Suporte ao Uso de Frameworks Orientados a Objetos com Base no Histórico do Desenvolvimento de Aplicações

EVANDRO CÉSAR FREIBERGER[1]
RICARDO PEREIRA E SILVA[2]

[1]CEFET – Centro Federal de Educação Tecnológica de Mato Grosso
Rua Professora Zulmira Canavarros 95 - 78005-390 Cuiabá - MT
UNIC – Universidade de Cuiabá
Av. Beira Rio, 3100. Jardim Europa. CEP: 78015-480
evandrofreiberger@unic.br

[2]UFSC – Universidade Federal de Santa Catarina
INE – Departamento de Informática e de Estatística
CP 476, 88037-000, Florianópolis, SC, Brasil
http://www.inf.ufsc.br/~ricardo
ricardo@inf.ufsc.br

Resumo

Este trabalho apresenta um conjunto de métricas a serem extraídas de aplicações baseadas em frameworks orientados a objetos. As métricas podem ser extraídas automaticamente de especificações em UML de aplicações e frameworks através de uma ferramenta desenvolvida. Elas ajudam desenvolvedores de aplicação em sua avaliação, assim como produzem uma realimentação, útil à avaliação de frameworks. Uma segunda ferramenta foi desenvolvida para suportar a navegação no conjunto de métricas, devido à significativa quantidade de informações obtidas em uma análise automática.

Palavras Chaves: Frameworks orientados a objetos, suporte ao uso de frameworks, desenvolvimento orientado a objetos, reuso.

Abstract

This work presents a set of metrics to be obtained from object-oriented framework based applications. The metrics can be extracted automatically from framework and application UML design specifications by a developed tool and are pointed to help application developers as well as produce a feedback useful for framework evaluation. A second tool was developed to support metric browsing because the significant amount of information obtained in an automatic analysis.

Keywords: Object-Oriented framework, framework use support, object-oriented development, software reuse.

1 Introdução

O desenvolvimento de software através do modelo orientado a objetos é impulsionado por características tais como: robustez, flexibilidade à mudança de requisitos e arquitetura, perspectiva de reutilização de código e projeto. Contudo, o modelo orientado a objetos por si só não garante todas essas características: é necessário o uso de técnicas e diretrizes que

conduzam o desenvolvedor para atingir tais resultados, principalmente quando o tamanho e a complexidade das aplicações aumentam [7,10].

Os frameworks orientados a objetos representam uma categoria de software potencialmente capaz de promover reuso de alta granularidade [6]. Trata-se de uma estrutura de classes interligadas, correspondendo a uma aplicação inacabada, para um conjunto de aplicações de um determinado domínio. A extensão dessa estrutura produzirá aplicações, proporcionando reuso.

O desenvolvimento de software a partir de frameworks orientados a objetos apresenta duas dificuldades inerentes. A primeira está associada à complexidade do desenvolvimento de frameworks, pois o projetista necessita construir uma estrutura de classes capaz de generalizar os requisitos de um domínio e, ao mesmo tempo, permitir a adaptação às especificidades de cada aplicação. A segunda dificuldade está associada ao uso dos frameworks, pois na maioria das vezes, o usuário de um framework (desenvolvedor de aplicações) precisa entender a estrutura interna do framework antes de usá-lo no desenvolvimento de aplicações [1]. Tanto o desenvolvedor do framework, quanto o desenvolvedor de aplicações, necessitam de informações a respeito de seus artefatos de software. O desenvolvedor do framework precisa de indicadores da conformidade de seu artefato com o domínio de aplicação pretendido, assim como, o desenvolvedor de aplicações precisa de indicadores da conformidade da sua aplicação desenvolvida sob um framework, em relação ao que foi projetado e pretendido no framework usado. Observa-se na literatura recomendações relacionadas ao desenvolvimento e uso de frameworks, porém também uma carência de subsídios para a avaliação dos resultados dessas atividades.

Nesse sentido, este trabalho propõe um conjunto de medidas a serem extraídas a partir de especificações de frameworks orientados a objetos e de especificações de aplicações desenvolvidas sob esses frameworks. Esses dados possibilitam a comparação de uma dada aplicação com outras aplicações produzidas sob o mesmo framework, subsidiando o desenvolvedor da aplicação para a avaliação de seu trabalho de desenvolvimento, através da comparação com outros desenvolvimentos existentes. Por outro lado, os dados referentes às especificações de aplicações sob um framework poderão ser utilizados pelo desenvolvedor do framework, como subsídio à análise da aderência do framework ao domínio tratado.

2 Aspectos de Qualidade de Frameworks Orientados a Objetos

O desenvolvedor de um framework está preocupado em produzir um framework que suporte o desenvolvimento de um conjunto amplo de aplicações de um determinado domínio. Preocupa-se também com a possibilidade de aplicações serem desenvolvidas com o menor esforço possível, no que se refere à extensão da estrutura do framework e à dificuldade de entendimento de como fazê-lo [1].

Um framework ideal dará suporte ao desenvolvimento de aplicações sem que o desenvolvedor da aplicação tenha necessidade de inserir novos conceitos do domínio[1], especificando sua aplicação através da especialização das classes do framework e sobrescrita de seus métodos. Porém, um framework ideal é um objetivo utópico, cuja busca pode ser bastante onerosa. Para frameworks reais existirá sempre a possibilidade de novos artefatos (classes, atributos, métodos) serem desenvolvidos na aplicação, para atender a requisitos

[1] A não inserção de novos conceitos do domínio significa que as classes criadas no desenvolvimento de uma aplicação sob um framework têm relação de herança com classes do framework, ou seja, são especializações de classes pertencentes ao framework, que modelam conceitos do domínio tratado. Classes criadas que não tenham relação de herança com as classes do framework, por outro lado, correspondem à modelagem de conceitos do domínio não previstos no framework.

específicos, não previstos no framework. Uma questão prática é a necessidade de avaliar em termos objetivos se uma aplicação foi desenvolvida utilizando adequadamente um framework, assim como se um framework é adequado a um domínio tratado.

Para tratar essa demanda de avaliação objetiva considerou-se os princípios conceituais da abordagem de frameworks: a modelagem de domínio e o "princípio de Hollywood". Com relação à modelagem de domínio, um framework deve conter todos os conceitos e características do domínio tratado, restando ao usuário especializá-las – e não, defini-las. Concretamente, isso pode ser avaliado verificando se há classes na aplicação sem relação de herança com as classes do framework, como já mencionado, e se as classes da aplicação possuem atributos que correspondam a características dos conceitos modelados. No caso da necessidade de criação desses atributos demonstraria que nem todas as características de conceitos existentes no framework foram consideradas quando do seu desenvolvimento.

