institute*), através da iniciativa PLP (*Product Line Practice*), estabeleceu as atividades essenciais de uma abordagem de LP (SEI, 2006). Essas atividades são: o **Desenvolvimento do Núcleo de Artefatos**, também conhecida como Engenharia de Domínio, o **Desenvolvimento do Produto,** também conhecida como Engenharia de Aplicação**,** e o **Gerenciamento da Linha de Produto**. A Figura 1 ilustra as interações entre essas atividades.

Os três círculos da Figura 1 indicam que as atividades de uma LP são altamente interligadas e iterativas. As flechas rotativas indicam que, além dos artefatos gerados no desenvolvimento de produtos do núcleo, são também realizadas revisões dos artefatos ou até mesmo inclusão de novos artefatos.

Na medida em que os produtos são desenvolvidos, o núcleo de artefatos pode evoluir. Um exemplo disto ocorre quando se identifica um novo requisito relevante ao domínio que inicialmente não fazia parte da especificação de requisitos da LP. Assim, essa especificação evolui, incorporando o novo requisito identificado e o núcleo de artefatos é atualizado.

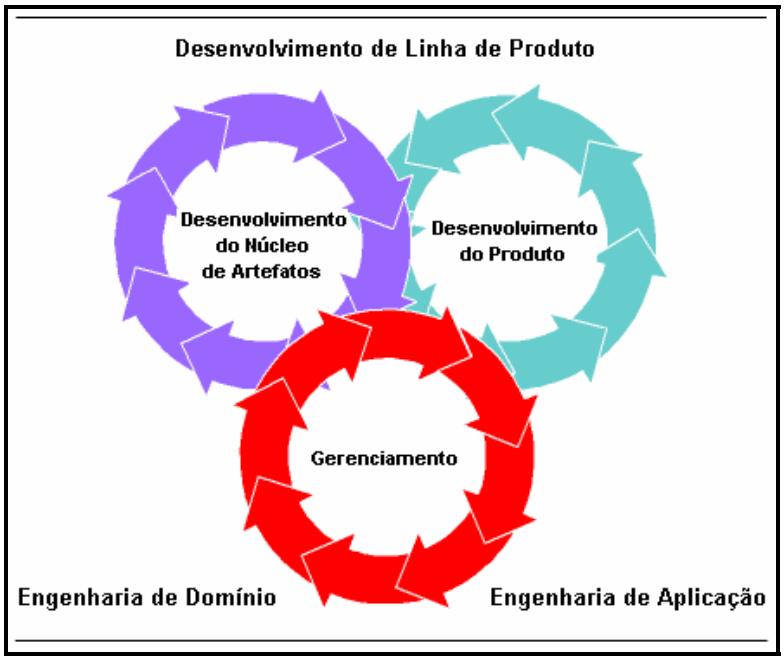

Figura 1: Atividades Essenciais de Linha de Produto (OLIVEIRA JUNIOR, 2005).

A seguir, é apresentada uma breve descrição das atividades essenciais de LP (OLIVEIRA JUNIOR, 2005).

2.1 Desenvolvimento do Núcleo de Artefatos

O objetivo da atividade de desenvolvimento do núcleo de artefatos é estabelecer uma infra-estrutura central que será reutilizada pelos produtos gerados a partir da LP.

Os artefatos de entrada para esta atividade são:

- as **restrições do produto:** as semelhanças e as variações entre produtos que constituirão a LP, as normas que devem ser seguidas para definir as características dos produtos e os limites de desempenho;
- os **estilos arquiteturais, padrões e *frameworks*:** utilizados para a construção da arquitetura de LP;
- as **restrições de produção:** os componentes COTS (*Commercial-Off-The-Shelf*) que serão utilizados, as normas que serão seguidas para a produção dos produtos e os componentes legados que serão reutilizados;
- a **estratégia de produção:** estratégia geral da abordagem para produzir os artefatos da LP;
- o **repositório dos artefatos pré-existentes:** os artefatos existentes como, por exemplo, componentes, especificações ou partes legadas do domínio catalogadas para a sua futura reutilização.

Esta atividade tem como principais artefatos de saída:

- o **contexto da linha de produto:** descrição dos produtos que constituirão a LP ou que esta é capaz de produzir. Esta descrição apresenta as semelhanças e as variações entre os produtos, além de incluir características e operações destes, tais como desempenho e atributos de qualidade;
- o **núcleo de artefatos da linha de produto:** base para a produção de produtos a partir da LP. Normalmente, o núcleo de artefatos inclui uma arquitetura a ser reutilizada pelos produtos, bem como os seus componentes;

- o **plano de produção dos produtos:** descrição das decisões a serem tomadas para instanciar produtos específicos a partir do núcleo de artefatos da LP.

A Figura 2 mostra os artefatos de entrada e saída da atividade de desenvolvimento do núcleo de artefatos.

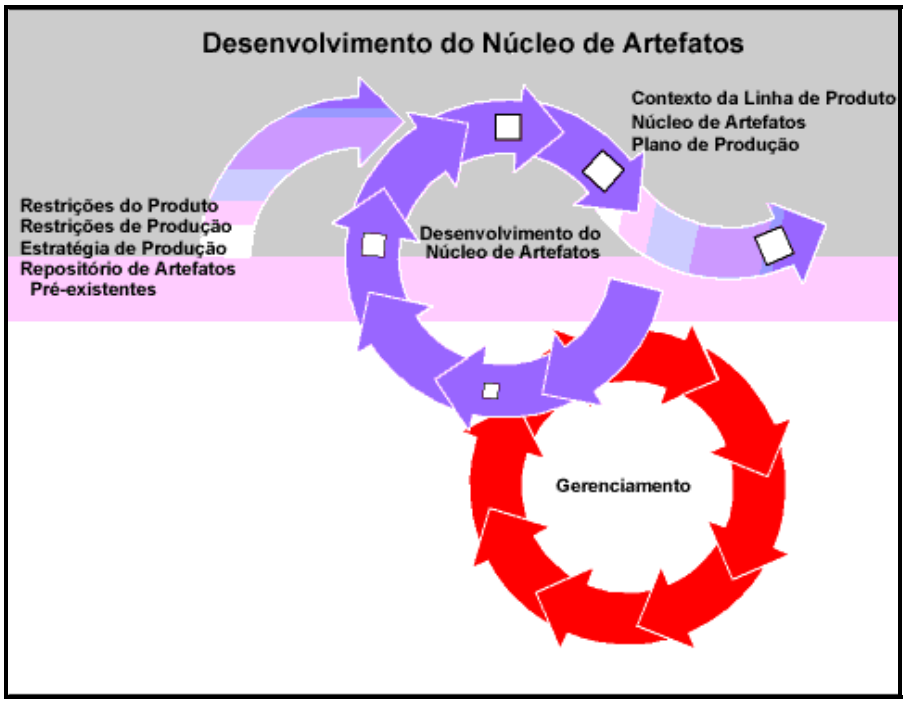

Figura 2: Desenvolvimento do Núcleo de Artefatos (OLIVEIRA JUNIOR, 2005).

2.2 Desenvolvimento do Produto

A atividade de desenvolvimento do produto tem como principal objetivo a geração de produtos de uma LP. Porém, é possível identificar requisitos que, inicialmente, não haviam sido especificados e, conseqüentemente, atualizar o núcleo de artefatos da LP.

Essa atividade depende substancialmente dos artefatos de saída da atividade anterior, que lhe servem como entrada. Esses artefatos são o **contexto da LP**, o **núcleo de artefatos** e o **plano de produção**. Além desses artefatos, são necessários também os requisitos para um produto específico.

As flechas rotativas indicam iteração e relacionamentos intrínsecos como, por exemplo, a existência e a disponibilidade de um produto específico podem afetar os requisitos de um produto subseqüente.

A Figura 3 mostra os artefatos de entrada e saída da atividade de desenvolvimento do produto.

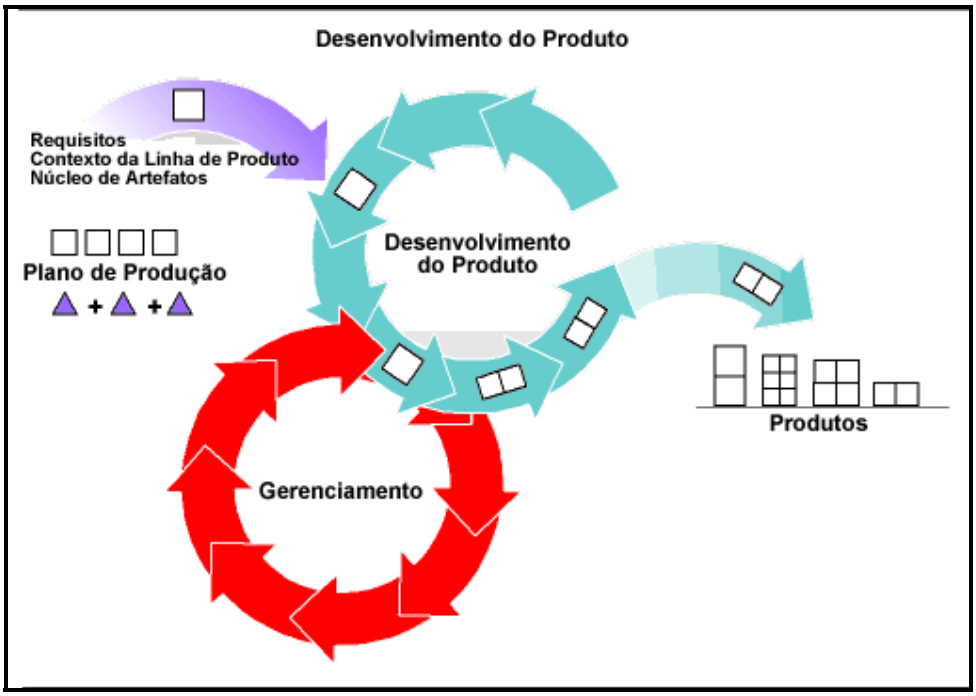

Figura 3: Desenvolvimento do Produto (OLIVEIRA JUNIOR, 2005).

2.3 Gerenciamento da Linha de Produto

A atividade de gerenciamento de LP deve garantir que todas as atividades técnicas sejam realizadas de acordo com um planejamento coordenado.

O gerenciamento pode ser dividido em duas categorias (SEI, 2006):

- **gerenciamento técnico**: coordena as atividades de desenvolvimento do núcleo de artefatos e desenvolvimento do produto para garantir que as equipes de desenvolvimento sigam os processos definidos para a LP e coletem dados suficientes para acompanhar o progresso desta;
- **gerenciamento organizacional**: garante que as unidades organizacionais recebem os recursos corretos (ex. treinamento) em quantidades suficientes.

Uma das ações mais importantes da atividade de gerenciamento da LP é a criação de um plano de adoção que descreva o estado desejado da organização e uma estratégia para alcançar tal estado.

O gerenciamento de LP também deve estabelecer como as atividades de desenvolvimento do núcleo de artefatos e desenvolvimento do produto devem interagir para permitir a evolução da LP e o gerenciamento das semelhanças e das variabilidades de cada artefato da LP.

3. Gerenciamento de Variabilidade em Linha de Produto

Dentre os principais aspectos que devem ser gerenciados em uma LP, estão as diferenças entre seus produtos, conhecidas como variabilidades. Variabilidade é um dos assuntos relacionados à LP que mais vem chamando a atenção tanto da comunidade acadêmica quanto da empresarial devido à complexidade do seu gerenciamento (HEYMANS; TRIGAUX, 2003). Esse fato pode ser evidenciado, por exemplo, com a existência de um *workshop* internacional específico sobre o assunto (*Workshop on Software Variability Management*) em uma das conferências internacionais mais importantes sobre LP, a *Software Product Line Conference* (SPLC, 2006).

3.1 Caracterização de Variabilidade

A variabilidade é descrita por pontos de variação e variantes. Um ponto de variação é um local específico de um artefato em que uma decisão de projeto ainda não foi resolvida. A cada ponto de variação está associado um conjunto de variantes. Cada variante corresponde a uma alternativa de projeto para instanciar uma determinada variabilidade. A resolução de um ponto de variação se dá através da escolha de uma ou mais variantes do conjunto de variantes relacionado (HEYMANS; TRIGAUX, 2003).

A representação explícita de variabilidade torna possível a geração de produtos específicos de uma LP. A variabilidade surge do adiamento de certas decisões fundamentais ao projeto de produtos de software. Essas decisões, quando tomadas no início do projeto, restringem o domínio no qual o sistema pode ser aplicado.

As decisões tomadas em fases iniciais do projeto de um sistema computacional correspondem à abordagem tradicional de desenvolvimento de um único produto e não à abordagem de LP. Assim, quanto maior o número de decisões de projeto adiadas, maior será o número de variabilidades de um produto de software (HALMANS; POHL, 2003).

Segundo Van Gurp e Bosch (2000), as variabilidades podem estar associadas a diferentes níveis de abstração, sendo eles a descrição da arquitetura, a documentação de projeto, o código fonte, o código compilado, o código ligado e o código

executável. Esses diferentes níveis de abstração podem estar associados a diferentes estágios do desenvolvimento de software.

As variabilidades em uma LP podem ser classificadas em relação a diferentes aspectos, sendo eles:

- **estado das decisões de projeto** (VAN GURP; BOSCH, 2001);
- **tempo e espaço** (HALMANS; POHL, 2003);
- **ponto de vista do cliente e do gerente da LP** (HALMANS; POHL, 2003);
- **adição e remoção de funcionalidades** (ANASTASOPOULOS, 2001);
- **local de ocorrência** (BACHMANN; BASS, 2001);
- **níveis** (SVAHNBERG; BOSCH, 2000).

Um fator importante no gerenciamento de variabilidade é o seu tempo de resolução. Ele indica em qual momento uma ou mais variantes serão associadas a um determinado ponto de variação (VAN GURP; BOSCH, 2000).

Como exemplo, pode-se considerar uma classe responsável por encapsular a lógica de ordenação de um vetor de elementos (CLAUß, 2001a). Os pontos de variação dessa classe podem ser o tipo de algoritmo usado, sendo as possíveis variantes *bubble_sort, quick_sort* ou *insert_sort* e o tipo de elemento que será ordenado, sendo as possíveis variantes *IntegerSort* ou *StringSort*, como mostra a Figura 4. Assim, o tempo de resolução do ponto de variação poderia ser, por exemplo, tempo de execução, o que permitiria ao usuário escolher, com a aplicação em execução, o algoritmo de ordenação a ser usado e o tipo de elemento a ser ordenado.

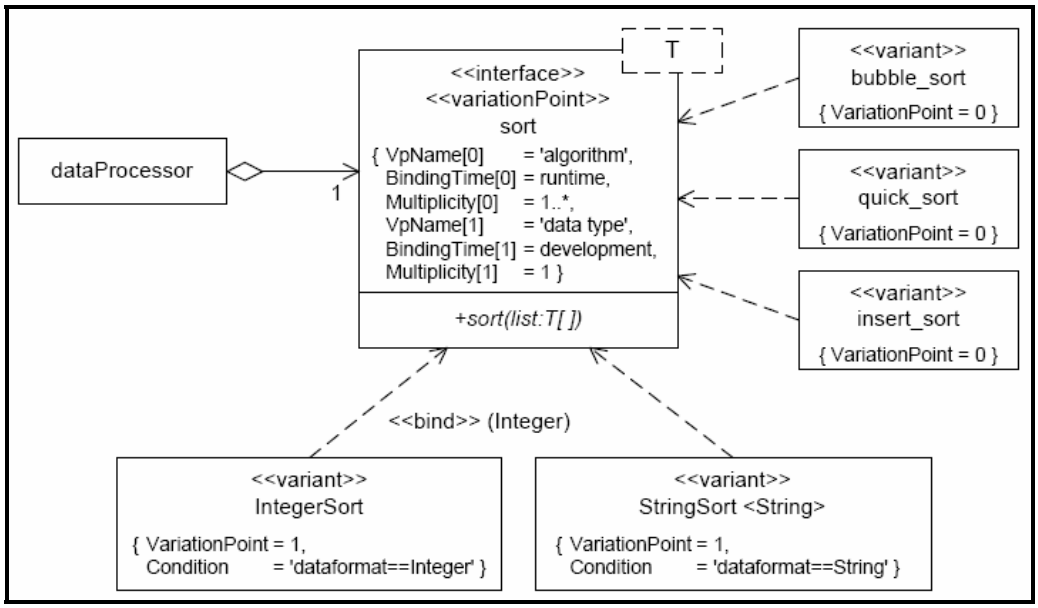

Figura 4: Exemplo de Ponto de Variação e Variantes (OLIVEIRA JUNIOR, 2005).

Segundo Anastasopoulos (2001), o tempo de resolução de variabilidade pode ser classificado como:

- **tempo de compilação:** a variabilidade é resolvida antes do programa ser compilado ou durante a compilação, usando diretivas de pré-processamento, por exemplo;
- **tempo de ligação:** a variabilidade é resolvida durante a ligação do módulo ou da biblioteca, selecionando diferentes bibliotecas com diferentes versões das operações exportadas;
- **tempo de execução:** a variabilidade é resolvida durante a execução do programa;
- **tempo de atualização:** a variabilidade é resolvida durante a atualização de programas ou após o início da execução destes (por exemplo, um programa de atualização que adiciona funcionalidades como o *Windows Update* da Microsoft).

Fritsch, Lehn e Strohm (2002), assim como Anastasopoulos (2001), classificam os tempos de resolução de variabilidade como compilação e tempo de execução, além de:

- **programação:** pode ser especializado em:
    - **desenvolvimento do núcleo de artefatos**: durante a construção dos artefatos da LP;
    - **desenvolvimento do produto**: durante a instanciação da arquitetura de LP e de seus componentes.
- **integração:** pode ser especializado em:
    - **compilação;**
    - **seleção do código fonte.**
- **montagem**.

Segundo Fritsch, Lehn e Strohm (2002), o tempo de resolução de variabilidade restringe a escolha de mecanismos de implementação de variabilidade. Por exemplo, se uma variabilidade é resolvida em tempo de execução, não se pode implementá-la com um mecanismo que é definido em tempo de compilação.

3.2 Processo de Gerenciamento de Variabilidade

A adoção de um processo de gerenciamento de variabilidade permite um melhor gerenciamento de LP, pois contribui para aumentar tanto a qualidade dos artefatos produzidos quanto o número de possíveis produtos a serem construídos a partir de um domínio. Para tanto, as atividades deste processo devem estar bem documentadas, assim como os seus artefatos.

Um processo de gerenciamento de variabilidade é proposto por Oliveira Junior (2005). A concepção deste processo tomou como base as atividades de gerenciamento de variabilidade, propostas por van Gurp e Bosch (2001), a construção de modelos de *features*, proposta por Griss, Favaro e D'Alessandro (1998) e a construção de um modelo genérico de variabilidades, proposta por Becker (2003).

O processo de gerenciamento de variabilidade é composto por atividades e artefatos como mostra a Figura 5. O processo é realizado ao final de cada atividade do desenvolvimento de LP, tornando iterativo. Ele também é um processo incremental, uma vez que a cada nova iteração o número de variabilidades aumenta.