O "princípio de Hollywood" (*frameworks chamam, não são chamados*) estabelece que o fluxo de controle de aplicações desenvolvidas a partir de um framework já está definido no framework. Uma forma objetiva de avaliar a obediência a este princípio é verificar se as classes criadas em uma aplicação possuem métodos que não sobrescrevam métodos herdados, previamente definidos no framework. Tais métodos, em princípio, não deveriam existir, pois, o "princípio de Hollywood" é baseado em herança e polimorfismo: para uma instância de uma classe invocar um método de uma classe criada posteriormente, é necessário que esta classe sobrescreva um método previamente definido.

Dados objetivos para a avaliação do uso de um framework seriam as contagens de classes da aplicação relacionadas e não-relacionadas por herança às classes do framework, de atributos criados nas classes da aplicação e de métodos que não sobrescrevam métodos herdados. Essas informações servem de subsídio para uma avaliação. Há, porém, outros aspectos a considerar, além dos valores numéricos. No caso dos atributos, há problema se sua finalidade for descrever uma característica de um conceito do domínio tratado, mas não, se for apenas preservar um dado entre a execução de diferentes métodos de uma classe. No caso dos métodos, não devem ser considerados inadequados os métodos que não sobrescrevam métodos herdados mas que foram criados para quebrar algoritmos longos, implementados em métodos que sobrescrevam métodos herdados.

Um outro foco para avaliar a criação de subclasses e a sobrescrita de métodos herdados consiste em avaliar sua adequação ao que foi previsto no projeto do framewok. Isto depende das classes do framework terem sido classificadas como redefiníveis, isto é, aquelas que o desenvolvedor do framework admite a criação de subclasses e as não-redefiníveis, isto é, as que não admitem especialização. De forma similar, a conveniência ou não da sobrescrita dos métodos depende de sua classificação explícita. Métodos *template*, isto é, métodos concretos cujo algoritmo consiste em invocar outros métodos (métodos *hook*[2]) correspondem a algoritmos genéricos que não admitem sobrescrita. Alterações de seu algoritmo devem ser tratadas através da sobrescrita dos métodos *hook* por ele invocados.

A verificação da presença nas aplicações sob um framework das características indesejáveis acima descritas serve de base para a avaliação das aplicações desenvolvidas, assim como para avaliar o próprio framework. Uma aplicação que apresente as tais características, à princípio, mostra que o framework não foi usado de forma adequada. Por outro lado, um framework que leve seus usuários a gerar classes com essas características não modela adequadamente o domínio tratado. Portanto, diante da constatação de classes com

[2] Considera-se a classificação de métodos como abstratos, *template* e regulares (estes, métodos concretos não-classificáveis como *template*). A classificação método *hook* é relativa: um método é classificado como *hook* porque um *template* o invoca. Todo método *hook* é, portanto, classificável como abstrato, *template* ou regular (esta, uma classificação absoluta)

características inadequadas, segundo os critérios descritos, há a necessidade de verificar se o problema está no mau uso do framework ou em sua estrutura.

Uma forma de avaliar a qualidade da construção e do uso um framework é considerar o histórico das aplicações desenvolvidas sob esse framework. Assim, o foco principal não é a qualidade do framework em si, e sim, a qualidade do uso desse framework no processo de desenvolvimento de aplicações. Essa abordagem possibilita também avaliar quanto uma determinada aplicação se manteve "na média" em relação a outras aplicações desenvolvidas sob um mesmo framework, no que se refere à fidelidade aos princípios de uso.

As medidas propostas no presente trabalho fornecem subsídios para avaliar tanto a aderência de um framework a um domínio tratado, como do uso adequado de um framework no desenvolvimento de uma aplicação, tomando por base um conjunto de aplicações geradas sob um mesmo framework. Também permitem quantificar o esforço de desenvolvimento de cada aplicação, através do dimensionamento dos produtos desenvolvidos.

3 Medidas obtidas do histórico de uso de frameworks orientados a objetos

O primeiro grupo de medidas busca quantificar a proporção de desenvolvimento de artefatos de software (classes, atributos, métodos e comandos dos algoritmos dos métodos) em relação ao que foi reusado nas aplicações desenvolvidas sob o framework. O segundo grupo de medidas busca quantificar a proporção de especialização de artefatos de software (classes e métodos) em relação ao definido ("novo") nas aplicações desenvolvidas sob o framework. O último grupo visa quantificar a proporção dos artefatos (classes e métodos) sobrescritos pelas aplicações, destacando as sobrescrições previstas no framework (classes redefiníveis, classes essenciais, classes abstratas, métodos abstratos) em relação às sobrescrições não previstas.

3.1 Quantificação de proporção de software produzido e reusado

A tabela 1 apresenta o conjunto de medidas proposto para quantificação do reuso. Essas medidas visam estabelecer proporções de reuso de artefatos de software do framework, em relação aos artefatos desenvolvidos em cada aplicação. São considerados os artefatos classe, método, atributo e comando.

Tabela 1 - Avaliações de reuso de classes do framework pelas aplicações

Sigla	Descrição	Comentários
NCEA	Número de classes exclusivas da aplicação	Classes definidas na aplicação, com ou sem relação de herança com classes do framework
NCFRD	Número de classes do framework referenciadas de forma direta	Número de classes do framework que foram referenciadas em classes da aplicação.
NCFRI	Número de classes do framework referenciadas de forma indireta	Dado o conjunto de classes do framework referenciadas diretamente (medida anterior) são avaliadas as referências que estas fazem a outras classes do framework.
TCA	Total de classes da aplicação	Representa o somatório das classes exclusivas da aplicação, das classes do framework referenciadas de forma direta e das classes do framework referenciadas de forma indireta.
NCFNR	Número de classes do framework não referenciadas	Representa o número de classes do framework que a aplicação não usou nem de forma direta e nem de forma indireta.
PRC	Percentual de reuso de classes	Corresponde à proporção entre o total de classes de um framework referenciadas (NCFRD + NCFRI) e o total de classes de uma aplicação (TCA).
PDC	Percentual de desenvolvimento de classes	Corresponde à proporção entre o total de classes exclusivas (NCEA) e o total de classes de uma aplicação (TCA).

Analogamente ao conjunto de medidas acima descrito, foram desenvolvidos outros três conjuntos que medem o grau de reuso de métodos, atributos e comandos de métodos.

3.2 Quantificação de especialização no uso de frameworks

A tabela 2 apresenta o conjunto de medidas proposto para quantificação da especialização de conceitos do framework no desenvolvimento de aplicações. As medidas tratam artefatos do framework que foram estendidos ou sobrescritos pelas aplicações. São considerados os artefatos classe e método.

Tabela2 - Avaliações quanto à especialização de classes do framework

Sigla	Descrição	Comentários
NCE	Número de classes especializadas	Dentre as classes definidas na aplicação (NCEA), são consideradas as classes definidas como subclasses de classes do framework.
NCN	Número de classes "novas"	Dentre as classes definidas na aplicação (NCEA), são consideradas como classes novas, aquelas que não possuem relação de herança com classes externas à aplicação.
PEC	Percentual de especialização de classes	Corresponde à proporção entre o total de classes especializadas (NCE) e o total de classes de uma aplicação (NCEA).
PNC	Percentual de novas classes	Corresponde à proporção entre o total de classes "novas" (NCN) e o total de classes de uma aplicação (NCEA).

Analogamente ao conjunto de medidas acima descrito, há um outro conjunto que quantifica a proporção dos métodos que sobrescrevem ou não métodos herdados.

3.3 Quantificação da adequação de redefinições no uso de frameworks

O conjunto de medidas apresentado na tabela 3 visa quantificar as classes redefiníveis e não-redefiníveis do framework, bem como sua redefinição nas aplicações sob o framework.

Tabela 3 - Avaliações quanto à "redefinibilidade" de classes do framework

Sigla	Descrição	Comentários
NCRF	Número de classes redefiníveis do framework	São consideradas somente as classes classificadas explicitamente como redefiníveis no framework.
NCEAP	Número de classes especializadas, do conjunto previsto de redefinição	Essa medida permite a avaliação do número de classes do framework que foram classificadas como redefiníveis e que, de fato, foram redefinidas (especializadas) nas aplicações
NCENP	Número de classes especializadas, sem previsão de redefinição no framework	Essa medida permite a avaliação do número de classes da aplicação que especializaram classes do framework não classificadas como redefiníveis
TCE	Total de classes especializadas	É o somatório das classes redefinidas na aplicação com ou sem previsão de redefinição no framework
PRP	Percentual de especialização prevista	Corresponde à proporção entre o total de classes especializadas adequadamente (NCEAP) e o total de classes especializadas (TCE).
PRNP	Percentual de especialização não-prevista	Corresponde à proporção entre o total de classes especializadas inadequadamente (NCENP) e o total de classes especializadas (TCE).

O conjunto de medidas apresentado na tabela 4 visa quantificar a sobrescrita de métodos herdados, considerando sua classificação. As medidas propostas nesta seção possibilitam a comparação estatística de características de uma dada aplicação com outras

aplicações especificadas sob um mesmo framework. Esses dados subsidiam o desenvolvedor da aplicação na avaliação do seu trabalho de desenvolvimento, comparando-o com outros desenvolvimentos existentes. Outro aspecto importante é que, os dados que representam o histórico de desenvolvimento de aplicações, podem ser utilizados pelo desenvolvedor do framework como subsídio à análise da aderência do mesmo ao domínio tratado.

Tabela4 - Avaliações quanto à sobrescrita de métodos do framework

Sigla	Descrição	Comentários
NMTF	Número de métodos *template* do framework	São considerados somente os métodos definidos explicitamente como *template* no framework.
NMAF	Número de métodos abstratos do framework	São considerados somente os métodos definidos explicitamente como abstratos no framework.
NMRF	Número de métodos regulares do framework	São considerados somente os métodos definidos explicitamente como regulares no framework.
NMTS	Número de métodos *template* sobrescritos	Dentre os métodos definidos na aplicação, são contados aqueles que sobrescrevem métodos *template* do framework.
NMAS	Número de métodos abstratos sobrescritos	Dentre os métodos definidos na aplicação, são contados aqueles que sobrescrevem métodos abstratos do framework.
NMRS	Número de métodos regulares sobrescritos	Dentre os métodos definidos na aplicação, são considerados aqueles que sobrescrevem métodos regulares do framework.
TMS	Total de métodos sobrescritos	É o somatório do número de métodos *template* sobrescritos, dos métodos abstratos sobrescritos e dos métodos regulares sobrescritos.
PMTS	Percentual de métodos *template* sobrescritos	Corresponde ao percentual de métodos *template* sobrescritos, em relação ao total de métodos sobrescritos na aplicação (TMS).
PMAS	Percentual de métodos abstratos sobrescritos	Corresponde ao percentual de métodos abstratos sobrescritos, em relação ao total de métodos sobrescritos na aplicação (TMS).
PMRS	Percentual de métodos regulares sobrescritos	Representa o percentual de métodos regulares sobrescritos, em relação ao total de métodos sobrescritos na aplicação (TMS).
NMTHS	Número de métodos *template hook* sobrescritos	Dentre os métodos definidos na aplicação, são considerados aqueles que sobrescrevem métodos *hook* do framework e que têm classificação explícita como *template.*
NMAHS	Número de métodos abstratos *hook* sobrescritos	Dentre os métodos definidos na aplicação, são considerados aqueles que sobrescrevem métodos *hook* do framework e que têm classificação explícita como abstratos.
NMRHS	Número de métodos regulares *hook* sobrescritos	Dentre os métodos definidos na aplicação, são considerados aqueles que sobrescrevem métodos *hook* do framework e que têm classificação explícita como regulares
PMTHS	Percentual de métodos *template hook* sobrescritos	Corresponde ao percentual de métodos *template*, que também são *hook*, sobrescritos em relação ao total de métodos sobrescritos da aplicação
PMAHS	Percentual de métodos abstratos *hook* sobrescritos	Corresponde ao percentual de métodos abstratos, que também são *hook*, sobrescritos em relação ao total de métodos sobrescritos da aplicação
PMRHS	Percentual de métodos regulares *hook* sobrescritos	Corresponde ao percentual de métodos regulares, que também são *hook*, sobrescritos em relação ao total de métodos sobrescritos da aplicação

4 Processo de extração, armazenamento e manipulação de dados

Para automatizar a extração de dados de especificações de aplicação e de frameworks foi desenvolvido um protótipo de ferramenta capaz de obter as medidas propostas no capítulo anterior. O protótipo foi construído como uma extensão do ambiente SEA [8]. A tabela 5 apresenta o conjunto de dados extraído de especificações de frameworks e de aplicações.

SEA suporta o desenvolvimento e o uso de frameworks. O desenvolvimento de um framework ou de uma aplicação nesse ambiente consiste em construir uma especificação de projeto, utilizando UML com extensões[3], e, de forma semi-automatizada, verificar sua consistência e convertê-la em código.