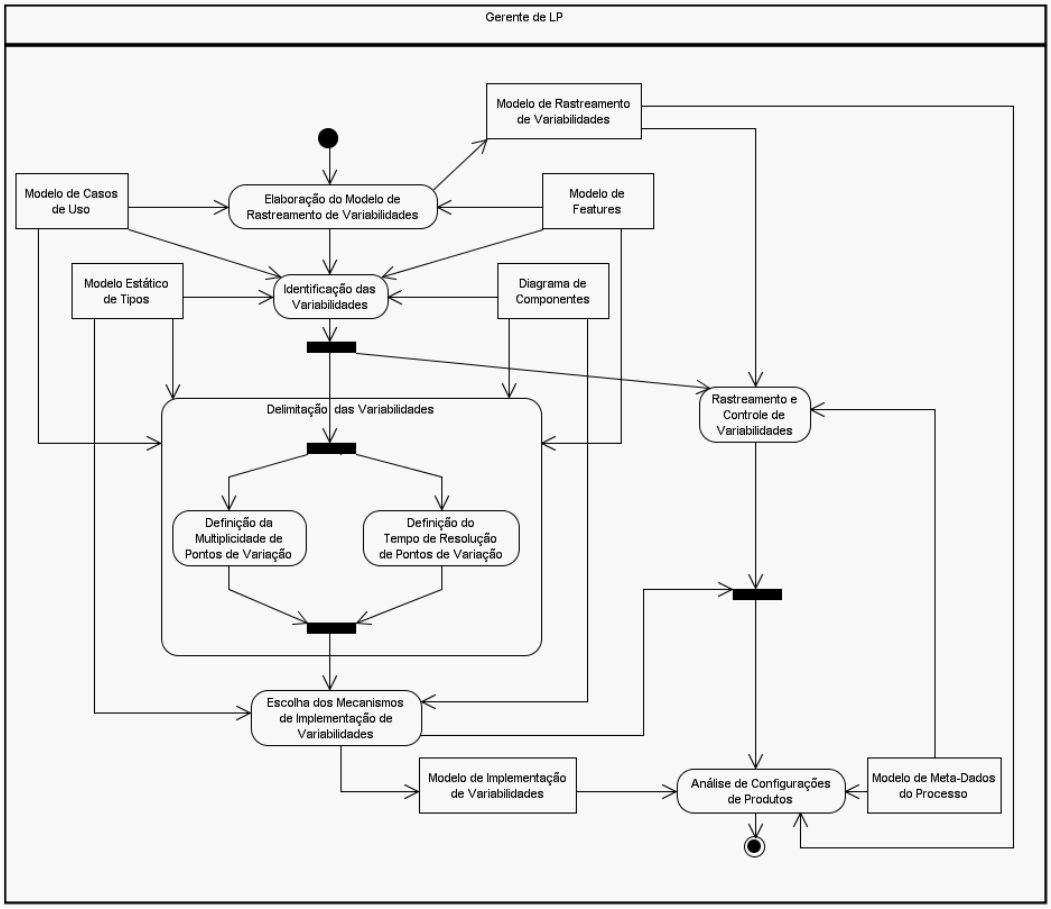

Figura 5: Atividades do Processo de Gerenciamento de Variabilidade para LP (OLIVEIRA JUNIOR, 2005).

No diagrama de atividades (Figura 5) que representa o processo de gerenciamento de variabilidade, as atividades são representadas pelas elipses e os artefatos de entrada e saída são representados pelos retângulos. Todas as atividades do processo são realizadas pelo Gerente de LP.

O processo é visto como um elemento que consome informações provenientes da atividade de desenvolvimento da LP e também produz informações para tal atividade como, por exemplo, o modelo de casos de uso e o modelo estático de tipos com as variabilidades identificadas e delimitadas. O processo também produz informações para a atividade de geração de produtos específicos como, por exemplo, o modelo de rastreamento de variabilidades e o modelo de implementação de variabilidades. A Figura 6 mostra o relacionamento entre o gerenciamento de variabilidade e as atividades de desenvolvimento de LP e geração de produtos. Nesta figura, o processo de gerenciamento de variabilidade está representado pelo retângulo

hachurado em cinza e corresponde a um dos elementos da atividade de gerenciamento de LP propostas pelo SEI (Figura 1).

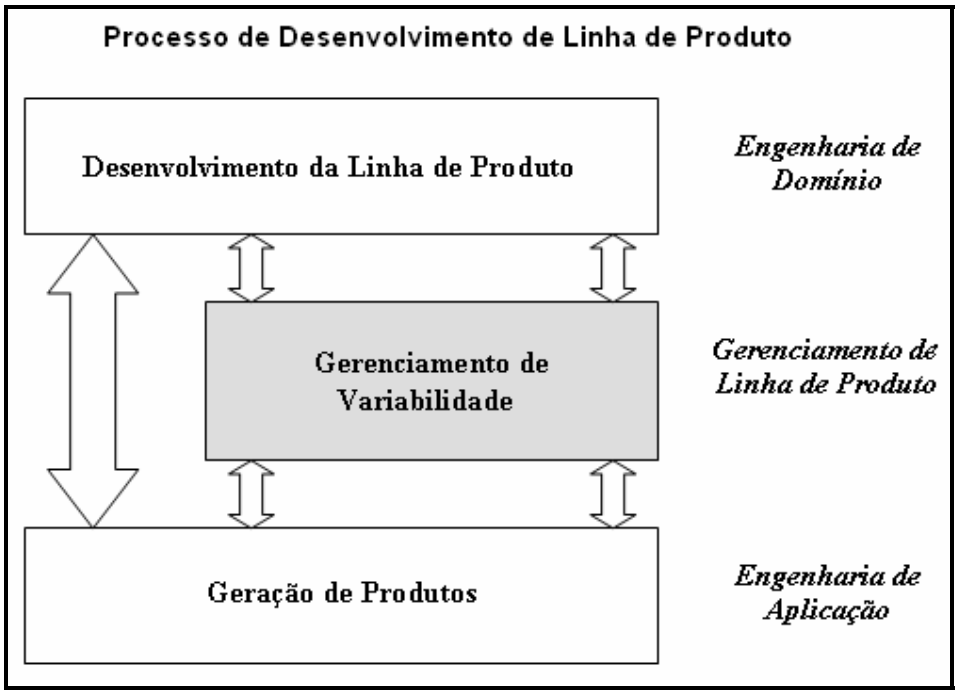

Figura 6: Processo de Desenvolvimento de LP com Gerenciamento de Variabilidade (OLIVEIRA JUNIOR, 2005).

A Figura 7 mostra a interação entre as atividades de desenvolvimento da LP, realizadas pelo engenheiro de LP, e as do gerenciamento de variabilidade, realizadas pelo gerente de LP. A figura mostra que ao final de cada atividade do desenvolvimento da LP é realizada uma iteração do gerenciamento de variabilidade. Comparando esta figura com a abordagem PLP ilustrada na Figura 1, as atividades alinhadas verticalmente à esquerda correspondem ao ciclo superior esquerdo das atividades propostas pelo SEI (Figura 1), enquanto as demais atividades correspondem ao ciclo inferior.

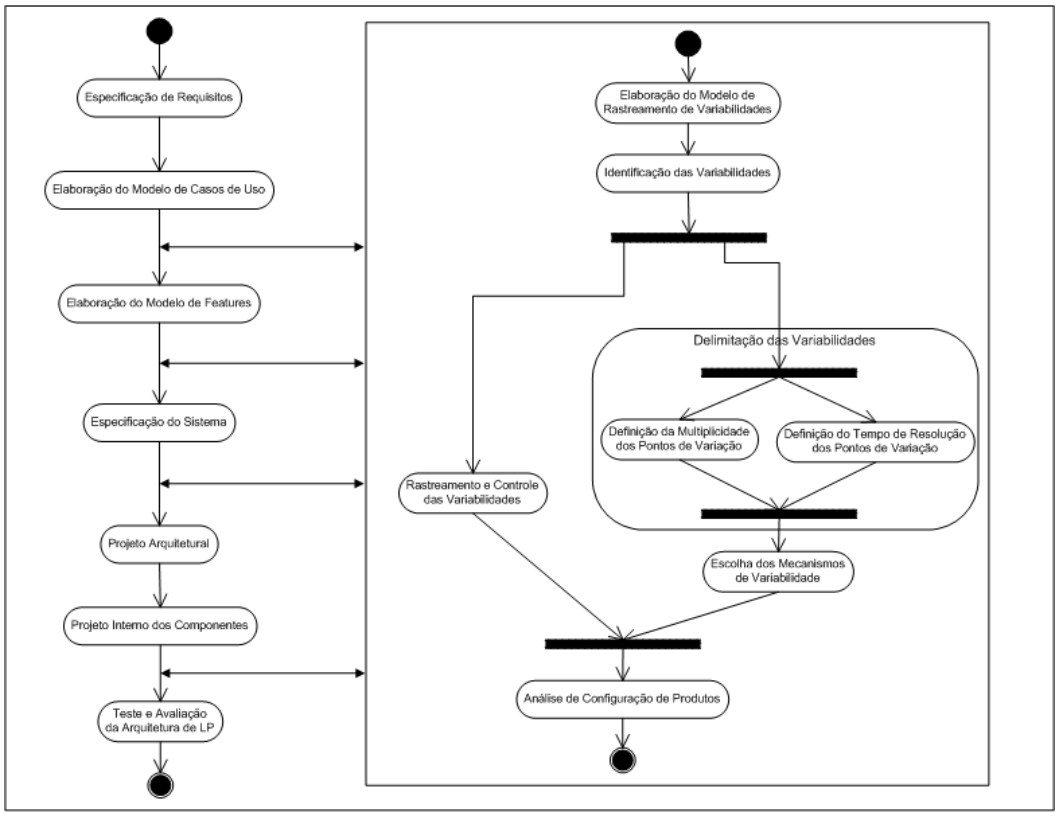

Figura 7: Interação entre as Atividades de Desenvolvimento de LP e as de Gerenciamento de Variabilidade (OLIVEIRA JUNIOR, 2005).

As seções a seguir apresentam uma breve descrição das atividades que formam o processo de gerenciamento de variabilidade. Maiores detalhes sobre as atividades do processo de gerenciamento de variabilidade podem ser encontrados em Oliveira Junior (2005).

3.2.1 Elaboração do Modelo de Rastreamento de Variabilidades

O modelo de rastreamento permite relacionar as variabilidades entre todos os artefatos de uma LP por meio da definição de quais casos de uso estão relacionados com quais *features* da LP. van Gurp e Bosch (2001) caracterizam *feature* como sendo uma unidade lógica de comportamento que corresponde a um conjunto de requisitos funcionais e de qualidade. Segundo van Gurp (2000) existe uma relação n:m entre *features* e requisitos. Isto significa que um determinado requisito pode ser aplicado a várias *features* e uma *feature* em particular pode abranger mais de um requisito.

A elaboração do modelo de rastreamento é realizada efetuando-se um cruzamento entre casos de uso e features e obedece à relação n:m entre casos de uso e

features. Assim, um caso de uso pode envolver várias *features* e uma *feature* em particular pode envolver mais de um caso de uso.

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

1. listar todas as *features* da LP;
2. listar todos os casos de uso da LP;
3. para cada *feature* listada, identificar os casos de uso que originam tal *feature*;
4. marcar com o símbolo "•" o cruzamento entre as *features* e os casos de uso relacionados.

Com este modelo, é possível, a partir de uma determinada *feature*, rastrear as variabilidades até chegar aos casos de uso e aos demais elementos da UML como, por exemplo, outros casos de uso, classes e componentes.

Dessa forma, é possível identificar quais são os casos de uso e os demais artefatos da LP afetados durante a escolha de determinadas *features* para um produto específico. Isto permite, por exemplo, simular a configuração de produtos específicos a partir do modelo de *features* da LP e analisar o impacto nos demais artefatos ao adicionar ou remover determinada *feature*.

3.2.2 Identificação das Variabilidades

Esta atividade tem como objetivo identificar as variabilidades de uma LP em diagramas de casos de uso, diagramas de classes e diagramas de componentes. Além disso, essa atividade visa, também, a representação gráfica das variabilidades.

Esta atividade tem como artefato de entrada o modelo de casos de uso, além do diagrama de classes e de componentes da LP. É realizada percorrendo-se os artefatos citados acima em busca de pontos de variação.

A identificação de pontos de variação em diagramas de casos de uso depende da habilidade do gerente de LP em reconhecer quais são os casos de uso que podem representar pontos de variação e quais casos de uso podem representar as variantes de um ponto de variação. Para isso, algumas orientações podem ser seguidas:

- as variantes do tipo alternativas inclusivas ou exclusivas são normalmente identificadas através da relação com o estereótipo <<extends>>. Assim, os casos de uso, que representam variantes, são identificados com um dos

estereótipos <<alternative_OR>> ou <<alternative_XOR>> enquanto os pontos de variação são representados com o estereótipo <<variationPoint>>. Por exemplo, na Figura 8, pode-se identificar o caso de uso "Gerar Nota Fiscal" como um ponto de variação e os casos de uso "Gerar Nota Fiscal *Online*" e "Gerar Nota Fiscal Impressa" como variantes alternativas inclusivas indicando que zero ou mais destas variantes podem ser escolhidas para resolver o ponto de variação;

- as variantes dos tipos opcionais e obrigatórias são normalmente identificadas como uma simples relação de associação. Por exemplo, na Figura 8, pode-se identificar o caso de uso "Incluir Produto na Cesta" como opcional, uma vez que um produto de software pode ou não conter a inclusão de um produto na cesta de compras, e o caso de uso "Aprovar Pedido" como obrigatório;
- existem casos em que é necessário representar as relações entre variantes, sendo estes quando uma variante exige a existência de outra e quando uma variante exige que outra variante não exista. Estas relações são representadas sempre através da relação de dependência da UML, com os estereótipos <<requires>> e <<mutex>>, respectivamente.

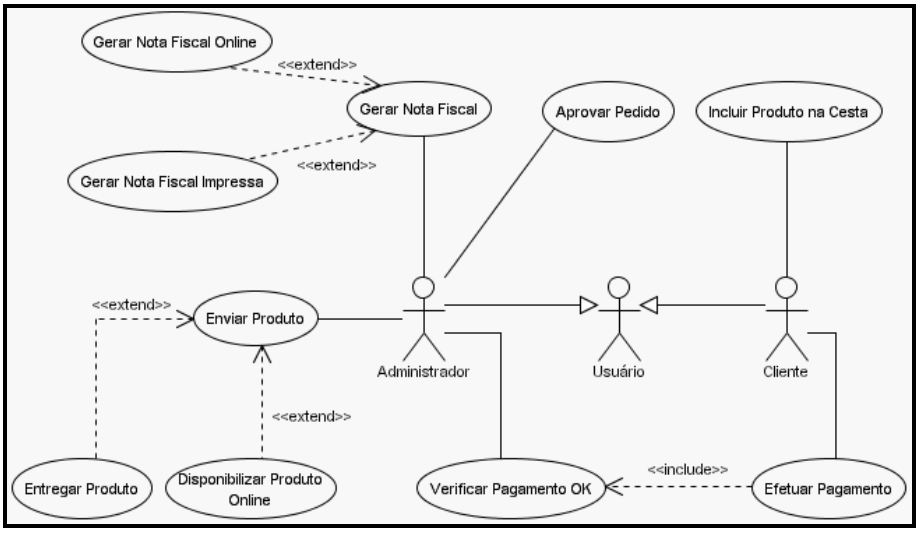

Figura 8: Modelo de Casos de Uso de um Processo de Compra via Internet (OLIVEIRA JUNIOR, 2005).

O mesmo acontece na identificação de pontos de variação em diagramas de classes, em que normalmente classes com a relação de herança representam variantes alternativas inclusivas ou exclusivas, enquanto classes com outros tipos de relações da

UML (associação, agregação, composição e dependência) representam variantes do tipo opcionais ou obrigatórias. Um exemplo poderia ser uma classe "TipoEmbalagem" como sendo um ponto de variação e tendo como variantes alternativas inclusivas as classes "EmbalagemPapel" e "EmbalagemAcolchoada".

A identificação de variabilidades em diagramas de componentes depende do conhecimento e experiência do gerente de LP e das classes que formam o componente. Os componentes formados por classes que possuem variação são também considerados como pontos de variação. Um exemplo poderia ser o componente "GerenciadorEnvio", formado pela classe "TipoEmbalagem", que possui duas variantes, o componente "GerenciadorEntregaOnline" e o componente "GerenciadorEntregaDomicílio".

A representação gráfica das variabilidades em diagramas de casos de uso, diagramas de classes e diagramas de componentes tem como base alguns trabalhos como o de Clauß (2001a), o de Bühne, Halmans e Pohl (2003), o de Gomaa e Webber (2004), e o de Gomaa (2005), e segue a tabela apresentada no Anexo I.

A Figura 9 apresenta um exemplo de identificação de variabilidades em diagrama de classes para um processo de compra via Internet.

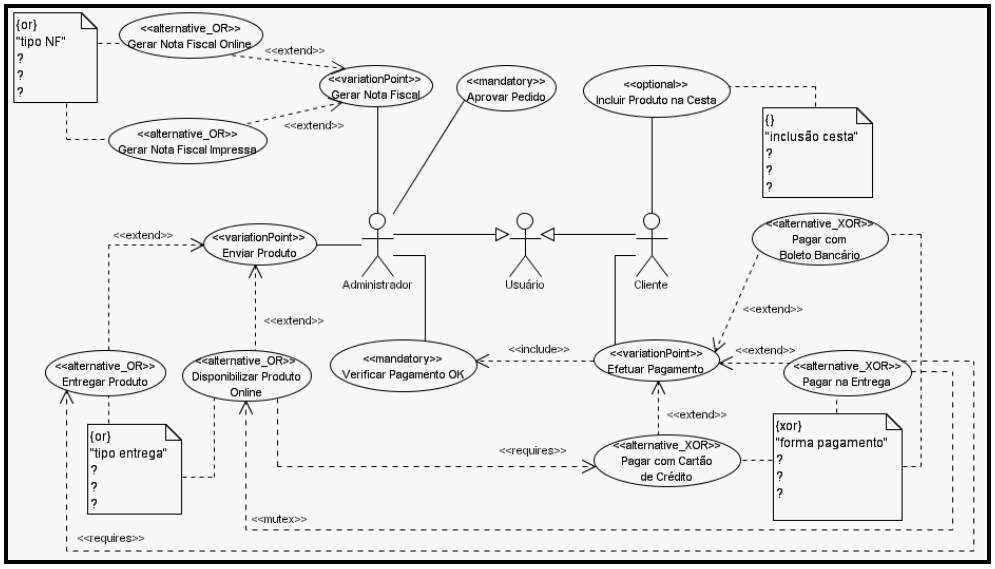

Figura 9: Identificação de Variabilidades em Modelo de Casos de Uso de um Processo de Compra via Internet (OLIVEIRA JUNIOR, 2005).

Esta representação exige a evolução dos artefatos (diagramas de casos de uso, diagramas de classes e de componentes). Assim, tais artefatos devem ser modificados no sentido de incluir elementos que possibilitem a representação de variabilidades.

Além dos estereótipos propostos para representar variabilidades, são usadas notas da UML para representar informações adicionais sobre os pontos de variação e suas variantes como, por exemplo, o nome da variação e o seu tipo, sendo que:

- {} significa que a variante é obrigatória ou opcional;
- {or} significa que a variante é alternativa inclusiva;
- {xor} significa que a variante é alternativa exclusiva.

Nota-se, também, a existência de três "?". Estes símbolos indicam que as variabilidades ainda não foram delimitadas, o que será feito na próxima atividade do processo.

A representação de variabilidade em diagramas de componentes utiliza os mesmos estereótipos utilizados em casos de uso e classes, porém o estereótipo <<variationPoint>> é substituído pelo estereótipo <<variable>>, uma vez que um componente é formado por classes que possuem pontos de variação. Assim, diz-se que o componente é variável ao invés de dizer que o componente é um ponto de variação.

3.2.3 Delimitação das Variabilidades

Esta atividade tem como objetivo definir: a multiplicidade de cada ponto de variação; o tempo de resolução de cada ponto de variação; 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.

A multiplicidade de um ponto de variação indica quantas variantes do conjunto associado ao ponto de variação, no mínimo, devem ser escolhidas. A definição da multiplicidade deve considerar o tipo de relação existente entre o ponto de variação e as suas variantes.

Pontos de variação com **relações do tipo opcional** sempre terão a sua multiplicidade definida como 0 (zero), uma vez que a variante pode ou não ser escolhida.

Pontos de variação com **relações do tipo obrigatória** sempre terão suas multiplicidades definidas como 1 (um).

Assim como as relações obrigatórias, os pontos de variação com **relações do tipo alternativa exclusiva** sempre terão sua multiplicidade definida como 1 (um), pois, dentre as possíveis variantes, uma e somente uma deverá ser escolhida.

Pontos de variação com **relações do tipo alternativa inclusiva** podem variar sua multiplicidade de 0 (zero) até o número de variantes associadas ao ponto de variação, pois 0 (zero) ou mais variantes podem ser escolhidas.

A Tabela 1 apresenta um resumo da multiplicidade de acordo com o tipo de relação existente entre pontos de variação e suas variantes.

Tabela 1: Resumo Relação/Multiplicidade para Pontos de Variação e Variantes (OLIVEIRA JUNIOR, 2005).

	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

O tempo de resolução de um ponto de variação indica em que momento as variantes do conjunto associado ao ponto de variação serão escolhidas.

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. Um exemplo, seria incorporar um novo módulo disponível durante a execução do programa.

A definição do tempo de resolução dos pontos de variação é fundamental na escolha dos mecanismos de implementação de variabilidades.

A representação gráfica da multiplicidade e do tempo de resolução de pontos de variação é realizada nos diagramas de casos de uso, diagramas de classes e diagramas de componentes.

Para isto, utiliza-se a nota da UML, presente nos diagramas provenientes da atividade anterior. Nesses diagramas, as notas da UML contêm o tipo da relação existente ({}, {or} ou {xor}), o nome da variação e três "?". Os símbolos "?" são utilizados nesta atividade para indicar, respectivamente, a multiplicidade do ponto de variação, o tempo de resolução do ponto de variação e se o ponto de variação permite a adição de mais variantes (*true* ou *false*), como na Figura 9.

3.2.4 Escolha dos Mecanismos de Implementação de Variabilidade

Esta atividade tem como objetivo 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.

Os artefatos que servem de entrada para esta atividade são os diagramas de classes e os diagramas de componentes com suas variabilidades representadas e delimitadas.

Um modelo de implementação de variabilidades é gerado como saída desta atividade. Este modelo é construído com base nas técnicas de implementação de variabilidades propostas por Svahnberg, van Gurp e Bosch (2002). Estas técnicas se baseiam nos mecanismos de implementação de variabilidade identificados por Jacobson, Griss e Jonsson (1997) e Anastasopoulos (2001) como herança, extensão, parametrização, configuração e geração.

3.2.5 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.

3.2.6 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 *features* são escolhidas para o produto.

A partir do modelo de *features* de uma LP, criam-se diferentes configurações de produtos para a LP escolhendo quais as *features* necessárias para tais produtos. Assim, para cada produto especificado, deve-se analisar o impacto na adição, modificação ou remoção de cada *feature* com relação aos demais artefatos da LP. Isto é possível utilizando-se o modelo de meta-dados do processo e o modelo de rastreamento de variabilidades.

Dessa forma, é possível comparar os produtos e verificar se a LP precisa ser modificada para atender à exigência dos novos produtos. Além disso, é possível, a partir das especificações dos produtos e dos dados de cada uma delas, realizar análises estatísticas, bem como realizar experimentos formais.

Porém, para alcançar o resultado esperado desta atividade, é fundamental a existência de uma ferramenta automatizada de suporte ao gerenciamento de variabilidades que permita realizar as especificações de uma LP e dos seus produtos. O projeto e a implementação de uma ferramenta para apoiar esta atividade estão fora do contexto deste trabalho, podendo ser realizados como um trabalho futuro.

4. Variabilidade no Ambiente TIDIA-Ae

O TIDIA-Ae é um ambiente que visa o desenvolvimento de ferramentas de suporte ao ensino e aprendizagem por meio da interação de pessoas ou remotamente de forma síncrona ou assíncrona. Dentre os seus objetivos principais, pode-se citar a pesquisa e o desenvolvimento de tecnologia de informação, visando a especificação e desenvolvimento de ferramentas *open-source* para aprendizagem eletrônica (KUDO et al., 2005).

O ambiente TIDIA-Ae foi concebido por meio da colaboração de vários pesquisadores resultando em um ambiente extensível e integrado considerando sua interface, tecnologias e a estrutura do seu código-fonte.

No contexto do TIDIA-Ae, o *Learning Management System* (LMS) é o núcleo. Ele fornece ferramentas que mapeiam e dão suporte a atividades de aprendizagem, participantes com a definição de cargos e permissões, conteúdo, objetos e comunicação síncrona e assíncrona. A Figura 10 apresenta o modelo conceitual do LMS, obtido utilizando-se o processo *UML Components* (CHESSMAN; DANIELS, 2001) para a sua especificação.

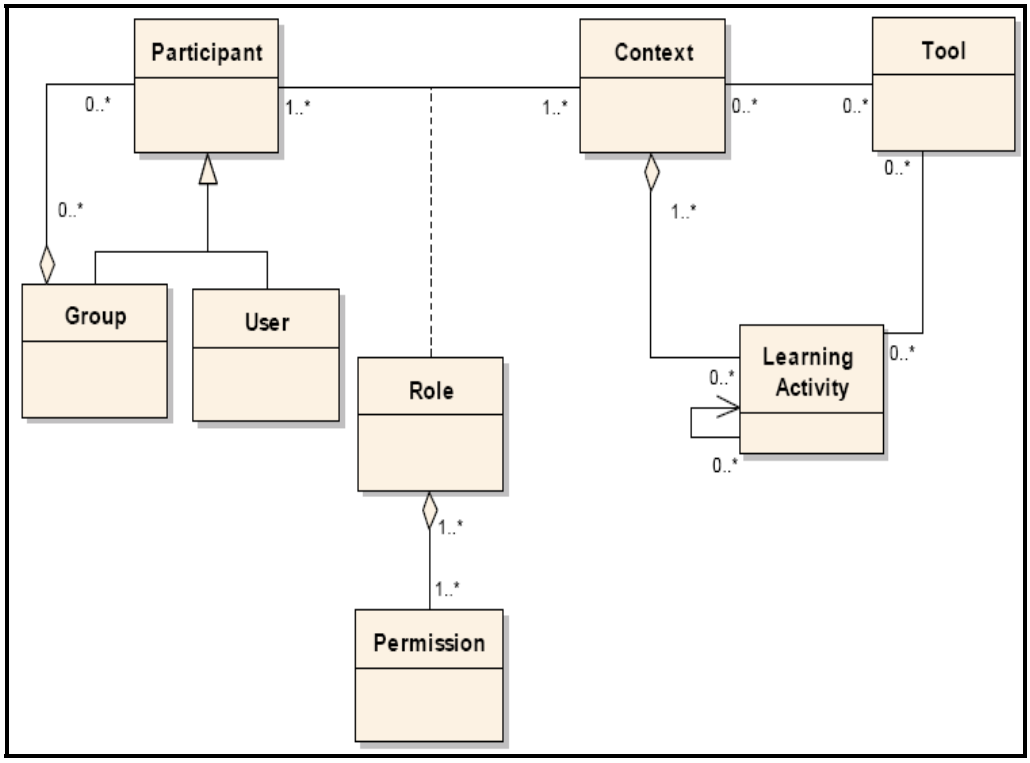

Figura 10: Modelo Conceitual do LMS (SILVA, 2005).

A seguir é apresentada uma descrição de cada um dos elementos do modelo conceitual da Figura 10 (SILVA, 2005).

Uma atividade de aprendizagem (***Learning Activity***) é uma atividade que pode ser proposta em um contexto, podendo ser composta por outras atividades. Um contexto (***Context***) é um espaço criado no ambiente por um usuário com permissões para tal ação. Usuário (***User***) é o nome dado à pessoa que utiliza o ambiente, possuindo um nome e uma senha e podendo pertencer a um grupo. Um grupo de usuários (***Group***) é um conjunto de participantes que pode ser criado, inicialmente, vazio. Um conjunto de permissões (***Permission***) define quais permissões um participante tem para executar determinadas ações em um contexto. Uma ferramenta (***Tool***) é um recurso que permite aos usuários efetuarem diferentes tarefas no ambiente. Um papel (***Role***) define a função de um usuário com relação a um contexto.

Com base no modelo conceitual do LMS, várias ferramentas foram desenvolvidas para apoiar o ambiente TIDIA-Ae, como, por exemplo:

- **Ferramenta de *Chat***: permite a conversa em tempo real por meio da troca de mensagens;
- **Ferramenta *Whiteboard***: semelhante a um mural e tem a função de ser um espaço para que todos os participantes possam disponibilizar informações consideradas relevantes no contexto de um curso.

Neste trabalho foram consideradas as seguintes ferramentas do Núcleo de São Carlos no projeto TIDIA-Ae: ferramenta de *chat*, *whiteboard* e *instant messenger*

As seções a seguir apresentam a aplicação do processo de gerenciamento de variabilidade (seção 3.2) na especificação das ferramentas definidas para este trabalho.

4.1 Variabilidade no Modelo Conceitual do LMS

Antes de iniciar a identificação e delimitação das variabilidades nas ferramentas do ambiente TIDIA-Ae, foi necessário aplicar tais atividades ao modelo conceitual do LMS, pois este representa o núcleo para a especificação e desenvolvimento das ferramentas que compõem o TIDIA-Ae.

A Figura 11 apresenta o modelo conceitual do LMS com variabilidades identificadas, delimitadas e representadas graficamente.

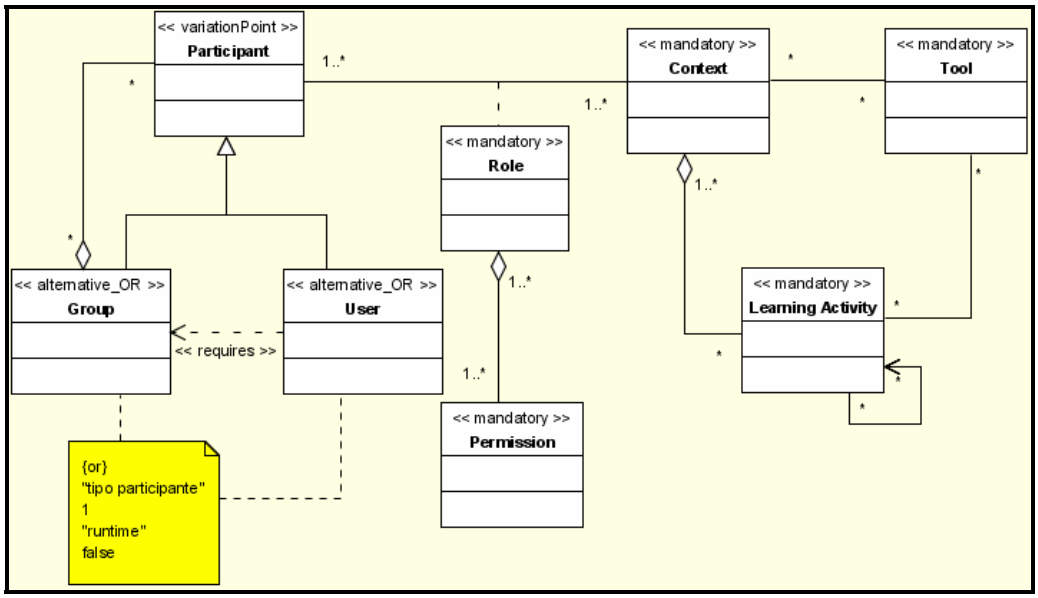

Figura 11: Variabilidades no Modelo Conceitual do LMS.

Nota-se que na Figura 11 a única variabilidade encontrada foi com relação ao tipo de participante, chamada "tipo participante", sendo que o seu ponto de variação é a classe *Participant* e as suas variantes as classes *Group* e *User*.

A relação entre ponto de variação e variantes, para esta variabilidade, foi definida como alternativa inclusiva (*OR*), a multiplicidade com o valor 1 (um), e o tempo de resolução como sendo *runtime*. Além disso, não existe a possibilidade de se incluir mais nenhuma variante ao conjunto associado ao ponto de variação. Resumidamente, do conjunto de variantes associado ao ponto de variação, pelo menos uma variante deve ser escolhida para resolver o ponto de variação em tempo de execução.

Nota-se, ainda, na Figura 11 uma relação entre as variantes *User* e *Group*. Tal relação é representada pelo estereótipo <<requires>> e pela relação de dependência da UML e indica que quando a variante *User* for escolhida para realizar a variabilidade "tipo participante", a variante *Group* obrigatoriamente também deverá ser selecionada. Porém, a recíproca não é verdadeira.

Uma questão importante a ser refletida é a determinação do tempo de resolução de uma variabilidade. Assim, na variabilidade identificada no modelo

conceitual do LMS, se o tempo de resolução tivesse sido definido como sendo *design*, por exemplo, isso poderia afetar a possibilidade de se ter mais de um tipo de usuário para acessar o ambiente em tempo de execução, caso somente uma das variantes fosse escolhida.

Isto evidencia a necessidade de o gerenciamento de variabilidade ser realizado por uma pessoa ou equipe de pessoas que tenham plenos conhecimentos do domínio da linha de produto.

Com base nessas idéias, as seções seguintes apresentam a identificação, delimitação e representação de variabilidades nas ferramentas do TIDIA-Ae.

4.2 Variabilidade na Ferramenta de *Chat*

A especificação da ferramenta de *chat*, assim como as demais, foi realizada com base no processo *UML Components*. Assim, as figuras a seguir apresentam os artefatos gerados durante a especificação da ferramenta e as suas variabilidades.

A Figura 12 apresenta a integração do modelo conceitual da ferramenta de *chat* com o modelo conceitual do LMS. Nota-se que a classe *ChatRoom* é uma especialização de *Tool*, que nesse modelo não aparece representada.

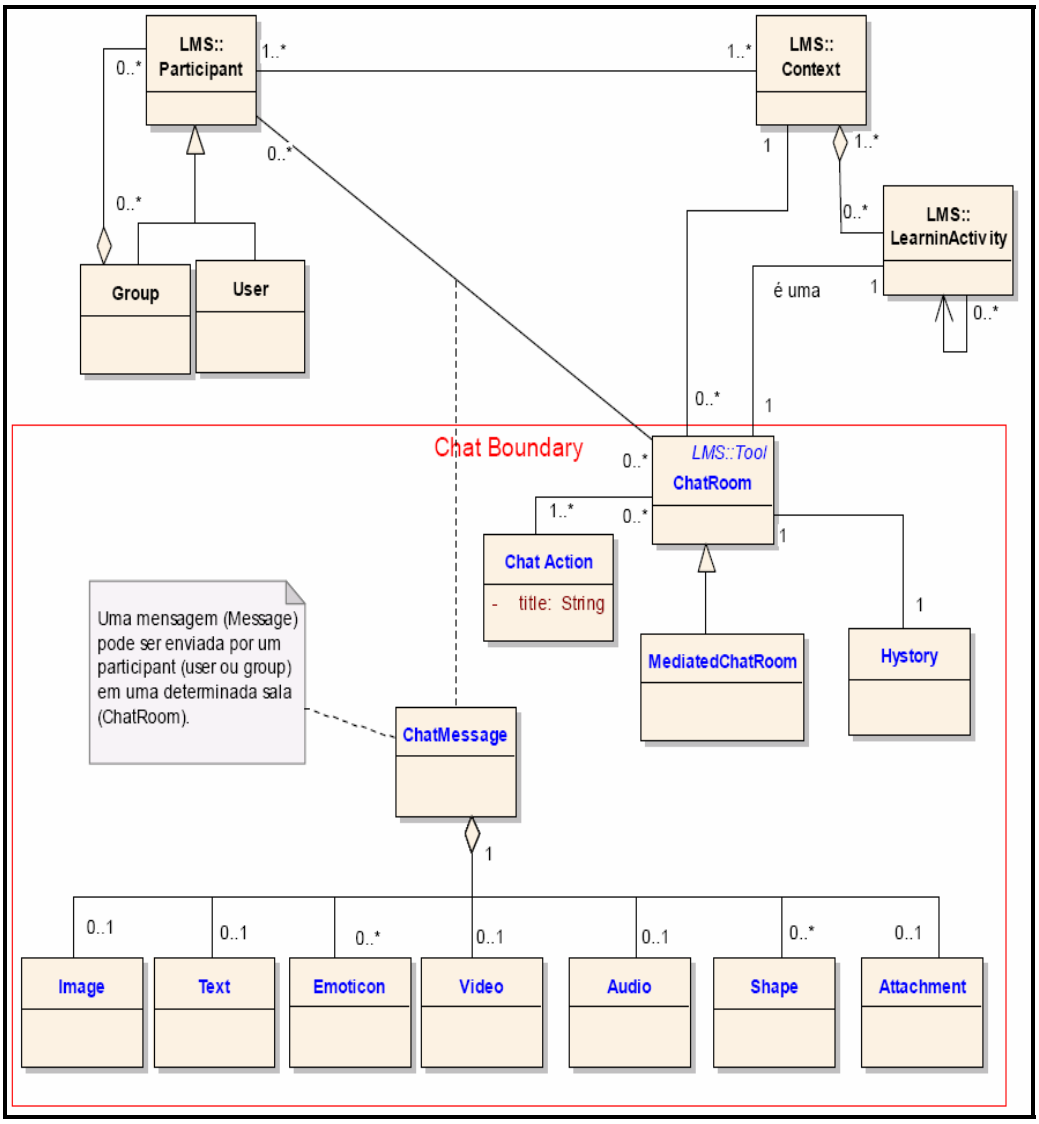

Figura 12: Modelo Conceitual da Ferramenta de *Chat* Integrado ao LMS (SILVA, 2005).

A Figura 13 apresenta o diagrama de casos de uso da ferramenta de *chat*. Para tanto, foram definidos 15 casos de uso, sendo os principais: *gerenciar sala de chat*, *utilizar chat*, *mediar sala de chat*, e *observar sala de chat*.

Os casos de uso são realizados por cinco atores, todos eles especializações do ator *Usuário*, sendo eles: *Coordenador*, *Participante*, *Entrevistado*, *Mediador* e *Obervador*.

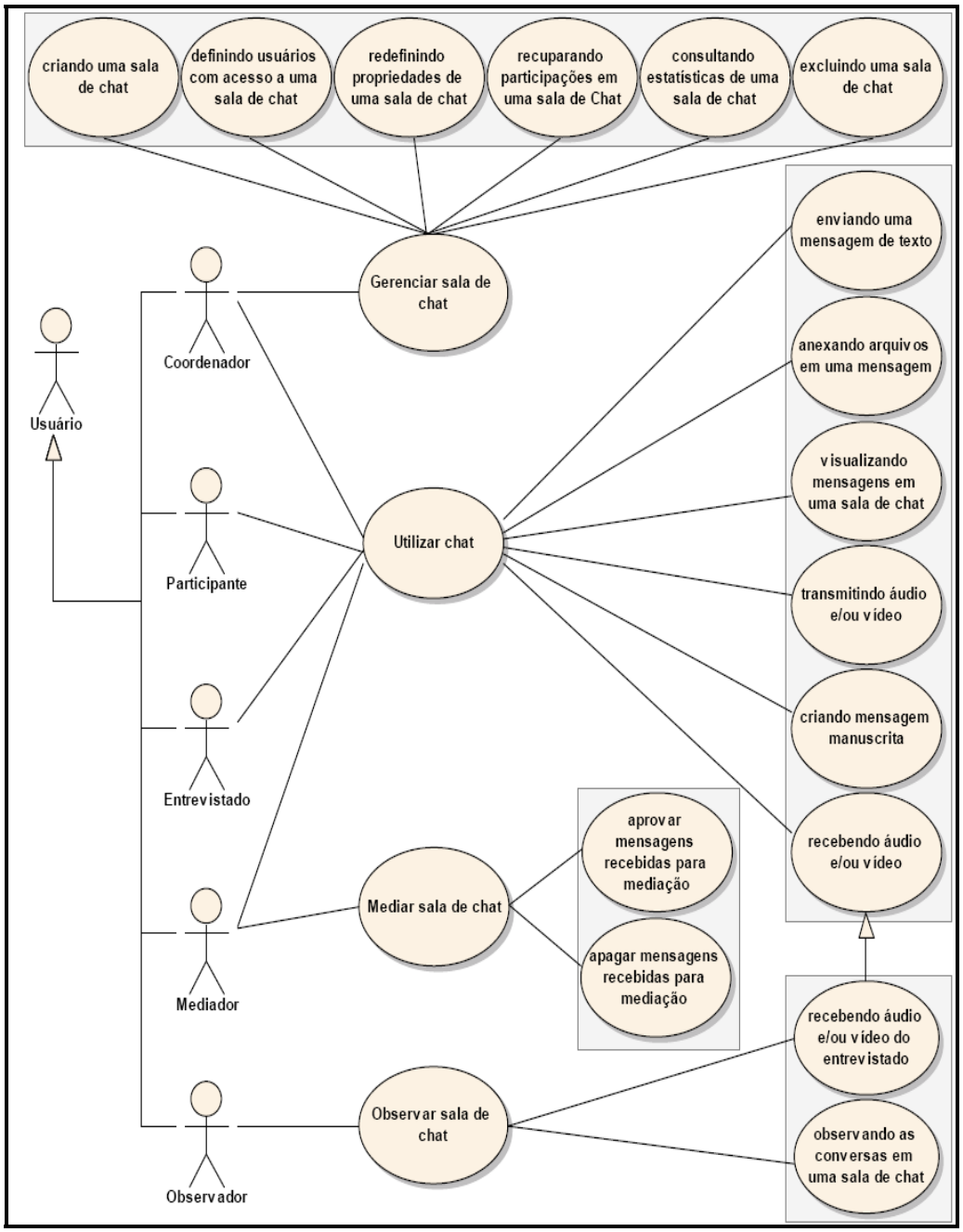

Figura 13: Diagrama de Casos de Uso da Ferramenta de *Chat* (KUDO et al., 2006).