O protótipo da ferramenta de extração de dados manipula especificações de frameworks e aplicações produzidas no ambiente SEA e permite o armazenamento de dados de aplicações especificadas sob um framework, criando assim uma base de dados correspondente ao histórico do uso desse framework.

Tabela 5 – Atributos extraídos das especificações analisadas

Artefato de software	Atributos extraídos do artefato
Classes	• Identificação; • Lista de atributos e seus respectivos tipos[4]; • Classificação (abstrata ou concreta); • Origem (da própria especificação ou externa); • Superclasse (caso exista); • Lista de métodos; • Redefinibilidade, no caso de frameworks;
Atributos	• Identificação; • Tipo; • Origem (da própria especificação ou externa);
Métodos	• Identificação; • Lista de parâmetros e seus respectivos tipos; • Tipo de retorno; • Classificação (abstrato, template ou base); • Natureza do método (de instância ou de classe); • Origem (da própria especificação ou externa); • Classe (identificação da classe onde foi definido); • Override (sobrescrição de algum método); • Número de comandos[5] especificados no DCM[6] (descreve o algoritmo do método em questão); • Lista de variáveis temporárias do DCM e seus respectivos tipos; • Lista de mensagens enviadas a partir do DCM do método em questão; • Lista de atributos e seus respectivos tipos, referenciados pelo DCM do método em questão;
Parâmetros de métodos	• Identificação; • Tipo; • Origem (da própria especificação ou externa);
Variáveis temporárias de métodos (obtidas em comandos de declaração de variáveis contidos em DCMs)	• Identificação; • Tipo; • Origem (da própria especificação ou externa);
Mensagens (comandos de envio de mensagem contidos em DCMs)	• Identificação; • Objeto destino (qual o objeto que implementa o método invocado); • Método (método invocado do objeto destino);

[3] SEA possui um modelo para a descrição de algoritmos de métodos, o que possibilita que em alguns casos a geração automática de código gere 100% de uma aplicação ou de um framework. Além disso, SEA permite a classificação das classes de um framework como redefinível ou não-redefinível, bem como a classificação de métodos citada.

[4] Neste trabalho, a expressão "tipos de dado" representa a instância de uma classe ou um tipo, como String, de linguagens híbridas.

[5] No ambiente SEA os comandos são os *Statements* dos diagramas de corpo de métodos.

[6] DCM – Diagrama de Corpo de Método: modelo usado no ambiente SEA para especificação de algoritmos de métodos através de comandos, como *if*, *return* etc.

4.1 Mecanismo de extração de dados de especificações

Tem a finalidade de extrair dados da especificação de uma aplicação desenvolvida a partir de um ou mais frameworks no ambiente SEA. Nesse processo são extraídos dados referentes à especificação da aplicação em edição e da especificação do framework (podendo ser mais de um framework). Abaixo são descritos os dados extraídos dessas especificações.

4.2 Mecanismo de armazenamento de dados de especificações

Após extrair os dados da especificação de uma aplicação e dos frameworks a ela associados, a ferramenta armazena esses dados em uma base de objetos persistentes.

O armazenamento é feito através de três arquivos de objetos. O primeiro armazena os objetos *Eapplication*, o segundo os objetos *Eframewok* e o terceiro objetos *Econnect*. O objeto *Econnect* representa as relações de um framework com as aplicações desenvolvidas a partir dele, como também relaciona uma aplicação com os frameworks usados no seu desenvolvimento. A figura 1 mostra, através de um diagrama de classes, a estrutura desses relacionamentos.

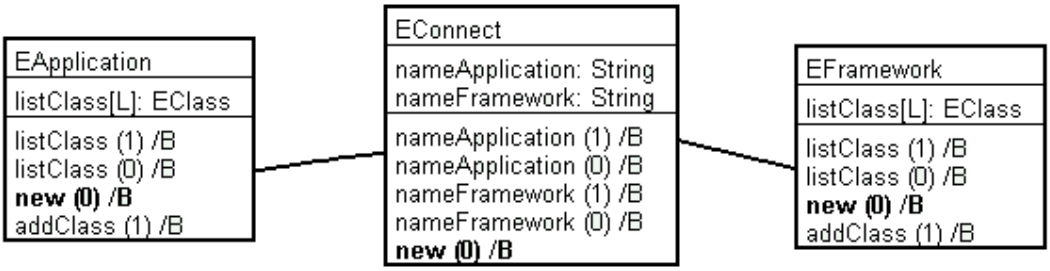


Figura 1 – Diagrama de Classe (Econnect)

A figura 2, a seguir, mostra o relacionamento da ferramenta de análise desenvolvida com o framework *Ocean* [8]. Isso é observado no ambiente SEA pela cor vermelha dos contornos das classes *OceanFileManager, OceanAnalyser, OceanTool, Specification e SpecificationOfElement,* constituindo o conjunto de classes externas provenientes do framework *Ocean*.

A classe *OceanFileManager* implementa serviços de escrita, padronizada em arquivos de saída de ferramentas de análise do framework *Ocean*. A classe *OceanTool* implementa as funcionalidades genéricas das ferramentas do framework. Serve como superclasse para qualquer ferramenta a ser desenvolvida. A classe *OceanAnalyser* é uma especialização da classe *OceanTool* e representa a superclasse para as ferramentas de análise de especificações.

A classe *SpecificationElement* representa a superclasse de todos os elementos de especificação possível, tais como: classes, atributos, métodos, relacionamentos, etc. A classe *Specification* representa a especificação em si, com todos os elementos de especificação agregados. Cabe ressaltar que a representação de uma especificação (classe *Specification*) e seus elementos (*SpecificationElement*) , assim como seus relacionamentos, no modelo da ferramenta de apoio ao uso de framework, não reproduzem a estrutura de meta-modelo do framework *OCEAN*, por questão de simplificação - maiores detalhes podem ser vistos em [8].