A Figura 14 apresenta o modelo de tipos de negócio da ferramenta de *chat*. Nesse modelo, várias classes e interfaces foram definidas como, por exemplo, *ChatMessage* e todas as suas especializações: *Image*, *Text*, *Video*, entre outras.

A classe mais importante desse modelo é *ChatRoom*, pois é a partir dela que é definida a interface *IChatRoom*, permitindo o contrato entre componentes e a possível utilização dos serviços oferecidos pela ferramenta de *chat*.

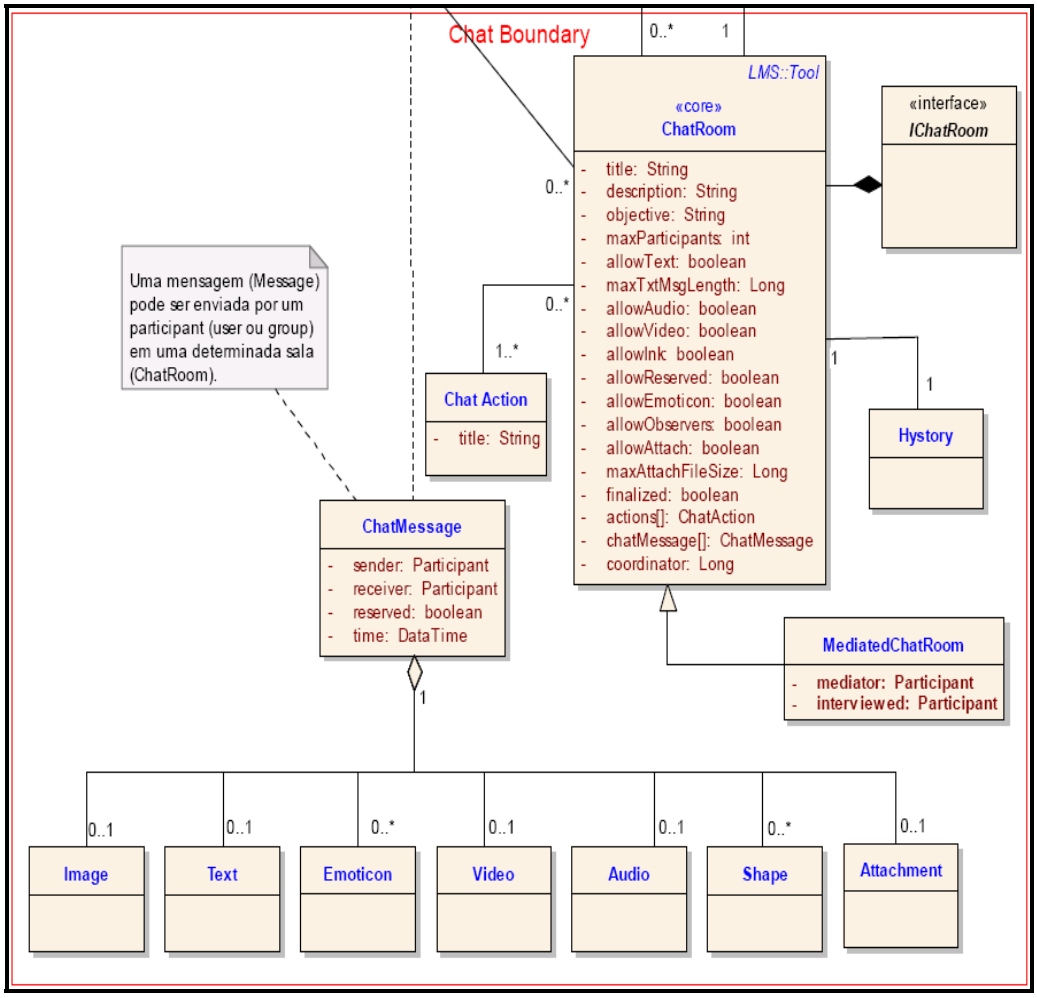

Figura 14: Modelo de Tipos de Negócio da Ferramenta de *Chat* (KUDO et al., 2006).

A Figura 15 apresenta a arquitetura de componentes da ferramenta de *chat* integrada aos componentes do LMS. O componente *ChatRoomMgr* possui a interface *IChatRoomMgt* que fornece a assinatura aos serviços fornecidos por tal componente. É possível notar nesta arquitetura que o componente *ChatRoomMgr* utiliza os serviços oferecidos pelo componente *CommunicationMgr* por meio da interface *ICommunicationMgt*.

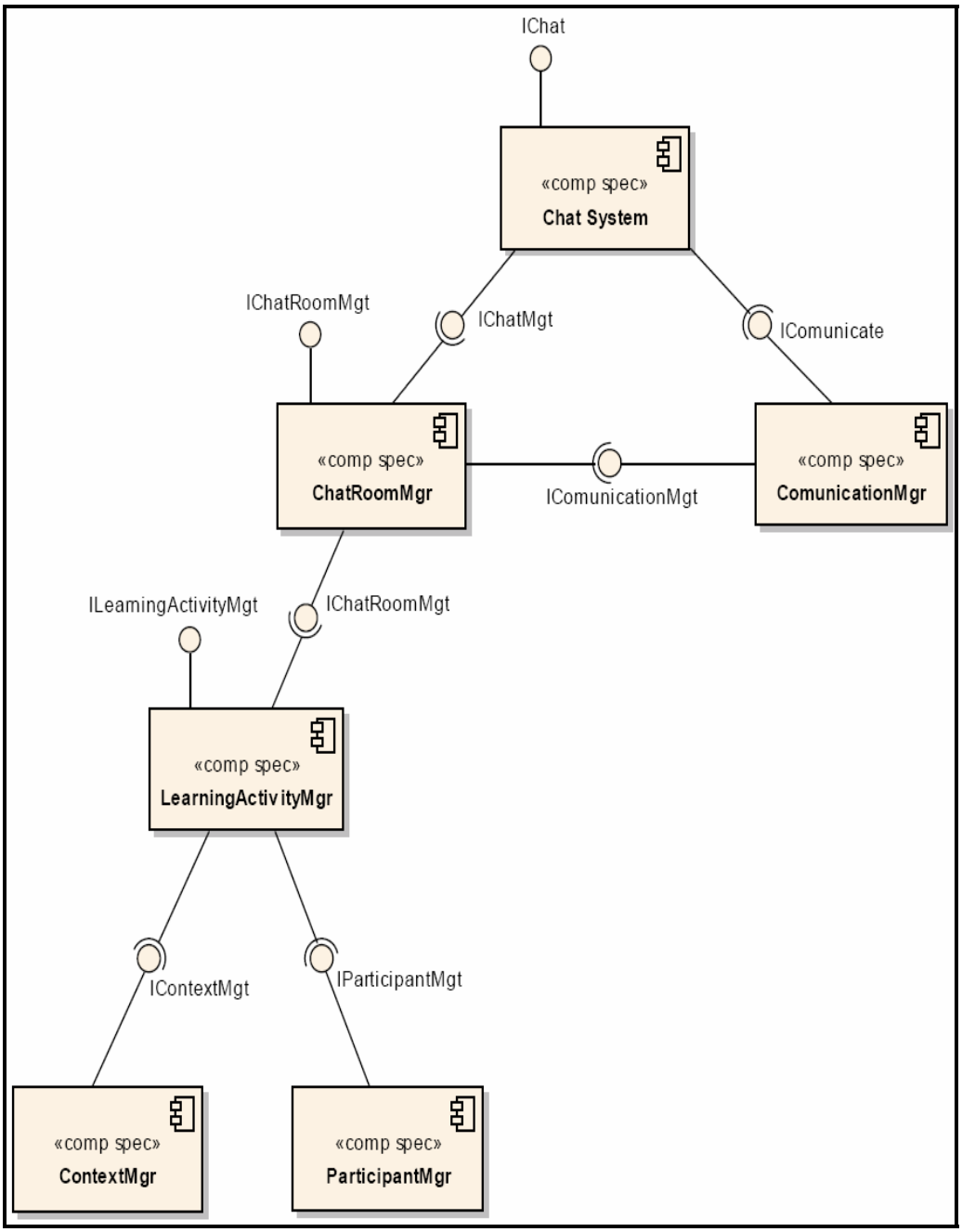

Figura 15: Arquitetura de Componentes da Ferramenta de *Chat* (KUDO et al., 2006).

Com base nos modelos de casos de uso, tipos de negócio e arquitetura da ferramenta de *chat* e no processo de gerenciamento de variabilidade (seção 3.2) foi possível identificar, delimitar e representar graficamente as suas variabilidades.

Dessa forma, as figuras a seguir apresentam os artefaots analisados, bem como as suas variabilidades identificadas e uma breve discussão.

A Figura 16 apresenta o diagrama de casos de uso da ferramenta de *chat* e as suas variabilidades. Nota-se, dessa forma, a existência de 04 variabilidades, sendo elas: o tipo de gerenciamento, a forma de utilização, o tipo de mediação e o tipo de observação de um *chat*.

A principal variabilidade deste artefato é o tipo de gerenciamento de uma sala de *chat*, composta por:

- 01 ponto de variação, que é o caso de uso *Gerenciar Sala de Chat*;
- 06 variantes, sendo elas os casos de uso:
    - *Criando uma Sala de Chat*;
    - *Definindo Usuários com Acesso a uma Sala de Chat*;
    - *Redefinindo Propriedades de uma Sala de Chat*;
    - *Recuperando Participações em uma Sala de Chat*;
    - *Consultando estatísticas de uma Sala de Chat*; e
    - *Excluindo uma Sala de Chat*.

Todas as variantes são alternativas inclusivas, pois mais de uma delas pode ser escolhida para resolver o ponto de variação em tempo de execução, sendo que nenhuma outra variante pode ser adicionada ao conjunto já associado ao ponto de variação, conforme a nota da UML associada às variantes na Figura 16.

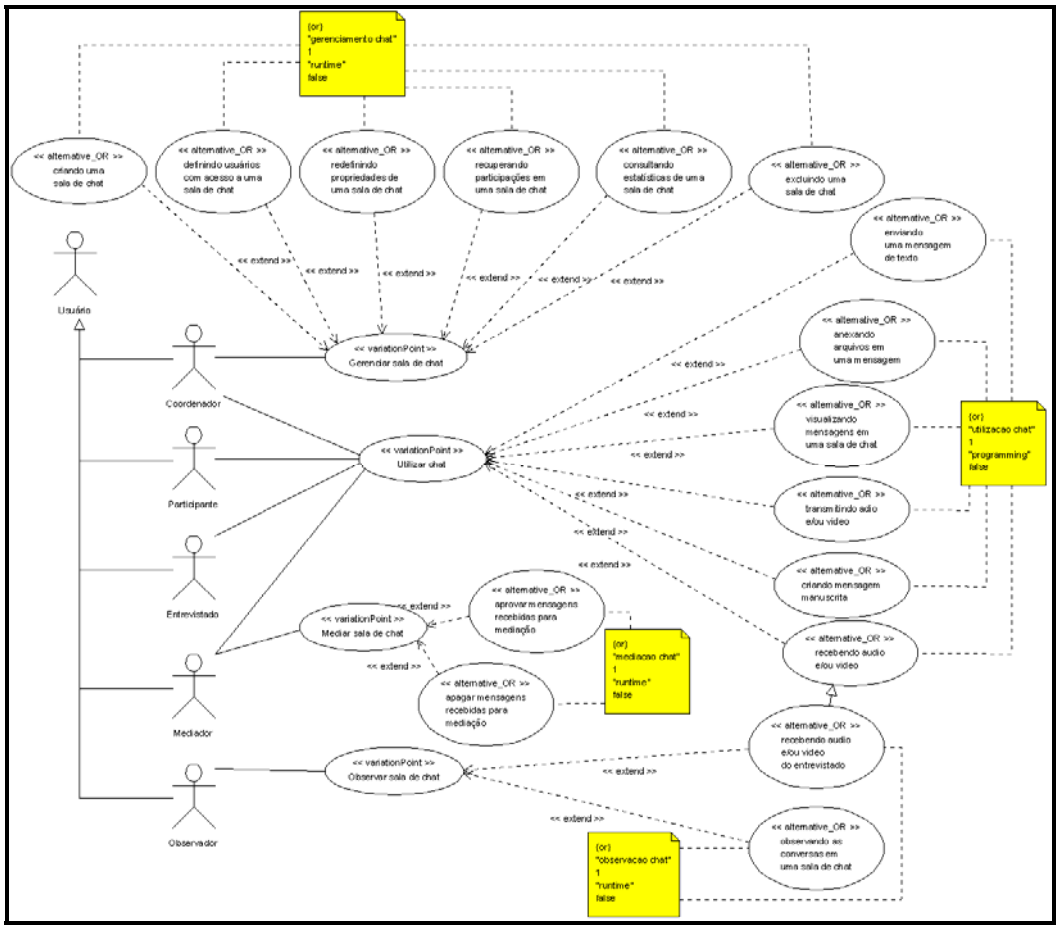

Figura 16: Variabilidades no Diagrama de Casos de Uso da Ferramenta de *Chat*.

A Figura 17 apresenta o modelo de tipos de negócio da ferramenta de *chat* e as suas 09 variabilidades, que são; a ação em um *chat*, o tipo de sala, e a mensagem com imagem, texto, emoticon, vídeo, audio, shape, attachment. A variabilidade relativa ao tipo de participante não é considerada, pois é inerente ao LMS.

A variabilidade relativa à ação em uma sala de *chat*, por exemplo, é composta por um ponto de variação, a classe *ChatRoom*, e uma única variante, a classe *ChatAction*. Tal variabilidade é do tipo opcional, sendo que esta pode ou não estar presente nos produtos finais. Ela é resolvida em tempo de programação e nenhuma outra variante pode ser adicionada ao conjunto associado ao ponto de variação para tal variabilidade.

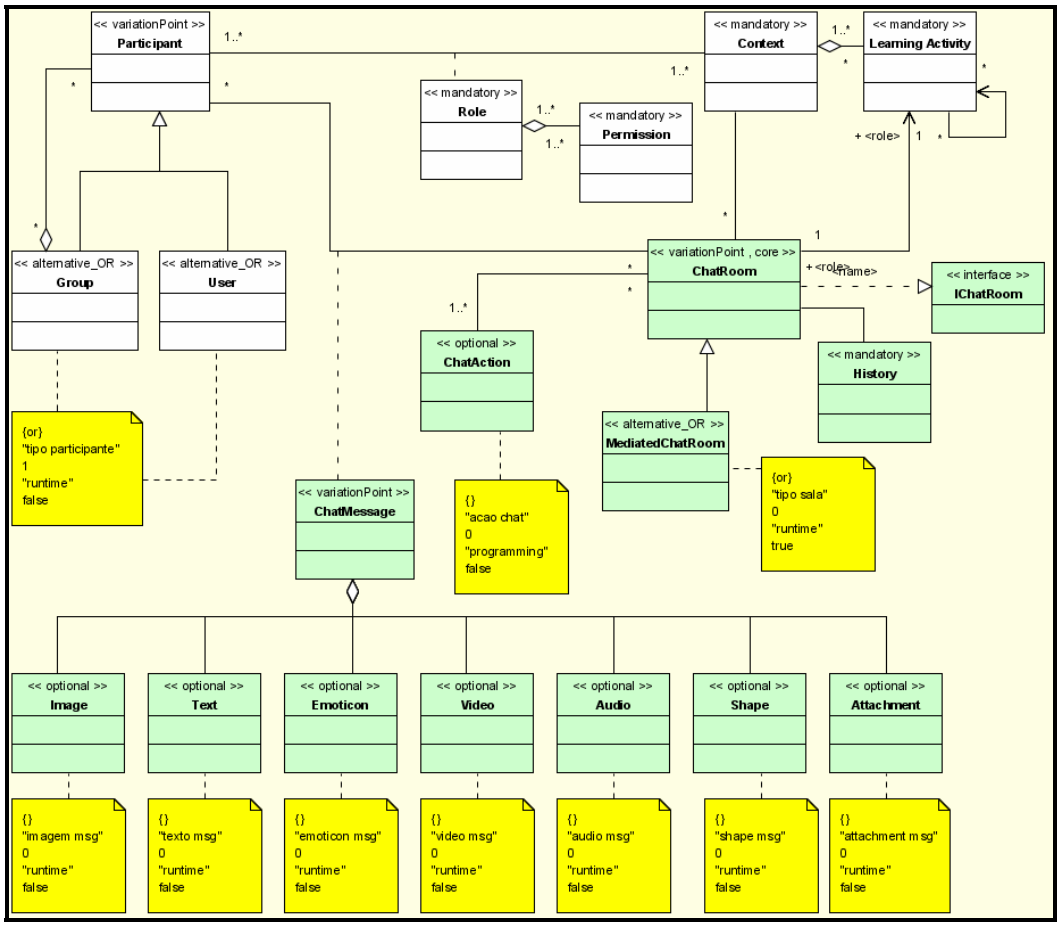

Figura 17: Variabilidades no Modelo de Tipos de Negócio da Ferramenta de *Chat*.

A Figura 18 apresenta a arquitetura de componentes da ferramenta de *chat* e a sua variabilidade com relação ao componente *ChatRoomMgr*. A variabildade relativa ao componente *ParticipantMgr* não é considerada pois é inerente ao LMS.

Assim, o componente *ChatRoomMgr* apresenta variação, uma vez que as classes que o compõe possuem variação, como apresentado na Figura 17.

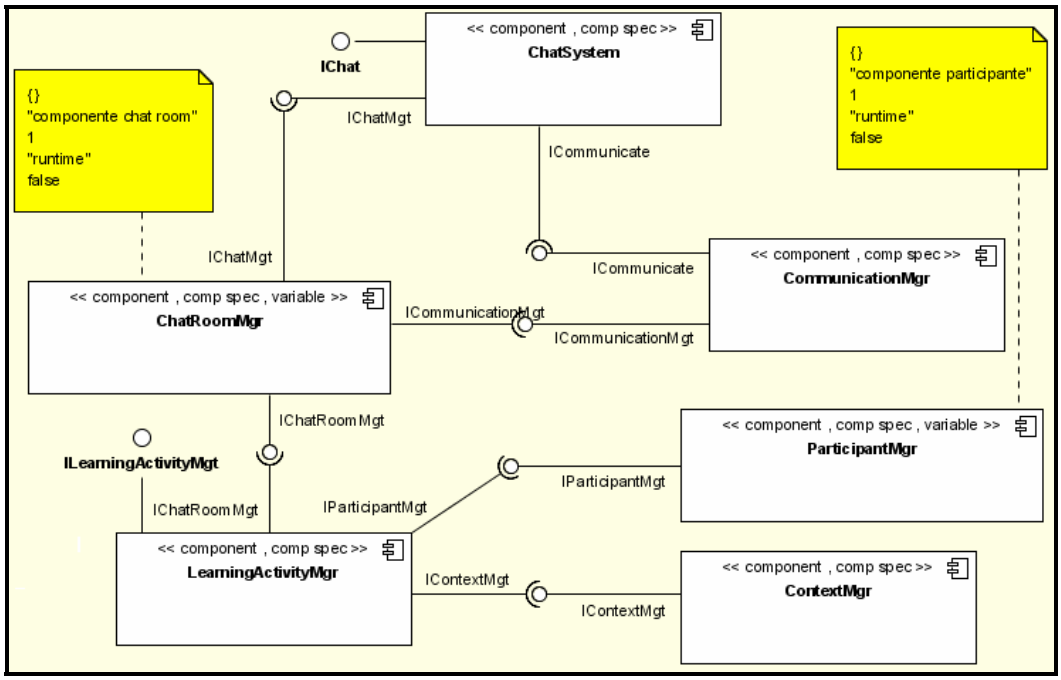

Figura 18: Variabilidades na Arquitetura de Componentes da Ferramenta de *Chat*.

4.3 Variabilidade na Ferramenta *Whiteboard*

A feramenta *whiteboard* foi especificada com base no processo *UML Components*, assim como as demais. A seguir são apresentados os artefatos gerados durante a sua especificação.

A Figura 19 apresenta o modelo conceitual da ferramenta *whiteboard* integrado ao do LMS. O modelo conceitual da ferramenta *whiteboard* é formado por 15 classes principais, todas elas localizadas na parte central e inferior da Figura 19.

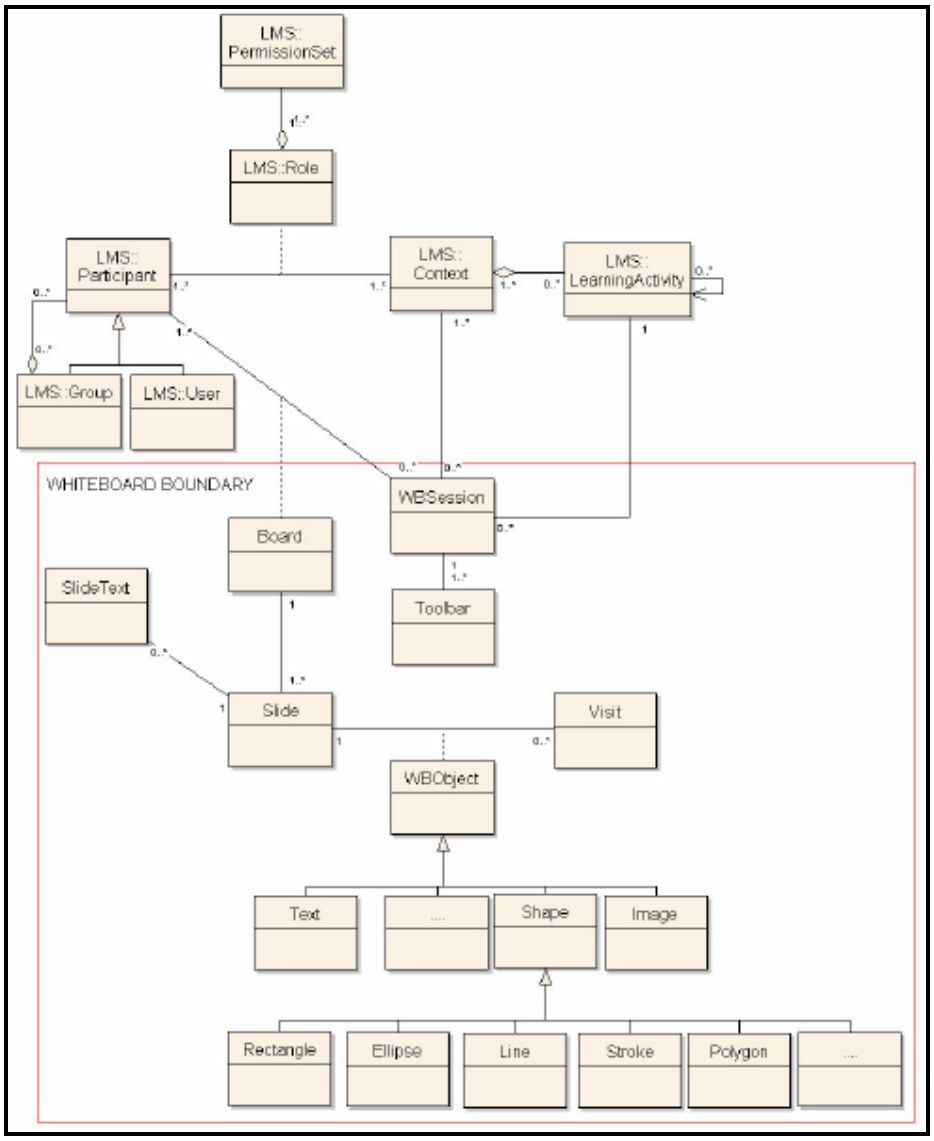

Figura 19: Modelo Conceitual da Ferramenta *Whiteboard* Integrado com o LMS (KUDO; JARDIM, 2006).

A Figura 20 apresenta o diagrama de casos de uso da ferramenta *whiteboard*, formado por 24 casos de uso e 02 atores. Estes casos de uso estão relacionados, principalmente, ao gerenciamento de sessões da ferramenta e ao gerencimento de conteúdo como slides, anotações, imagens, escrita e outros.

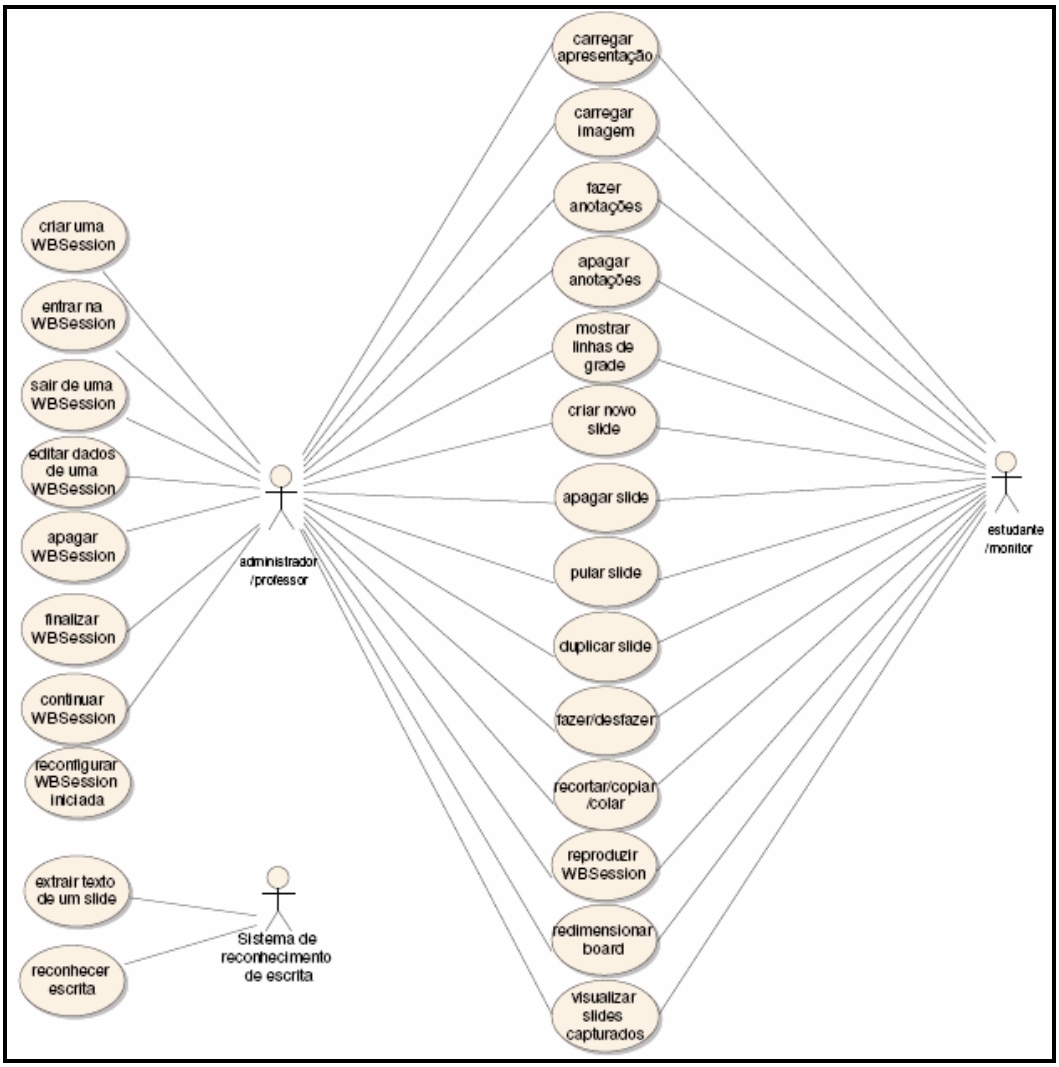

Figura 20: Diagrama de Casos de Uso da Ferramenta *Whiteboard* (KUDO; JARDIM, 2006).

A Figura 21 apresenta o modelo de tipos de negócio da ferramenta *whiteboard*. Este artefato guia a identificação das interfaces e componentes que forma a arquitetura da ferramenta. As principais classes deste modelo são *WBSession* e *WBBoard*.

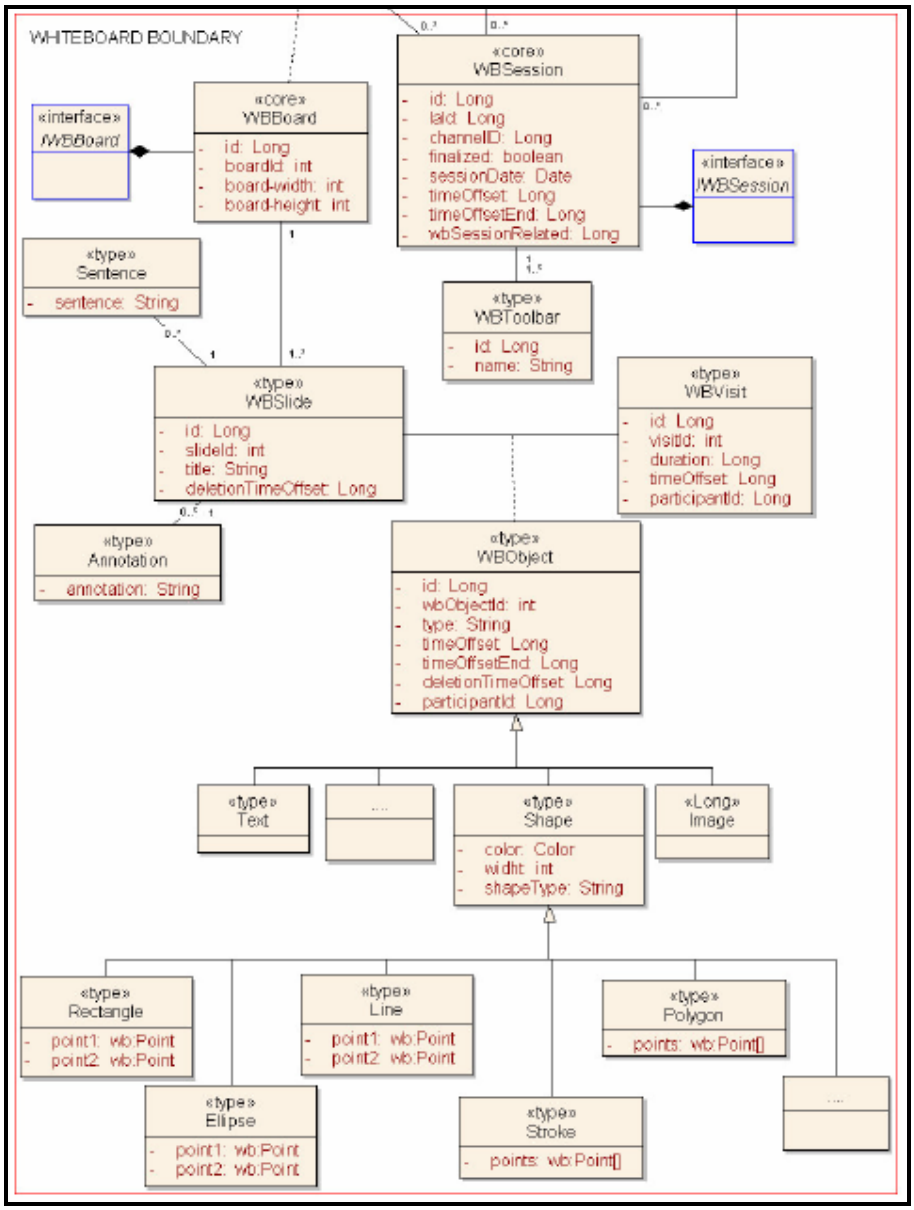

Figura 21: Modelo de Tipos de Negócio da Ferramenta *Whiteboard* (KUDO; JARDIM, 2006).

A Figura 22 apresenta a arquitetura de componentes da ferramenta *whiteboard*, em que os dois componentes principais são: *WBSession* e *Board*. Os demais componentes são provenientes da arquitetuda do LMS.

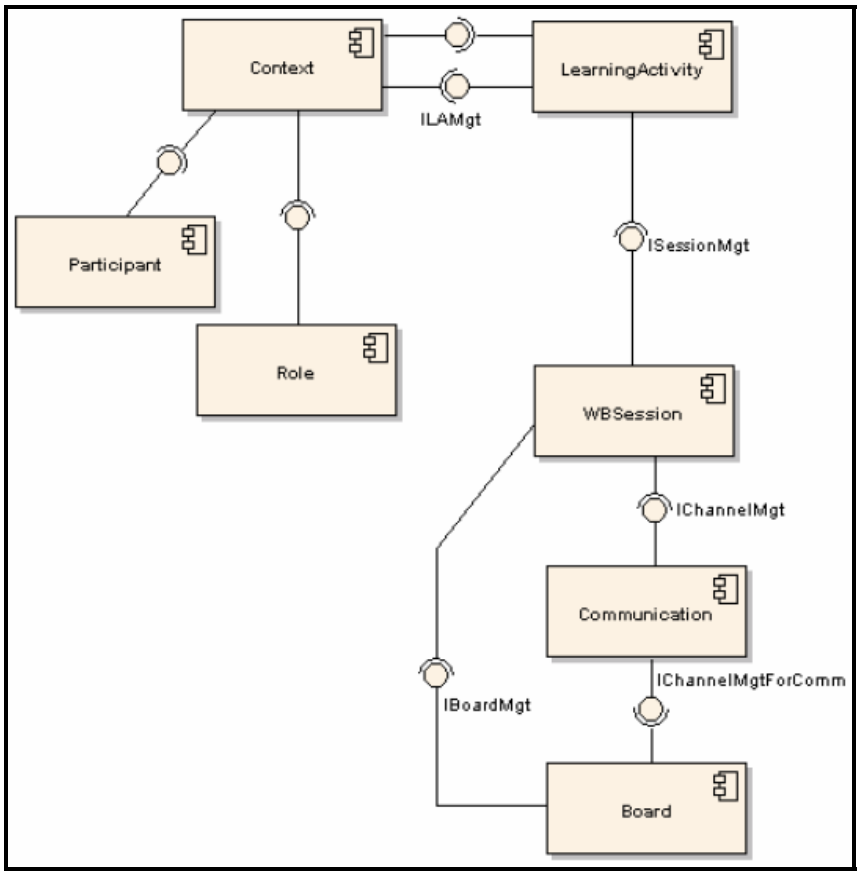

Figura 22: Arquitetura de Componentes da Ferramenta *Whiteboard* (KUDO; JARDIM, 2006).

Com base nos artefatos gerados durante a especificação da ferramenta *whiteboard* e no processo de gerenciamento de variabilidade (seção 3.2) foi possível identificar, delimitar e representar graficamente as variabilidades em tais artefatos.

As figuras a seguir apresentam os artefatos analisados, bem como as suas variabildades e uma breve discussão.

A Figura 23 apresenta o diagrama de casos de uso da ferramenta *whiteboard*, bem como a única variabilidade encontrada, que está relacionada ao gerenciamento de sessões da ferramenta.

Tal variabilidade é composta por:

- 01 ponto de variação: o caso de uso *Gerenciar WBSession*;
- 08 variantes, sendo elas os casos de uso:
    - *Criar uma WBSession*;
    - *Entrar na WBSession*;
    - *Sair de uma WBSession*;

- *Editar Dados de uma WBSession*;
- *Apagar WBSession*;
- *Finalizar WBSession*;
- *Continuar WBSession*;
- *Reconfigurar WBSession Iniciada*.

Um ponto importante a considerar é que neste artefato foi identificada somente uma única variabilidade. Analisando o diagrama de casos de uso (Figura 23) percebemos que os demais casos de uso (lado direito do diagrama) foram modelados com uma associação direta com os atores que os realizam. Dessa forma, seguido as orientações do processo de gerenciamento de variabilidade, tais casos de uso devem ser definidos como obrigatórios. Porém, é possível agrupar tais casos de uso e estabelecer uma relação de dependência destes com um caso de uso de maior granularidade, assim como foi feito com o caso de uso *Gerenciar WBSession*. Assim, o número de variabilidades deste artefato tende a aumentar, o que é desejável ao consider tal domínio propício à abordagem de LP.

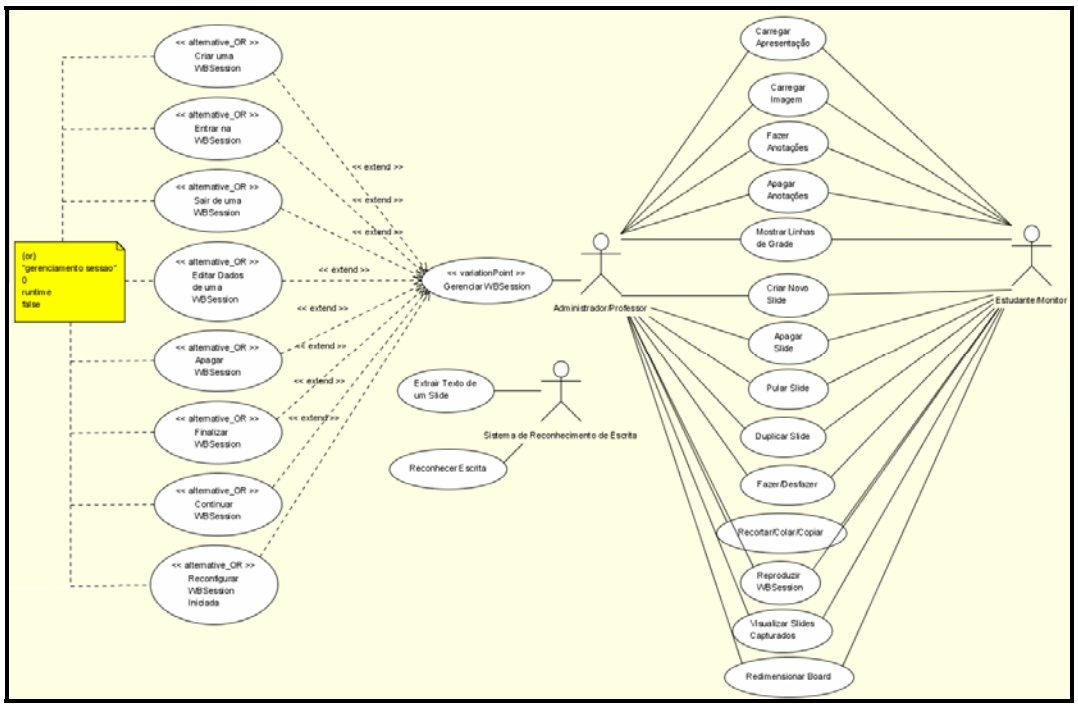

Figura 23: Variabilidades no Diagrama de Casos de Uso da Ferramenta *Whiteboard*.