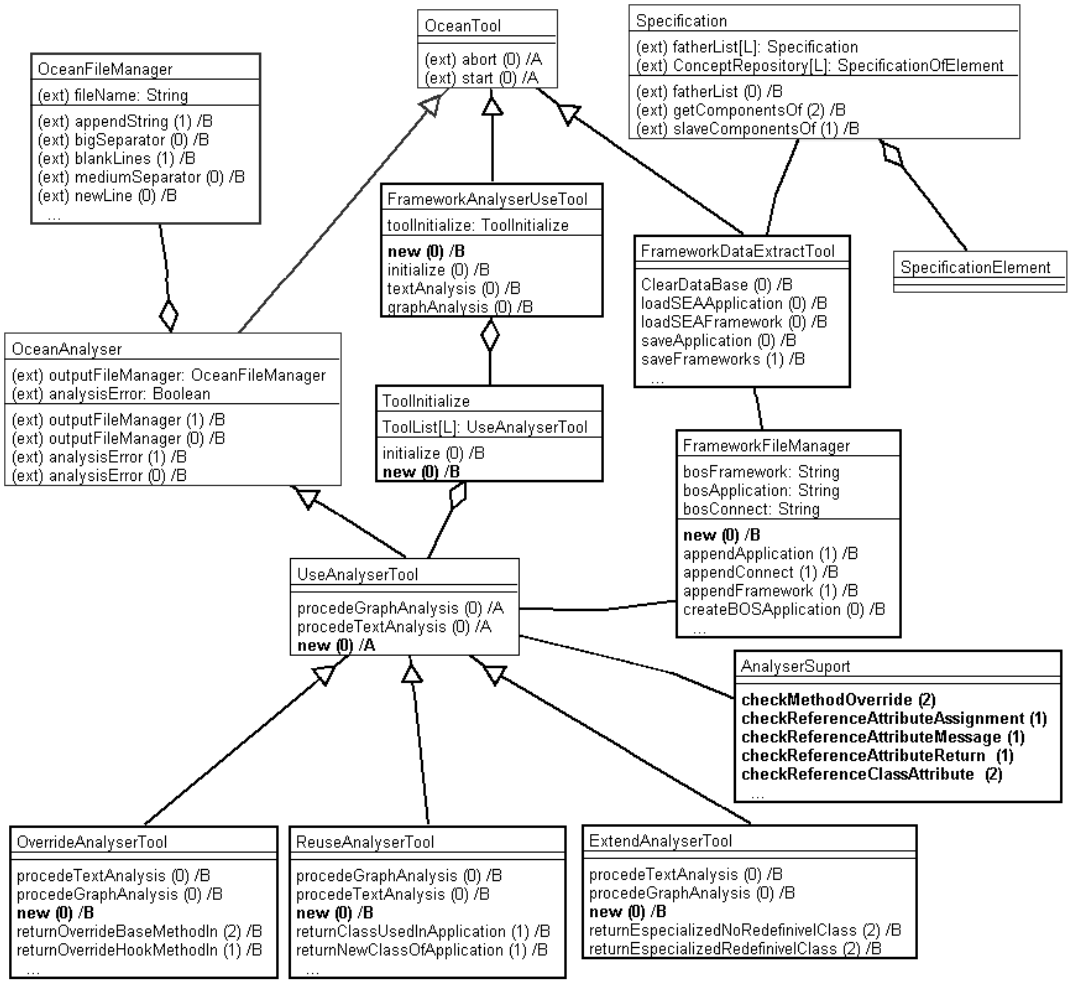


Figura 2 – Diagrama de Classes – Detalhe da ferramenta de análise de uso de frameworks

4.2.1 Análise dos dados de especificações

A função de análise das informações da ferramenta é representada pela classe *UseAnalyserTool*, que é superclasse abstrata das classes concretas responsáveis, efetivamente, pela avaliação e comparação dos dados das especificações de aplicações, desenvolvidas sob um dado framework. A subclasse *OverrideAnalyserTool* é responsável pela avaliação das características de sobreposição de artefatos do framework, através aplicações desenvolvidas a partir dele. Essas medidas dizem respeito à avaliação dos artefatos dos frameworks que foram previamente previstos para serem sobrepostos e que efetivamente foram sobrepostos, ou ainda, artefatos que foram sobrepostos pelas aplicações e que não foram previamente previstos para serem.

A subclasse *ReuseAnalyserTool* tem o papel de avaliar o montante de artefatos dos frameworks que foram reusados nas especificações das aplicações, possibilitando, assim, que seja obtido um percentual de reuso, e em contrapartida, um percentual do esforço de desenvolvimento despendido para o desenvolvimento das aplicações sob um determinado framework.

A subclasse *ExtendAnalyserTool* implementa a avaliação referente ao montante de artefatos dos frameworks, que foram estendidos pelas aplicações desenvolvidas. Essas

medidas mostram o montante de artefatos do framework que foram estendidos pelas aplicações.

A ferramenta foi desenvolvida com uma organização de classes tal, que se for necessário inserir um novo grupo de medidas, será necessário somente definir uma nova subclasse de *UseAnalyserTool* e acrescentá-la à lista de ferramentas de *ToolInitialize,* que contém a lista de subclasses *UseAnalyserTool.* Assim, a classe *ToolInitialize* é responsável pela instanciação de cada elemento dessa lista, evitando que a inclusão de um novo conjunto de medidas exija alteração da classe *FrameworkAnalyserUseTool*, responsável por implementar a interface da ferramenta de avaliação com o usuário.

4.3 Informações obtidas com a ferramenta

A ferramenta possibilita duas formas de disponibilizar as informações obtidas. A primeira consiste em um relatório com dados do framework e de cada aplicação desenvolvida a partir dele. A segunda, em um arquivo texto com dados formatados em registros que são usados como dados de entrada para o mecanismo de visualização gráfica. A seguir serão mostrados exemplos desses formatos de saída.

4.3.1 Informações no formato de relatório

O quadro 4.1 mostra um trecho do relatório, gerado em arquivo texto, pronto para leitura do usuário final da ferramenta. Os valores foram obtidos através de um estudo de caso, que consistiu na avaliação da especificação de um framework para a geração de jogos de tabuleiro e quatro aplicações desenvolvidas sob esse framework (jogos).

```
Nome da aplicação: corrida2
---------------------------------------------------------------------------------------------------
Número de classes exclusivas da aplicação = 3
        JcorridaInterface, JcorridaBoard, JCorridaAutomPlayer
---------------------------------------------------------------------------------------------------
Número de classes do framework referenciadas pela aplicação = 15
        GameInterface, Board, PlayerAutomatic, PositionDefaultView, Player,
        PlayerDefault, PositionDefault, PieceDefault, Piece, Position, Dice, DiceSixFaces,
        Model, ClickModel, View
---------------------------------------------------------------------------------------------------
Número de classes do framework referenciadas de forma indireta pela aplicação = 5
        ApplicationModel, DiceDecorator, Equipment, Controller, Sensor
---------------------------------------------------------------------------------------------------
Número de classes do framework não referenciadas pela aplicação = 8
        Account, ControllerWithMenu, ClickController, PositionContents, PositionDecorator
        DiceSixFacesView, FraGDefaultImageRepository, GraphicsContext
---------------------------------------------------------------------------------------------------
Total de classes referenciadas pela aplicação = 20
Total de classes da aplicação = 23
Percentual de reuso de classes do framework na aplicação = 88.46
Percentual de desenvolvimento de novos artefatos na aplicação = 11.53
```

Figura 3 – Dados no formato de relatório

4.3.2 Informações no formato de registros

A segunda possibilidade de saída de informações consiste em um arquivo texto com registros, contendo dados que serão interpretados pelo mecanismo de visualização gráfica. No quadro 4.2 é apresentado um exemplo desse formato de arquivo.

```
...
[R2]
corrida2
[R21]
3
JCorridaInterface
JCorridaBoard
JCorridaAutomPlayer
[F21]
…
```

Figura 4 – Dados no formato de registro de exportação

4.4 Mecanismo de visualização gráfica

Esse mecanismo foi desenvolvido como complemento à ferramenta extração de dados. Atua como um browser que permite ao usuário navegar pelo conjunto de informações obtidas em uma análise, possibilitando centrar o foco da atenção no conjunto das aplicações analisadas, em apenas uma, na sobrescrita de métodos etc. Sua ação consiste em ler um arquivo contendo os dados da avaliação de uso de um framework, em formato de registro e apresentá-los ao usuário. A seguir são apresentados alguns exemplos de como esse mecanismo apresenta informações. A figura 5 mostra a visualização dos dados no formato hierárquico.

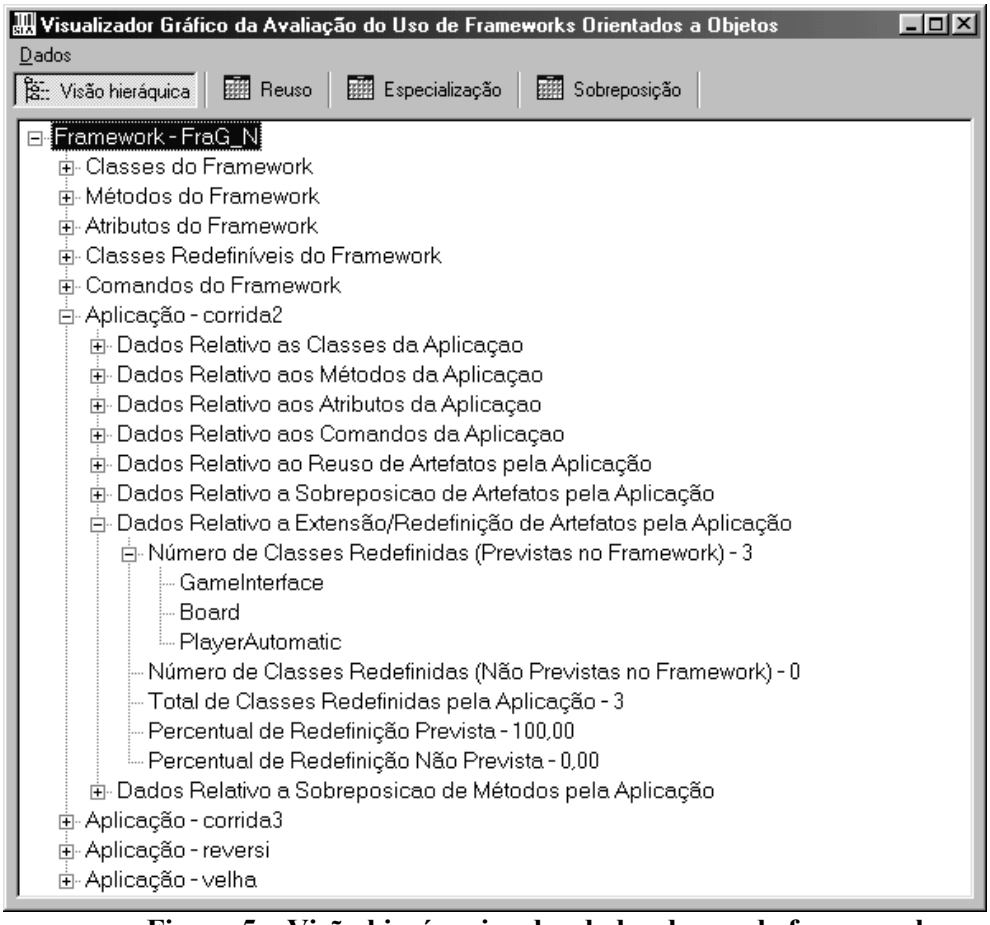
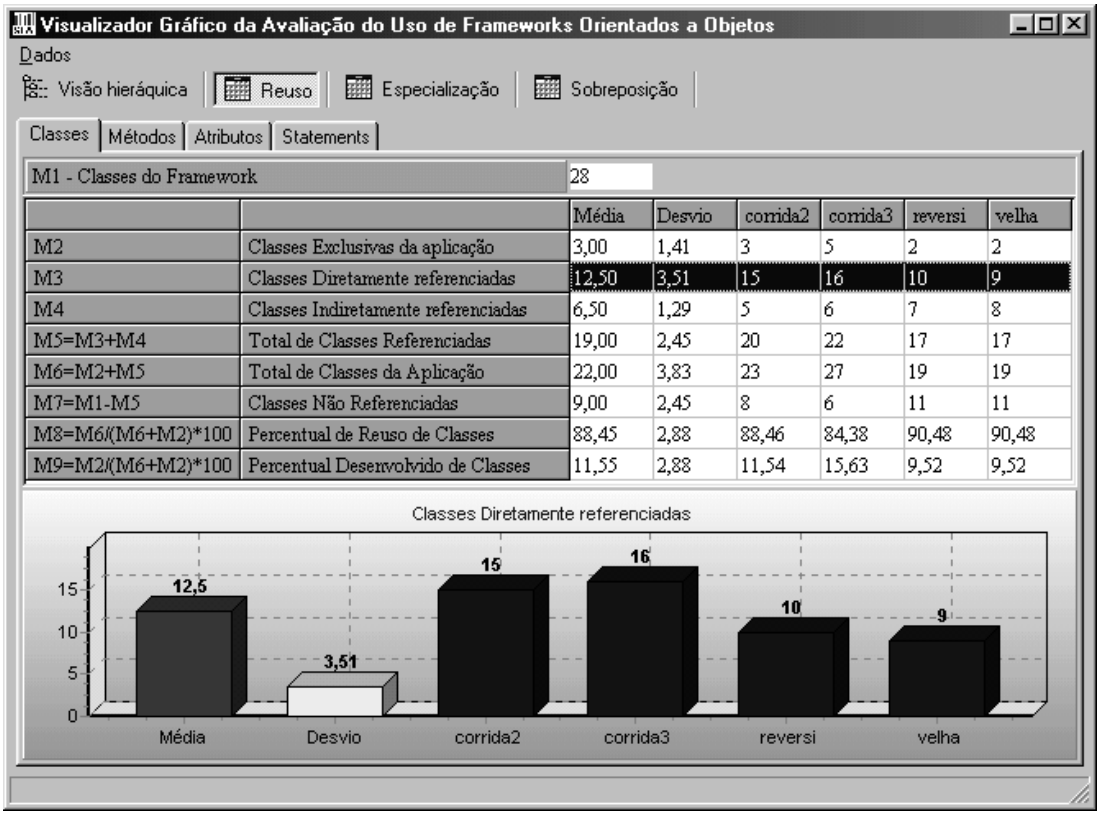