A Figura 24 apresenta o modelo de tipos de negócio da ferramenta *whiteboard* e as suas 02 variabilidades, sendo elas com relação ao tipo de objeto e ao tipo de figura da ferramenta.

A variabilidade relativa ao tipo de objeto é composta por:

- 01 ponto de variação: a classe *WBObject*;
- 03 variantes, sendo elas as classes: *Text*, *Shape* e *Image*

A relação ponto de variação/variante é do tipo alternatica inclusiva, sendo que zero ou mais variantes do conjunto associado podem ser escolhidas. A variabilidade é resolvida em tempo de execução e não é possível adicionar nenhuma variante ao conjunto associado ao ponto de variação.

A variabilidade relativa ai tipo de figura é composta por:

- 01 ponto de variação: a classe *Shape*;
- 05 variantes, sendo elas as classes: *Rectangle*, *Elipse*, *Line*, *Stroke* e *Polygon*.

Essa variabilidade, assim como o tipo de objeto, possui uma relação ponto de variação/variante do tipo alternativa inclusiva, sendo que zero ou mais variantes do conjunto podem ser escolhidas. Ela é resolvida em tempo de execução e nenhuma variante pode ser adicionada ao conjunto associado.

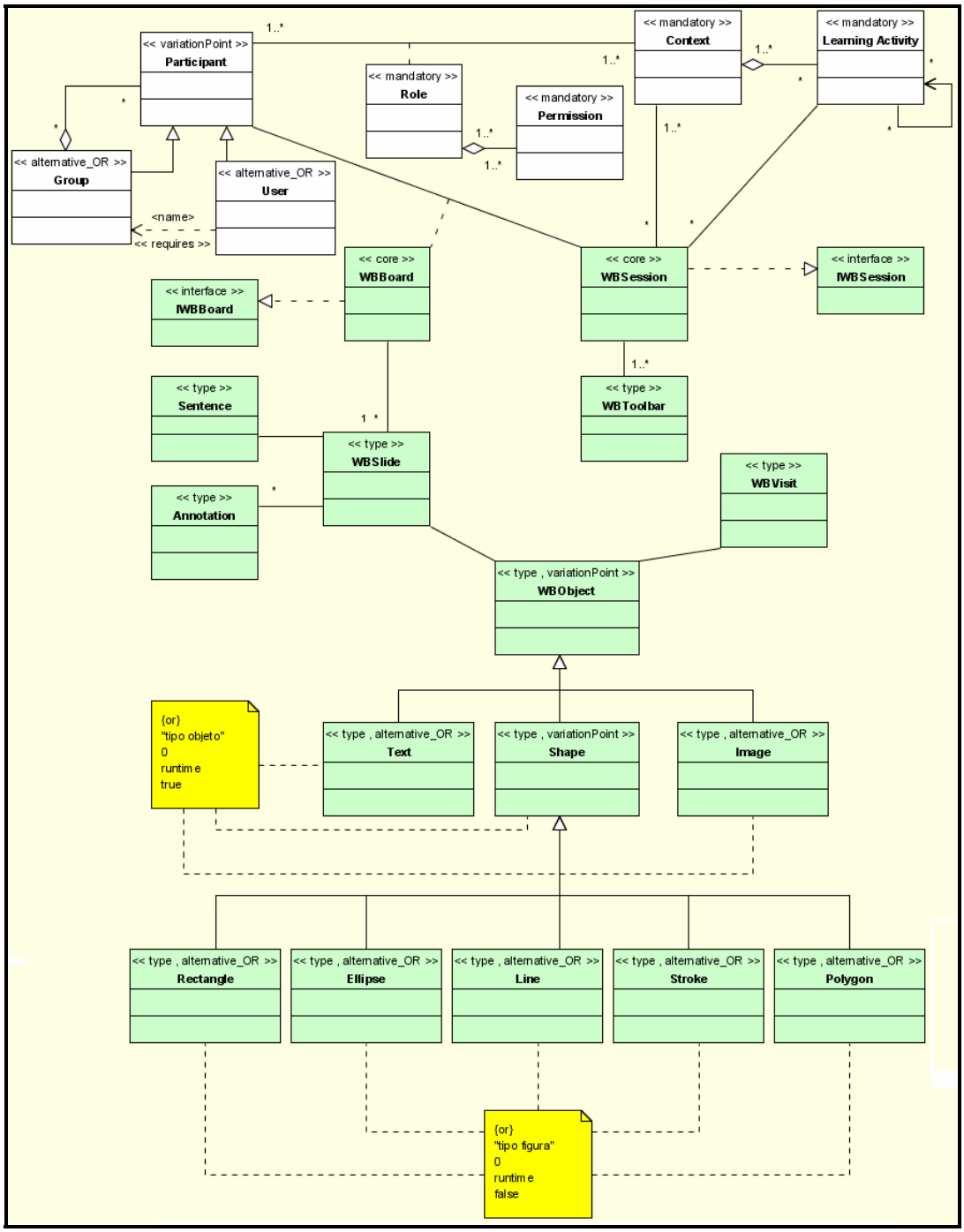

Figura 24: Variabilidades no Modelo de Tipos de Negócio da Ferramenta *Whiteboard*.

A Figura 25 apresenta a arquitetura de componentes da ferramenta *whiteboard* e as suas variabilidades. Assim, como a arquitetura da ferramenta de *chat*, a variação no componente *ParticipantMgr* não é considerada por ser inerente ao LMS.

Dessa forma, o único componente variável é *BoardMgr*, uma vez que as classes que o compõe possuem variação (Figura 24).

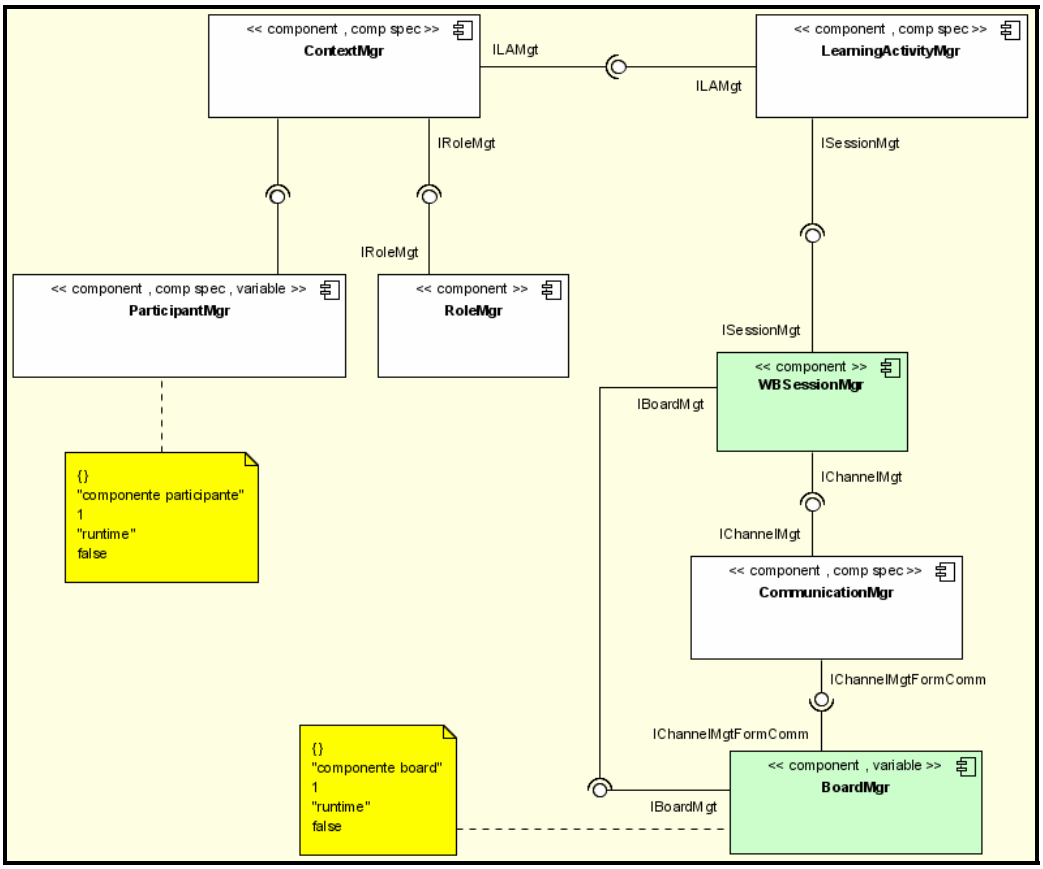

Figura 25: Variabilidades na Arquitetura de Componentes da Ferramenta *Whiteboard*.

4.4 Variabilidade na Ferramenta *Instant Messenger*

Da mesma forma como as demais, a ferramenta *instant messenger* foi modelada seguindo o processo *UML components*. A seguir são apresentados os artefatos gerados durante a sua especificação, as variabilidades encontradas em tais artefatos, bem como uma breve discussão.

A Figura 26 apresenta o modelo conceitual da ferramenta *instant messenger* integrado ao LMS. Nota-se que a classe *Comunicador Instantâneo* representa a especialização da ferramenta *instant messenger*.

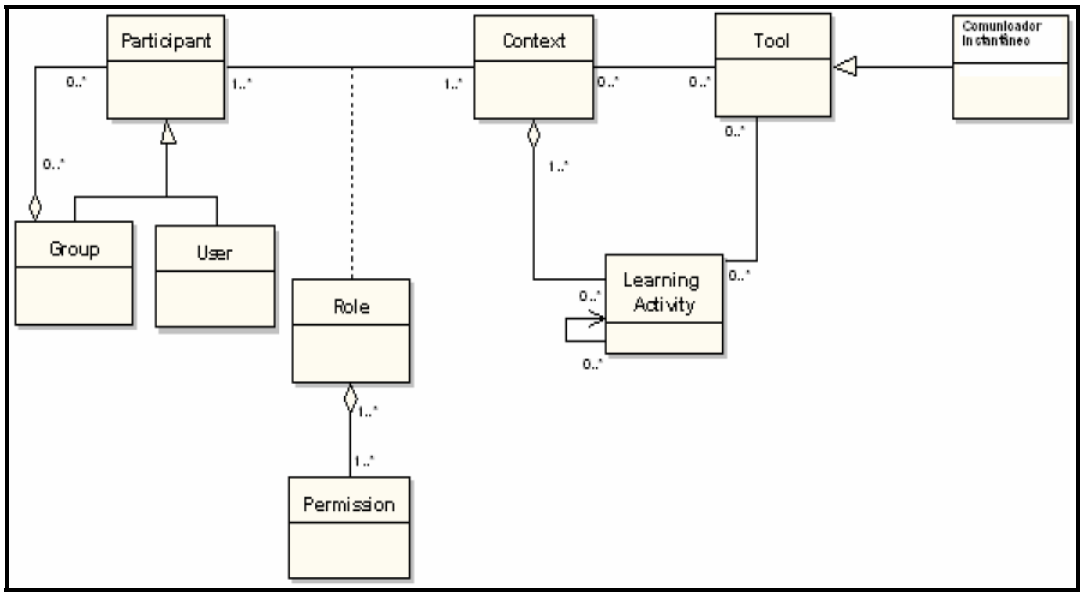

Figura 26: Modelo Conceitual da Ferramenta *Instant Messenger* Integrado com o LMS (PIRES et al., 2005).

A Figura 27 apresenta o modelo conceitual da ferramenta *instant messenger*. Pode-se perceber que a classe *Participant* é proveniente do modelo conceitual do LMS. As demais classes do modelo pertencem diretamente ao modelo conceitual da ferramenta em questão.

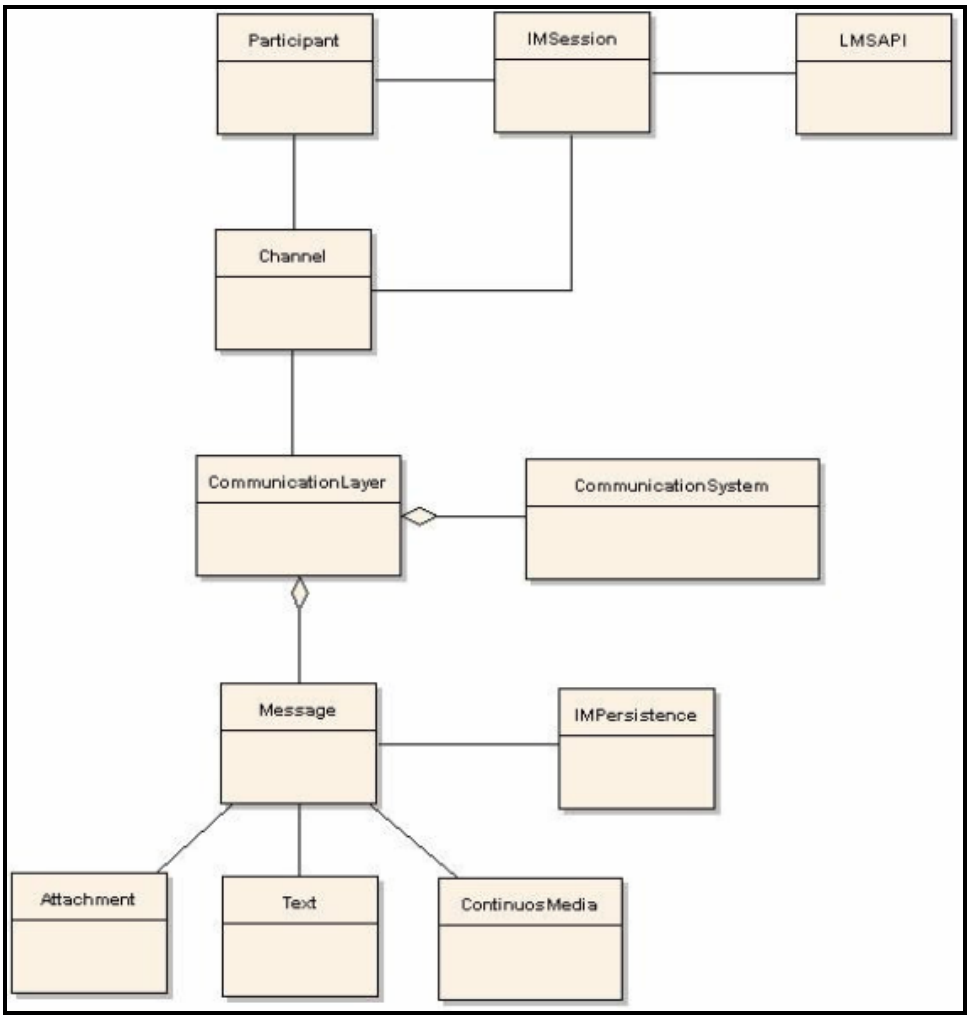

Figura 27: Modelo Conceitual da Ferramenta *Instant Messenger* (PIRES et al., 2005).

A Figura 28 apresenta o diagrama de casos de uso relativo à comunicação na ferramenta *instant messenger*. O caso de uso principal é deste artefato é *Fazer Comunicação*, sendo realizado pelos atores *Usuário* e *Contato*.

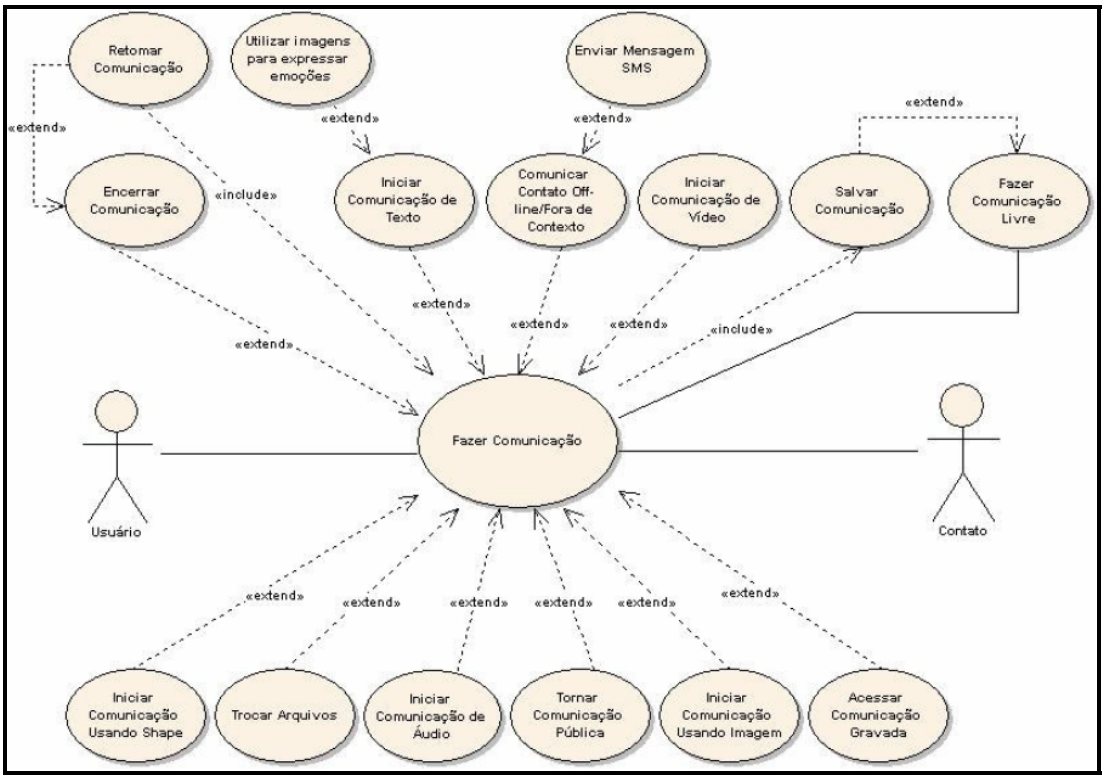

Figura 28: Diagrama de Casos de Uso para Comunicação da Ferramenta *Instant Messenger* (PIRES et al., 2005).

A Figura 29 apresenta o diagrama de casos de uso para o ator *Usuário*, sendo os principais casos de uso: *Administrar Contatos*, *Gerenciar Status do Usuário*, *Alertar Entrada/Saída no Sistema* e *Fazer Comunicação*.