Figura 5 – Visão hierárquica dos dados de uso de framework

A figura 6, abaixo, mostra a visão dos dados de uso de framework através de tabela e gráfico de barras. Nessa visão, é possível observar o valor obtido em cada métrica para cada aplicação desenvolvida sob um framework. É possível também observar a média e o desvio padrão de cada métrica em relação a todas aplicações desenvolvidas sob o mesmo framework.



		Média	Desvio	corrida2	corrida3	reversi	velha
M2	Classes Exclusivas da aplicação	3,00	1,41	3	5	2	2
M3	Classes Diretamente referenciadas	12,50	3,51	15	16	10	9
M4	Classes Indiretamente referenciadas	6,50	1,29	5	6	7	8
M5=M3+M4	Total de Classes Referenciadas	19,00	2,45	20	22	17	17
M6=M2+M5	Total de Classes da Aplicação	22,00	3,83	23	27	19	19
M7=M1-M5	Classes Não Referenciadas	9,00	2,45	8	6	11	11
M8=M6/(M6+M2)*100	Percentual de Reuso de Classes	88,45	2,88	88,46	84,38	90,48	90,48
M9=M2/(M6+M2)*100	Percentual Desenvolvido de Classes	11,55	2,88	11,54	15,63	9,52	9,52

Figura 6 – Visão de tabela/gráfico dos dados de uso de framework

É importante observar que a ferramenta de extração das medidas usa como base de investigação as especificações de projeto e não, código fonte. Isso dá aos desenvolvedores do framework e das aplicações, a possibilidade de fazer medições e investigações a qualquer momento do desenvolvimento.

5 Síntese de um estudo de caso

Como parte do trabalho de pesquisa foi desenvolvido um estudo de caso que consistiu na avaliação da especificação de um framework e quatro aplicações desenvolvidas sob esse framework. Esse experimento teve como objetivo a aplicação das medidas propostas e validação da ferramenta de extração, através de um caso real de desenvolvimento de software. A especificação do framework foi desenvolvida através de engenharia reversa de um framework (FraG – framework para jogos de tabuleiro) e quatro jogos desenvolvidos sob esse framework (Jogo da Corrida, Jogo da Corrida Plus, Jogo da Velha e o Jogo Reversi) [9], desenvolvidos em *Smalltalk*, especificados no ambiente SEA. Estão disponíveis no endereço http://www.inf.ufsc.br/~ricardo/download/frag_analysis.zip o conjunto de medidas obtido e a ferramenta de visualização gráfica.

O estudo de caso mostrou que, mesmo se tratando de um framework relativamente pequeno e de apenas quatro aplicações instanciadas a partir desse, os resultados obtidos com o

uso da ferramenta de análise foram bastante satisfatórios. Através das métricas obtidas foi possível detectar boas características do framework, mas também foram detectados alguns problemas nas aplicações e no próprio framework. Alguns desses resultados são descritos a seguir.

O framework conseguiu abranger os conceitos do domínio de aplicação, pois não foram verificadas classes nas aplicações sem relação de herança com classes do framework. Em relação a características dos conceitos (atributos), o framework se mostrou bastante satisfatório, com baixo número de novos atributos nas aplicações. A média de reuso de atributos verificada foi de 96,3%, com desvio padrão, 3,3%. O número de atributos criados em algumas aplicações foi diferente de zero. Nas aplicações com novos atributos foram detectadas duas situações de falta de característica de conceito no framework: número de posições do tabuleiro, que pode ser variável, e conjunto de posições afetadas por uma ação do usuário, ambas características presentes em apenas algumas aplicações do domínio e que não foram previstas no framework. Em relação à avaliação das aplicações, não foi verificada criação imprópria de atributos, o que reflete bom uso do framework.

Foi verificada uma situação de criação imprópria de subclasse na aplicação "Corrida 3". Investigando a causa na especificação de projeto da aplicação, concluiu-se que se tratava de mau uso do framework: criação de subclasse ao invés de delegação, que era a solução para o caso, prevista no projeto do framework.

Um problema do projeto do framework difícil de verificar sem utilização de métricas é a proporção elevada de métodos regulares sobrescritos que não são classificados como *hook*. Verificando os números, 20.83% dos métodos sobrescritos são regulares. Apenas um quarto deles, 5.21%, são também classificados como *hook*. Examinando o projeto do framework, pode-se concluir que ocorreu má classificação de métodos: alguns métodos que deveriam Ter sido classificados como template foram classificados como regulares. Este tipo de conclusão é praticamente impossível de obter sem uma avaliação objetiva, baseada em números.

Com os dados capturados e com as conclusões obtidas a partir de sua análise, observou-se que a ferramenta de análise apresentou bons resultados, permitindo que aspectos impróprios nas aplicações e no próprio framework pudessem ser investigados e esclarecidos. A ferramenta não aponta erros ou acertos diretamente, porém permite que situações que dificilmente poderiam ser detectados manualmente possam ser analisadas e investigadas de forma objetiva, permitindo avaliar as suas causas.

6 Trabalhos Correlatos

Nessa seção são apresentados trabalhos descritos na literatura, relacionados a métricas e ferramentas desenvolvidas com a finalidade de medir ou auxiliar o uso de frameworks orientados a objetos.

Em [4], foi produzida uma coletânea de métricas para software orientado a objetos e implementada uma ferramenta que automatiza a extração dessas métricas, denominada MET (*Metrics Extraction Tool*). Essa ferramenta tem como objetivo dar apoio à avaliação de especificações orientadas a objetos, nas fases de análise e projeto. O conjunto de métricas implementadas contempla fatores relacionados com herança, polimorfismo, acoplamento e coesão. Apóia a avaliação de software orientado a objetos em geral e não trata aspectos específicos de frameworks.