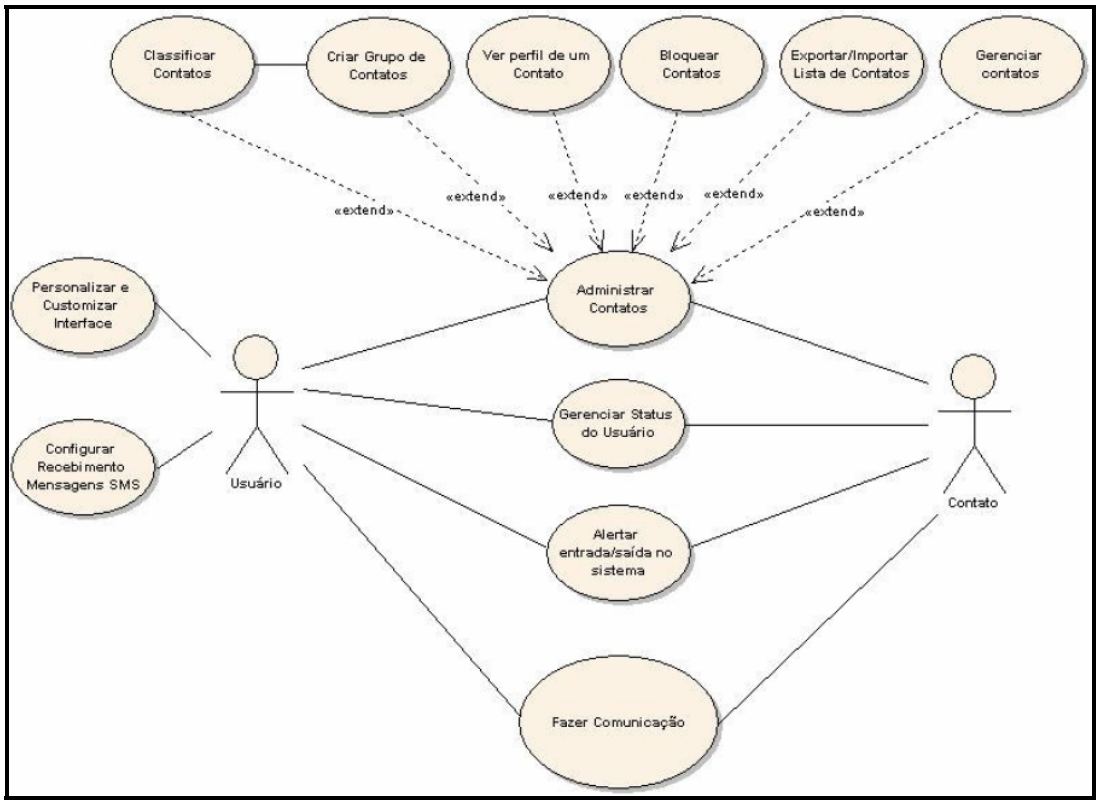

Figura 29: Diagrama de Casos de Uso para o Ator Usuário da Ferramenta *Instant Messenger* (PIRES et al., 2005).

A Figura 30 apresenta o modelo de tipos de negócio da ferramenta *instant messenger*. As principais classes que formam este modelo são: *IMSession*, *CommunicationLayer* e *Message*.

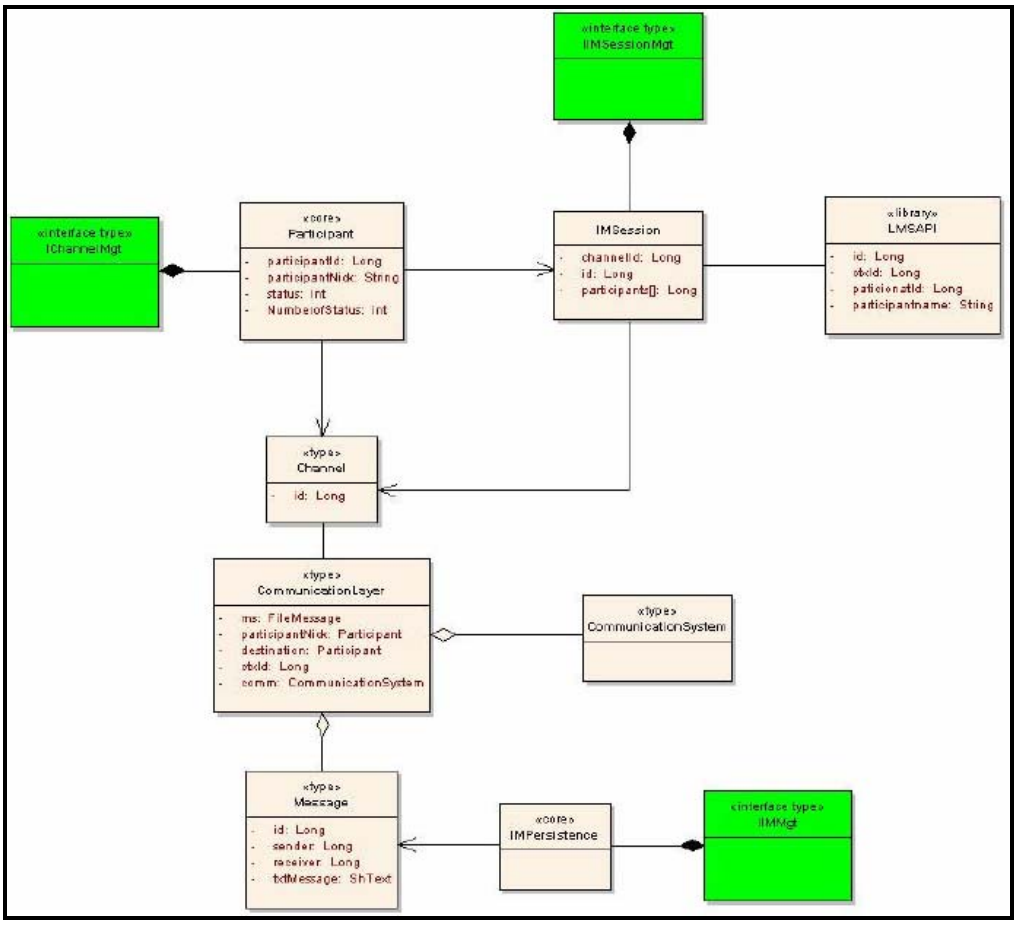

Figura 30: Modelo de Tipos de Negócio da Ferramenta *Instant Messenger* (PIRES et al., 2005).

A Figura 31 apresenta a arquitetura de componentes da ferramenta *instant messenger*, sendo que os principais componentes são: *Instant Messenger* e *SessionCommunicationMgr*.

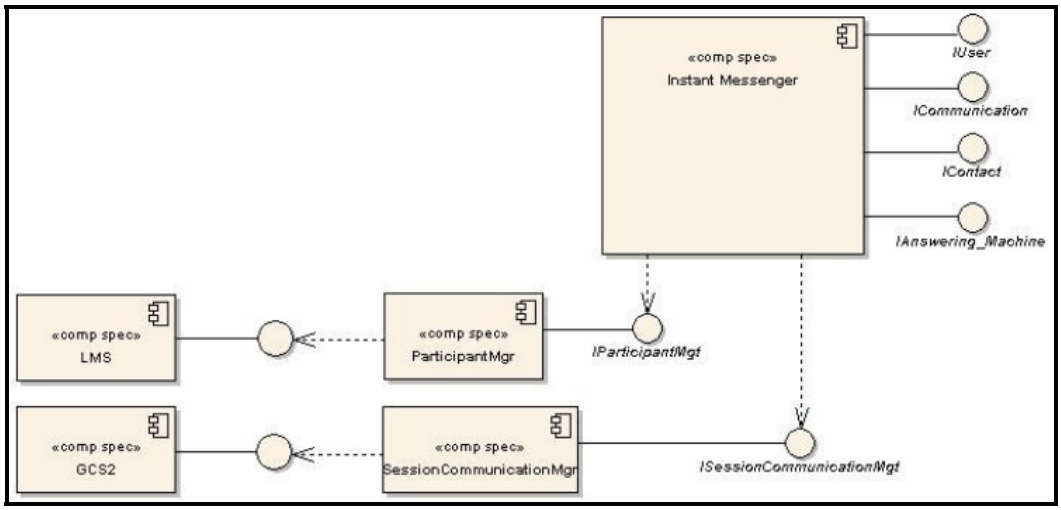

Figura 31: Aquitetura de Componentes da Ferramenta *Instant Messenger* (PIRES et al., 2005).

A Figura 32 apresenta o diagrama de casos de uso relativo à comunicação na ferramenta *instant messenger*, bem como as variabilidades encontradas neste artefato.

A principal variabilidade deste artefato está relacionada a forma de comunicação, sendo composta por:

- 01 ponto de variação: o caso de uso *Fazer Comunicação*;
- 08 variantes, sendo elas os casos de uso:
    - *Iniciar Comunicação de Texto*;
    - *Iniciar Comunicação Usando Shape*;
    - *Trocar Arquivos*;
    - *Iniciar Comunicação de Áudio*;
    - *Tornar Comunicação Pública*;
    - *Acessar Comunicação Gravada*;
    - *Iniciar Comunicação Usando Imagem*;
    - *Iniciar Comunicação de Vídeo.*

A relação entre ponto de variação e variantes foi definida como sendo alternativa inclusiva, sendo que esta é resolvida em tempo de execução e nenhuma nova variante pode ser adicionada e zero ou mais variantes do conjunto associado podem ser selecionadas.

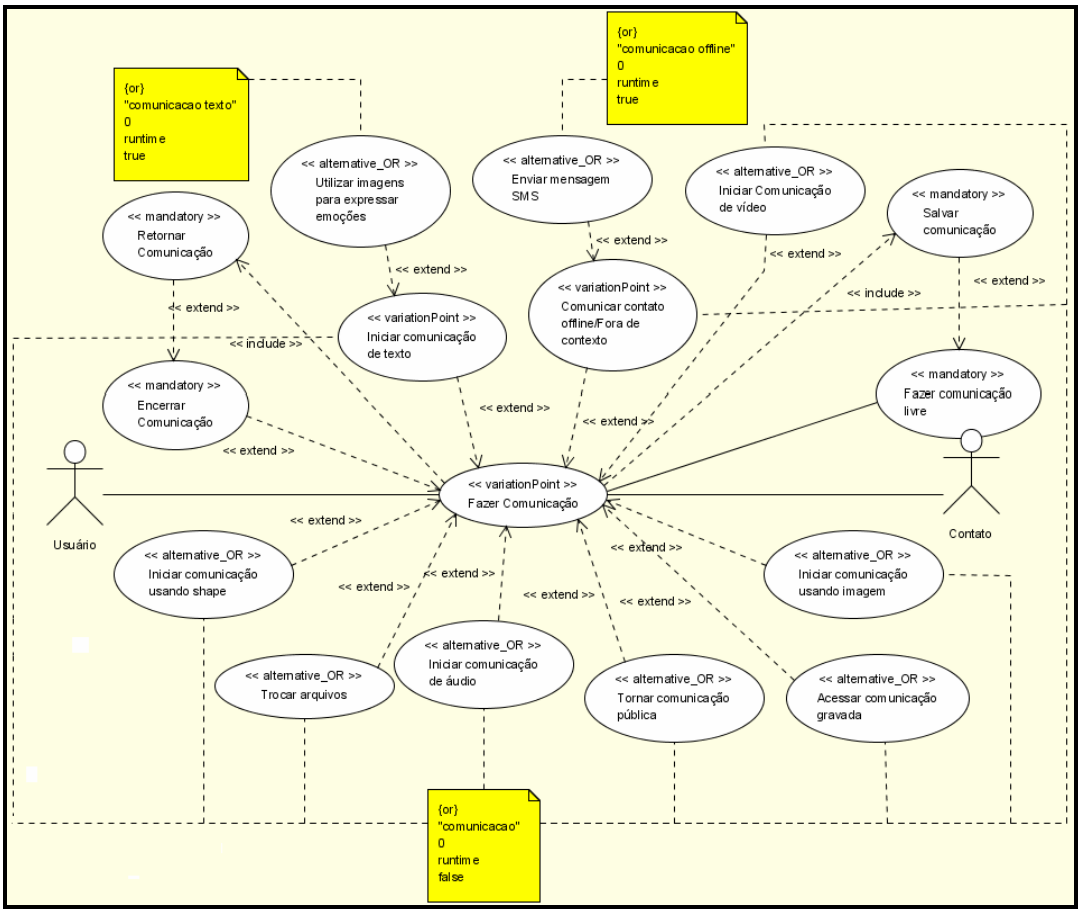

Figura 32: Variabilidades no Diagrama de Casos de Uso para Comunicação da Ferramenta *Instant Messenger*.

A Figura 33 apresenta o diagrama de casos de uso para o ator *Usuário*, bem como a variabilidade identificada com relação à administração de contatos. Essa variabilidade é composta por:

- 01 ponto de variação: o caso de uso *Adminstrar Contatos*;
- 06 variantes, sendo elas os casos de uso:
    - *Classificar Contatos*;
    - *Criar Grupo de Contatos*;
    - *Ver Perfil de um Contato*;
    - *Bloquear Contatos*;
    - *Gerenciar Contatos*;
    - *Exportar Lista de Contatos*.

A relação entre ponto de variação e variantes foi definida como sendo alternativa inclusiva, sendo que esta é resolvida em tempo de execução e nenhuma nova variante pode ser adicionada e pelo menos uma variante do conjunto associado deve ser selecionada.

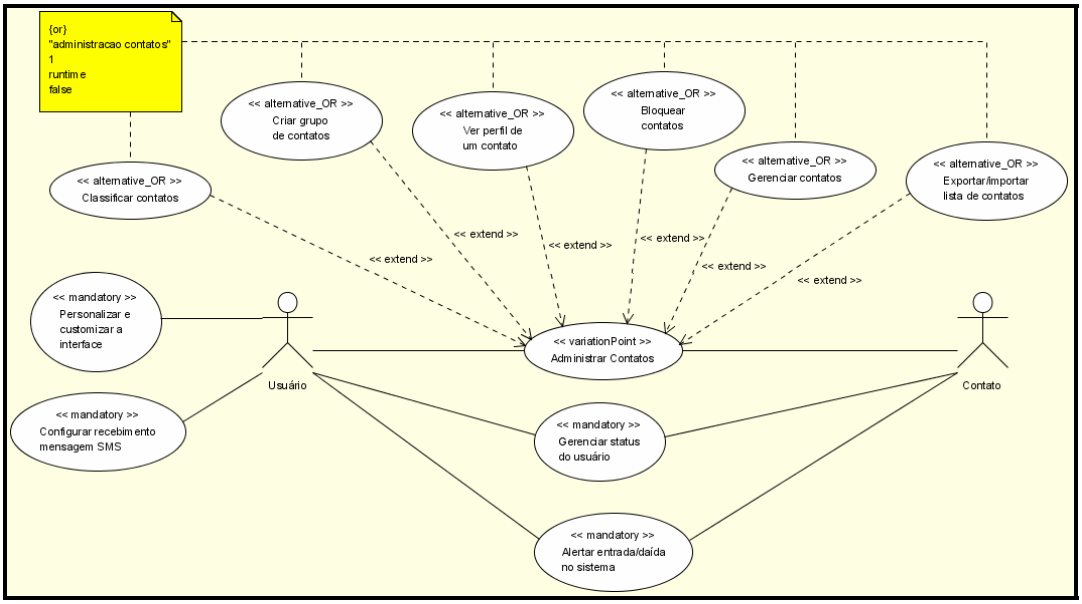

Figura 33: Variabilidades no Diagrama de Casos de Uso para o Ator Usuário da Ferramenta *Instant Messenger*.

A Figura 34 apresenta o modelo de tipos de negócio da ferramenta *instant messenger*, sendo que nenhuma variabilidade foi identificada seguindo o processo de gerenciamento de variabilidades. Todas as classes do modelo foram definidas de forma a obrigatoriamente pertencerem aos produtos gerados pela LP para o domínio em questão.

Deve-se considerar que a falta da variabilidade para este artefato possui uma relação direta na quantidade de possíveis produtos do domínio que podem ser gerados quando uma ferramenta *instant messenger* for selecionada para fazer parte de um ambiente de aprendizagem eletrônica. Assim, uma possível análise futura na modelagem deste artefato é necessária para saber se realmente a falta de variabilidades se deve a forma como as classes estão relacionadas ou se isto se deve a situações (relações entre classes) não previstas pelo processo de gerenciamento de variabilidades.

Uma forma de se chegar à uma conclusão mais precisa é mostrar o rastreamento das classes geradas a partir de cada caso de uso, o que não consta na documentação disponível da ferramenta. Com o rastreamento, seria possível identificar quais classes estão relacionadas com a realização de quais casos de uso, uma vez que existe variabildiade nos diagrama de casos de uso da ferramenta.

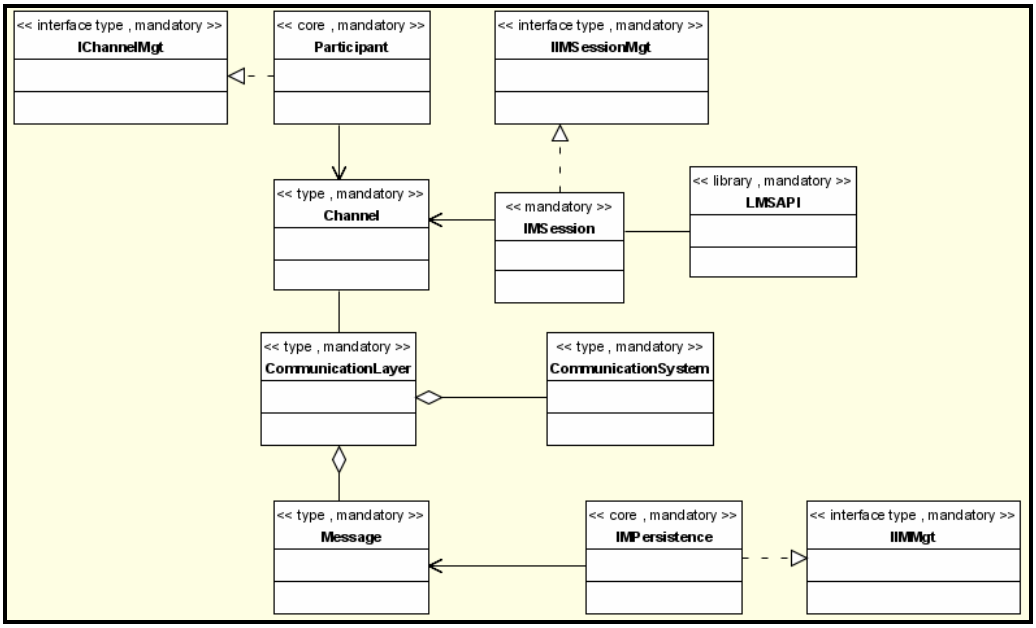

Figura 34: Modelo de Tipos de Negócio da Ferramenta *Instant Messenger* sem Variabilidades.

A Figura 35 apresenta a arquitetura de componentes da ferramenta *instant messenger* sem variabilidades, uma vez que as classes do modelo de tipos de negócio também não apresentaram varaiação.

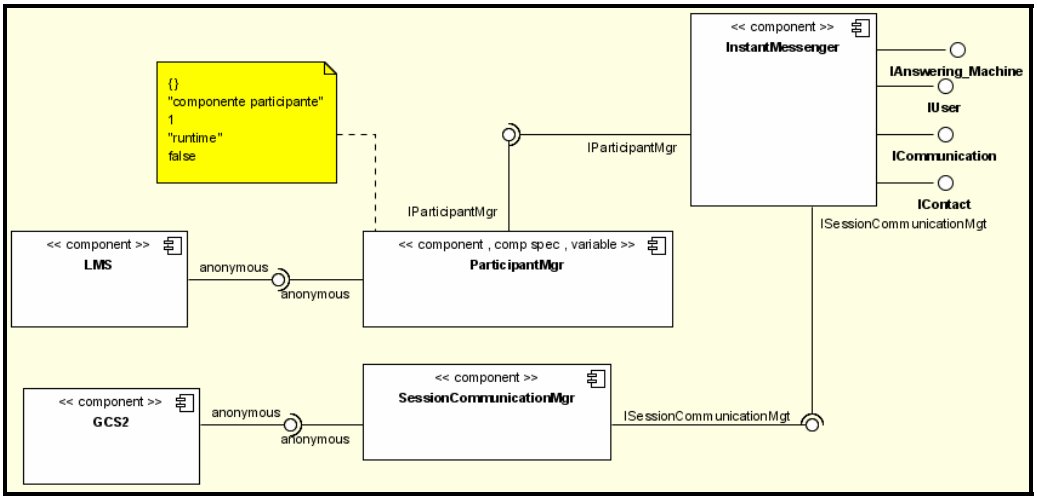

Figura 35: Variabilidades na Arquitetura de Componentes da Ferramenta *Instant Messenger*.

4.5 Análise dos Resultados e Dificuldades Encontradas

Aplicar as atividades do processo de gerenciamento de variabilidade requer essencialmente o conhecimento do domínio em questão, no caso o domínio de aprendizagem eletrônica. O conhecimento do domínio está relacionado diretamente à modelagem de uma LP e à acurácia na identificação, delimitação e representação das variabilidades.

Dessa forma, a grande dificuldade encontrada no desenvolvimento deste trabalho foi o conhecimento do domínio em questão. Tal dificuldade pôde ser amenizada com o estudo e conhecimento de forma geral da documentação do ambiente TIDIA-Ae já desenvolvido e em produção. Porém, alguns detalhes técnicos tanto de modelagem quanto de implementação acabam influenciando na modelagem de LP de forma reativa.

De uma forma geral, as ferramentas analisadas mostraram-se bastante propícias à abordagem de LP, salvo algumas particularidades. Os gráficos a seguir dão uma noção, de forma quantitativa, dos resultados obtidos com este estudo.

A Figura 36 apresenta um gráfico mostrando três valores para cada ferramenta, sendo eles: o número total de casos de uso da ferramenta, o número de casos de uso com variabilidade e a taxa de variabilidade percentual. Quando dizemos número de casos de uso com variabilidade, estamos nos referindo àqueles casos de uso com os estereótipos <<variationPoint>>, <<alternative_OR>, <alternative_XOR>> ou <<optional>>. Casos de uso com o estereótipo <<mandatory>> não entram na contagem, por serem obrigatórios para todos os produtos.

Assim, podemos notar que a ferramenta de *chat* possui a maior taxa de variabilidade, 100% dos seus casos de uso. Porém, a *whiteboard* posui somente 36% dos seus casos de uso variantes. De forma geral, as três ferramentas possuem uma taxa de variabillidade média de 69%, o que nos permite afirmar que é uma alta taxa segundo Jaring (2005).

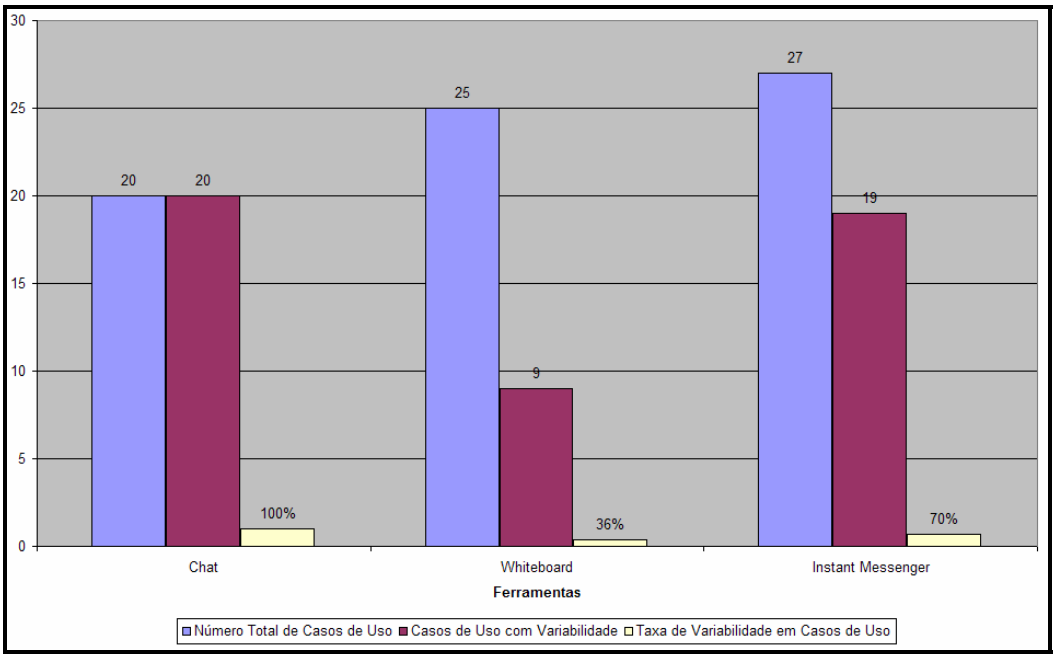

Figura 36: Taxa de Variabilidade em Casos de Uso das Ferramentas Analisadas.

O gráfico da Figura 37 apresenta a taxa de variabilidade para classes nas três ferramentas e ainda no LMS. A idéia da contagem de variabildiades é a mesma para casos de uso.

Podemos observar que a maior taxa de variabilidade é novamente a da ferramenta de *chat*, 92% das suas classes. A menor taxa de variabilidade é a da ferramenta *instant messenger*, já que em suas classes não foi encontrado nenhum tipo de variação. De forma geral, as três ferramentas e o LMS apresentaram uma taxa média de 46%. Podemos considerar uma taxa baixa segundo Jaring (2005). Isso ocorre pelo fato da ausência de variabilidade na ferramenta *instant messenger*, uma vez que os valores limites afetam a média aritmética. Se desconsiderarmos os valores limites, essa taxa sobe para 47%, o que a mantem baixa.

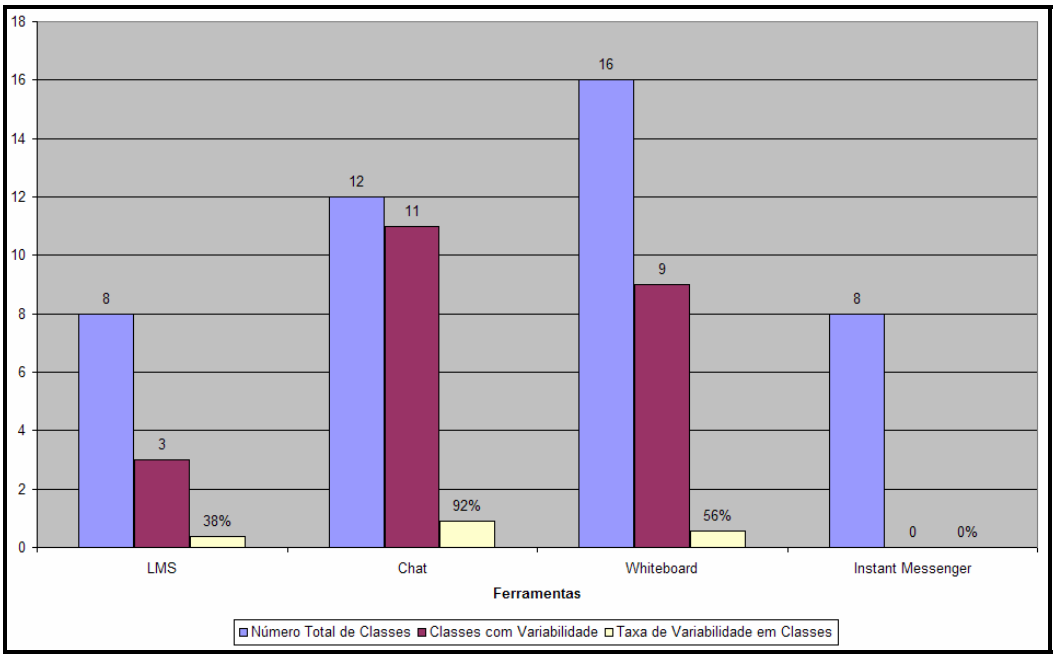

Figura 37: Taxa de Variabilidade em Classes das Ferramentas Analisadas.

De forma geral, o número de variabilidades encontrado em uma primeira análise da documentação das ferramentas nos mostra o potencial de reusabilidade e de possíveis produtos que podem ser gerados ao considerarmos o domínio de aprendizagem eletrônica. Neste sentido, dois pontos a serem considerados são fundamentais para o aumento do número de variabilidades do domínio em questão.

O primeiro deles é com relação à análise das demais ferramentas que compõem o ambiente TIDIA-Ae como um todo. Nesse trabalho somente três das seis ferramentas desenvolvidas pelo núcleo de São Carlos foram analisadas. Acredita-se que a análise das demais ferramentas tende a revelar um número maior de variabilidades em seus artefatos. Além disso, é necessário considerar as demais ferramentas desenvolvidas pelos outros núcelos do projeto TIDIA.

O segundo ponto a considerar é a possibilidade de se realizar uma análise na documentação das ferramentas estudadas de forma a corrigir a modelagem feita em alguns pontos como, por exemplo, o uso de relações de dependência (*extend* e *include*) entre casos de uso ao invés do uso de associações diretas entre estes. Isto contribui com o aumento de variabilidade sem, no entanto, mudar a semântica dos modelos. Isso é possível, uma vez que a UML é interpretativa e permite este tipo de relacionamento.

5. Considerações Finais

Este trabalho teve como objetivo principal estudar uma parte das ferramentas que compõem o ambiente TIDIA-Ae e aplicar o processo de gerenciamento de variabilidade proposto por Oliveira Junior (2005) no intuito de identificar, delimitar e representar graficamente as variabilidades nos artefatos de documentação destas ferramentas. Assim, foi possível fazer uma análise acerca das dificuldades encontradas na proposta de uma LP de forma reativa a partir de um produto já existente, bem como analisar se o domínio de aprendizagem eletrônica é propício à abordagem de LP e, com isso, realizar experimentos formais para apoiar a avaliação de LP's.

A grande dificuldade encontrada no desenvolvimento deste trabalho foi a falta de conheceimento do domínio em questão. Porém, a documentação disponível das ferramentas assim como o acompanhamento de parte da equipe do TIDIA amenizou tal dificuldade.

De modo geral, percebe-se que o domínio de aprendizagem eletrônica é propício à abordagem de LP por apresentar uma taxa bastante considerável de variabilidade em seus artefatos. Porém, uma análise mais profunda destes artefatos e das demais ferramentas que compõem o ambiente é necessária para que seja possível afirmar, com base em uma análise quantitativa, que tal domínio é recomendável à realização de experimentos formais de apoio à avaliação de LP.

Além disso, uma análise do processo de gerenciamento de variabilidade é recomendada, uma vez que foram observadas situações que não estavam previstas nos artefatos de documentação das ferramentas analisadas.

Finalmente, acredita-se que a realização deste trabalho contribuiu tanto para o conhecimento e possível adoção do domínio de aprendizagem eletrônica em futuros estudos complementares à proposta de doutorado quanto para uma análise mais profunda da documentação disponível do ambiente TIDIA-Ae e de suas ferramentas. Assim, acredita-se que a realização deste trabalho aconteceu de forma satisfatória com relação aos objetivos propostos.

Referências

Alves, V. et al. *Identifying Variations in Mobile Devices*. Journal of Object Technology, 4(3):47-52, Abril 2005.

Anastasopoulos, M. Implementing product line variabilities. ACM SIGSOFT Software Engineering Notes, New York, v. 26, n. 3, p. 109-117, May. 2001.

Bachman, F.; Bass, L. Managing variability in software architectures, In: SYMPOSIUM ON SOFTWARE REUSABILITY, 1., 2001, Toronto. Proceedings… New York: ACM, 2001. p. 126-132.

Bass, L.; Clements, P.; Kazman, R.. Software architecture in practice. 2. ed. Boston: Addison-Wesley, 2003. 560 p.

Becker, M. Towards a general model of variability in product families. In: SOFTWARE VARIABILITY MANAGEMENT WORKSHOP, 2003, Portland. Proceedings… Portland, 2003. p. 19-27.

Buhrdorf, R., Churchett, D. and Krueger, C. *Salion's Experience with a Reactive Software Product Line Approach*, in Proceedings of the 5th International Workshop on Product Family Engineering. Siena, Italy. Novembro 2003.

Bühne, S.; Halmans, G.; Pohl, K. Modelling Dependencies between Variation Points in Use Case Diagrams. In: 9th INTERNATIONAL WORKSHOP ON REQUIREMENTS ENGINEERING, 9., 2003, Austria. Proceedings... Austria, 2003. p. 59–69.

Cheesman, J.; Daniels, J. *UML Components – A Simple Process for Specifying Component-Based Software*. Addison-Wesley, 2001.

Clauß, M. Generic modeling using UML extensions for variability. In: OOPSLA 2001 WORKSHOP ON DOMAIN SPECIFIC VISUAL LANGUAGES, 1. 2001, Tampa Bay. Proceedings… Tampa Bay, 2001a. p. 11-18.

Clements, P.; Northrop, L. *Software Product Line: Practices and Patterns*. Addison Wesley Longman, 2001.

Czarnecki, K., Eisenecker, U. *Generative Programming: Methods, Tools, and Applications*. Addison-Wesley, 2000. 832 p.

Fritsch, C.; Lehn, A.; Strohm, T. Evaluating variability implementation mechanisms. In: INTERNATIONAL WORKSHOP ON PRODUCT LINE ENGINEERING, 2., 2002, Seattle. *Proceedings*… Seattle, 2002. p. 59-64.

Gimenes, I. M.; Travassos, G. H.. *O enfoque de Linha de Produto para Desenvolvimento de Software*. In: Sociedade Brasileira de Computação; Ingrid Jansch Porto. (Org.). XXI JAI - Livro Texto. Florianópolis, 2002, v. 2, p. 1-32.

Gomaa, H. Designing software product lines with UML: from use cases to pattern-based software architectures. Boston: Addison-Wesley, 2005.

Gomaa, H.; Webber, D. Modeling adaptive and evolvable software product lines using the variation point model. In: HAWAII INTERNATIONAL CONFERENCE ON SYSTEM SCIENCES, 37., 2004, Hawaii. Proceedings… Hawaii, 2004. p. 01-10.

Griss, M. L.; Favaro, J.; D´alessandro, M. Integrating feature modeling with the RSEB. In: INTERNATIONAL CONFERENCE ON SOFTWARE REUSE, 5., 1998, Washington. Proceedings… Washington, 1998. p. 76-85.

Halmans, G.; Pohl, K. Communicating the variability of a software-product family to customers. Journal on Software and Systems Modeling, New York, v. 2, n. 1, p. 15-36, mar. 2003.

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

Jacobson, I.; Griss, M.; Jonsson, P. Software reuse - architecture process and organization for business success. 1. ed. Boston: Addison-Wesley, 1997. 528 p.

Jaring, M. *Variability Engineering as an Integral Part of the Software Product Family Development Process*. 2005. 200 f. Tese (Doutorado) – Universidade de Groningen, Maandag, 2005.

Kudo, T. N.; Monteiro, J. D.; Eleutério, P. M. S.; Endres, A. F.; Silva, E. Q. TIDIA-Ae - Especificação da Ferramenta Chat. 22 f. Universidade de São Paulo, São Carlos, 2006.

Kudo, T. N.; Jardim, C. H. O. TIDIA-Ae - Especificação da Ferramenta Whiteboard. 20 f. Universidade de São Paulo, São Carlos, 2006.

Matinlassi, M. *Comparison of Software Product Line Architecture Design Methods: COPA, FAST, FORM, KobrA and QADA*. In: Proceedings of 26th International Conference on Software Engineering (ICSE'04), 2004.

Meantime Mobile Games Disponível em < http://www.meantime.com.br>. Acessado em 10 de nov. 2006.

Muccini, H.; van der Hoek, A. *Towards Testing Product Line Architectures*. Electronic Notes in Theoretical Computer Science 82 No. 6, 2003. http://www.elsevier.nl/locate/entcs/volume82.html.

Northrop, L. M. SEI's Software product line tenets. IEEE Software, v. 19, n. 14, p. 32-41, 2002.

Oliveira Junior, E. A. *Um Processo de Gerenciamento de Variabilidade para Linha de Produto de Software*. 2005. 155 f. Dissertação (Mestrado em Ciência da Computação) – Departamento de Informática, Universidade Estadual de Maringá, Maringá, 2005.

Oliveira Junior, E. A.; Gimenes, I. M. S.; Huzita, E. H. M.; Maldonado, J. C. A *Variability Management Process for Software Product Lines*. In: CASCON 2005, 2005, Toronto. Proceeding of CASCON 2005. Ottawa, Canada: National Research Council Canada, 2005. v. 1, p. 30-44.

Ommering, R. C. Building Product Populations with Software Components, http://irs.ub.rug.nl/ppn/275169561, University of Groningen 2004.

Pires, D. F.; Kudo, T. N. Kudo; Teixeira, C. A.; Gasparin, R.; Monteiro, L. N.; Bataglião, L.; Andrade, P. TIDIA-AE - Especificação da Ferramenta Comunicador Instantâneo. 20 f. Universidade de São Paulo, São Carlos, 2005.

SEI - Software Engineering Institute. A framework for software product line practice 4.2. Pittsburgh. Disponível em <http://www.sei.cmu.edu/plp/framework.html>. Acesso em: 10 de nov. 2006.

Silva, C. G., Modelo Conceitual do Learning Management System (LMS). 2005. 5 f. Relatório de Projeto – Grupo de Trabalho Interfaces Humano-Computador, e-Labora, Universidade Estadual de Campinas, Campinas, 2005.

SPLC – International Software Product Line Conference – Disponível em < http://www.sei.cmu.edu/splc2006/>. Acessado em 10 de nov. 2006.

Svahnberg, M.; van Gurp, J.; Bosch, J. A taxonomy of variability realization techniques. Technical report, Blekinge Institute of Technology, Sweden, 2002.

Svahnberg, M.; Bosch, J. Issues concerning variability in software product lines. In: INTERNATIONAL WORKSHOP ON SOFTWARE ARCHITECTURES FOR PRODUCT FAMILIES, 3., 2000, Las Palmas. Proceedings… Las Palmas, 2000. p. 146-157.

van der Linden, F., Bosch, J., Kamsties, E., Känsälä, K., Obbink, H. *Software Product Family Evaluation*. SPLC 2004, LNCS 3154, pp. 110-129, 2004. Spinger-Velarg Berlim Heidelberg, 2004.

van Gurp, J.; Bosch, J. On the notion of variability in software product lines. In: THE WORKING IEEE/IFIP CONFERENCE ON SOFTWARE ARCHITECTURE, 2001, Amsterdam. *Proceedings*… Amsterdam, 2001.

van Gurp, J.; Bosch, J. Managing variability in software product lines. In: THE WORKING IEEE/IFIP CONFERENCE ON SOFTWARE ARCHITECTURE, 2000, Amsterdam. Proceedings… Amsterdam, 2000.

van Gurp, J. Variability in software systems – the key to software reuse. 2000. 194 f. Trabalho de Conclusão de Curso (Graduação) - Department of Software Engineering and Computer Science, University of Groningen, Groningen, 2000.

ANEXO I - RELAÇÕES E ESTEREÓTIPOS USADOS NA REPRESENTAÇÃO GRÁFICA DE PONTOS DE VARIAÇÃO E VARIANTES

	Diagramas de Casos de Uso		**Diagramas de Classes**		**Diagramas de Componentes**	
	Relação da UML	**Estereótipo do Elemento**	**Relação da UML**	**Estereótipo do Elemento**	**Relação da UML**	**Estereótipo do Elemento**
Ponto de Variação	--------	<<variationPoint>>	--------	<<variationPoint>>	Dependência	<<variable>>
Variante Obrigatória	Associação Agregação Composição Dependência	<<mandatory>>	Associação Agregação Composição Dependência	<<mandatory>>	Dependência	<<mandatory>>
Variante Opcional	Associação Agregação Composição Dependência	<<optional>>	Associação Agregação Composição Dependência	<<optional>>	Dependência	<<optional>>
Variante Alternativa Inclusiva	Esp. / General. com o estereótipo <<extend>>	<<alternative_OR>>	Herança	<<alternative_OR>>	Dependência	<<alternative_OR>>
Variante Alternativa Exclusiva	Esp. / General. com o estereótipo <<extend>>	<<alternative_XOR>>	Herança	<<alternative_XOR>>	Dependência	<<alternative_XOR>>
Variante Mutuamente Exclusiva	Dependência	<<mutex>>	Dependência	<<mutex>>	Dependência	<<mutex>>
Variante Inclusiva	Dependência	<<requires>>	Dependência	<<requires>>	Dependência	<<requires>>