Em [1], foram propostos três métodos para o acompanhamento e avaliação da evolução de frameworks orientados a objetos, com base no histórico obtido em diferentes versões do framework BGW (domínio de telecomunicações). O primeiro método consiste na identificação da evolução histórica do framework, através de uma adaptação do método

proposto por [5] para que pudesse ser aplicado em software orientado a objetos. O segundo método mede a taxa de estabilidade do framework, através da avaliação das mudanças nas colaborações entre classes e mudanças de estruturas hierárquicas das classes. O terceiro método mede o esforço de desenvolvimento e uso de frameworks. Esse esforço é medido durante o desenvolvimento do framework e em todas as versões posteriores, através do parâmetro horas/homem. O foco do trabalho é analisar a evolução de um framework ao longo de suas versões.

Em [2], foi proposta uma ferramenta de análise dinâmica de aplicações desenvolvidas sob frameworks que ajuda o entendimento do framework, através de meta-objetos que mostram o fluxo de execução, extraído do fluxo de mensagens entre os objetos das aplicações. A ferramenta extrai essas informações do código fonte do framework e de aplicações sob ele desenvolvidas. A abordagem é voltada a apoiar o aprendizado do uso de um framework. Além das classes de frameworks e aplicações, disponíveis no código fonte analisado, a ferramenta fornece o fluxo de controle observado na execução das aplicações. A análise depende da atuação usuário da ferramenta, que caso não conheça o framework, terá dificuldade para definir os caminhos de execução significativos. Além disso, a compreensão de classes e métodos de frameworks e aplicações é altamente dependente de análise de código.

Em [2], é proposto o uso de padrões de projeto para auxiliar a compreensão de frameworks, através de uma ferramenta que reconhece os padrões na estrutura do framework. Essa ferramenta armazena a experiência de uso em várias aplicações, produzindo uma base de conhecimento expressa através de regras escritas em Prolog, capazes de reconhecer padrões. Usa heurística, inserida por especialistas, para auxiliar a distinção de padrões semelhantes. A ferramenta representa em Prolog as relações estáticas e dinâmicas de padrões.

Em [3], é apresentada a aplicação de um conjunto de métricas de projetos orientados a objetos como suporte ao desenvolvimento de frameworks orientados a objetos. Para isto é utilizado um modelo de qualidade denominado *multi-metrics*[7], que tem como objetivo obter um projeto de melhor qualidade. O trabalho apresenta a combinação de princípios, regras e métricas de projetos de diferentes autores dentro do modelo de qualidade.

Não foi identificado nenhum trabalho com foco semelhante ao do descrito nesse artigo. Isto é, que obtenha métricas a partir do histórico de desenvolvimento de aplicações sob um framework para subsidiar desenvolvedores de aplicações tanto no sentido de aprender a usar um framework quanto no sentido de avaliar o produto desenvolvido e que também apóie desenvolvedores de frameworks, na avaliação da aderência de frameworks aos respectivos domínios. Outro aspecto característico do trabalho desenvolvido é o tratamento de especificações de projeto, e não código fonte, para extrair informações.

7 Conclusão

Apresentou-se um conjunto de medidas a serem extraídas a partir de especificações de projeto de frameworks orientados a objetos e de especificações de aplicações desenvolvidas sob esses frameworks. Essas medidas possibilitam a comparação estatística de características de uma dada aplicação com outras aplicações especificadas sob o mesmo framework. Esses dados subsidiam o desenvolvedor da aplicação na avaliação do seu trabalho de desenvolvimento, comparando-o com outros desenvolvimentos existentes. Os dados que representam o histórico

[7] Modelo de qualidade que combina métricas, com regras, princípios e características de projeto orientado a objetos.

de desenvolvimento de aplicações podem ser utilizados também pelo desenvolvedor do framework, como subsídio à análise da aderência do framework ao domínio tratado.

Para viabilizar a extração e o armazenamento dos dados relativos às especificações de frameworks e suas respectivas aplicações, foi implementada uma ferramenta que automatiza essas tarefas, permitindo que elas sejam realizadas quantas vezes forem necessárias, durante a fase de especificação das aplicações, com quase nenhum impacto de esforço e tempo.

Ter instrumentos de medição durante o processo de desenvolvimento de qualquer software é valioso, e, em alguns casos, crucial. No caso específico do desenvolvimento de frameworks orientados a objetos e suas instâncias, os dados estatísticos fornecem um mecanismo comparativo que pode auxiliar tanto o desenvolvedor do framework, quanto o desenvolvedor das aplicações. A ferramenta implementada torna o levantamento estatístico viável no que se refere a tempo e esforço, permitindo a obtenção dos dados a qualquer momento durante a fase de especificação dos artefatos de software.

Bibliografia

1 BOSCH, Jan; MOLIN, Peter; MATTSSON, Michael; BENGTSSON, PerOlof; FAYAD, Mohamed E.*Bilding* **Application Framework: Object Oriented Foudations of Framework Design**. 1999.

2 CAMPO, Marcelo, et al. **Framework Comprehension and Design Patterns: A Reverse Engineering Approach**. 9th International Conference on Software Engineering and Knowledge Engineering, Madrid, España, June 1997.

3 ERNI, K., Lewerentz, C. **Applying design-metrics to object-oriented frameworks,** in Proc. 3rd International Software Metrics Symposium. Los Alamitos, CA, USA, IEEE Comput. Soc. Press, 1996.

4 FONSECA, Wannessa R.. **Ferramenta de extração de métricas para apoio à avaliação de especificações orientadas a objetos.** Dissertação de Mestrado, CPGCC/UFSC, 2002.

5 GALL, H.; et al; **Software Evolution Observations Based on Product ReleaseHistory**. Proceedings of the Conference on Software Maintenance, 1997, pp. 160-166

6 JOHNSON, Ralph E.; FOOTE, Brian. **Disigning reusable classes**. Journal of Object-Oriented Programming, p. 22-35, June/July 1998.

7 ROCHA, A. R. C; MALDONADO, J. C; WEBER, K. C. **Qualidade de Software.** São Paulo, Pretince Hall, 2001.

8 SILVA, Ricardo Pereira e. **Suporte ao desenvolvimento e uso de frameworks e componentes**. Porto Alegre. PPGC da UFRGS, 2000. Disponível em http://www.inf.ufsc.br/~ricardo/download/tese.pdf. Acesso em 12/2003

9 SILVA, Ricardo Pereira, PRICE, Roberto Tom***. O uso de técnicas de modelagem no projeto de frameworks orientados a objetos***. In: Proceedings of 26th International Conference of the Argentine Computer Science and Operational Research Society (26th JAIIO) / First Argentine Symposium on Object Orientation (ASOO'97). Buenos Aires, Argentine: aug. 1997. p.87-94.

10 WEBER, K. C.; ROCHA, A. R. C, **Qualidade e produtividade em software**. São Paulo, Makron Books, 1999.