Iniciação Científica CESUMAR
Jul./Dez. 2009, v. 11, n. 2, p. 159-168

CONSTRUÇÃO DE UM FRAMEWORK PARA O DESENVOLVIMENTO DE APLICAÇÕES WEB

Lincoln Fernandes Paulino dos Santos*
Aline Maria Malachini Miotto**

RESUMO: Atualmente, com o grande avanço da tecnologia, cada vez mais as empresas estão sentindo a necessidade de ferramentas que lhes forneçam velocidade, agilidade, segurança e também uma grande capacidade de portabilidade. Da mesma forma, a procura por sistemas Web também tem obtido um alto crescimento, principalmente pelo fato de tais aplicações possibilitarem a seus usuários a facilidade de acessarem seus dados "em qualquer lugar e a qualquer momento", e isso, é claro, de uma forma segura. Assim, visando oferecer suporte à agilidade no desenvolvimento de sistemas, à portabilidade e também à grande exigência de qualidade exercida sobre os sistemas na atualidade, este trabalho tem como objetivo principal apresentar um framework para aplicações Web, desenvolvido na linguagem PHP. Para alcançar tal objetivo foi realizado um estudo que se baseou em definir e descrever o que é um framework e também apontar os principais frameworks desenvolvidos em PHP. Por meio deste estudo foi possível identificar quais as técnicas e quais os melhores padrões que poderão ser utilizados para a construção do framework proposto.

PALAVRAS-CHAVE: Frameworks; Linguagem PHP; Agilidade.

BUILDING A FRAMEWORK FOR THE DEVELOPMENT OF WEB APPLICATIONS

ABSTRACT: Nowadays with the great advancement of technology, more companies are feeling the need for tools that provide them speed, agility, security and a great capacity for portability. Similarly, the demand for Web systems has also obtained a high growth, mainly because of these applications make their users the convenience of accessing their data "anywhere and anytime", and in a secure way. Thus, to support the agility in the systems development, the portability and the high demand for quality exerted on the systems at this moment, this work has as main objective to provide a framework for Web applications, developed in PHP language. To achieve this objective it was performed a study that was based on defining and describing what is a framework, and also to indicate the main frameworks developed in PHP. Through this study, it was possible to identify the techniques and what are the best standards that can be used to build the proposed framework.

KEYWORDS: Frameworks; PHP Language; Agility.

INTRODUÇÃO

Com o passar dos anos e também com o grande avanço da tecnologia, cada vez mais as empresas estão sentindo a necessidade de ferramentas que lhes forneçam velocidade, agilidade, segurança e também uma grande capacidade de portabilidade.

* Discente do curso de Sistemas de Informação do Centro Universitário de Maringá – CESUMAR. E-mail: lincolnfpsantos@hotmail.com
** Docente dos cursos de Análise e Desenvolvimento de Sistemas, Sistemas de Informação e Sistemas para Internet do Centro Universitário de Maringá – CESUMAR. E-mail: amiotto@cesumar.br

Apesar do grande número de softwares desenvolvidos para o gerenciamento de processos, nas mais diversas áreas do conhecimento, em geral as empresas preferem sistemas que sejam customizados aos seus processos, para que, assim, consigam atingir um gerenciamento total dos mesmos.

Desta forma, a procura pela customização de sistemas e a exigência de uma excelência no poder de gerenciamento destas ferramentas têm obtido um alto crescimento. Da mesma forma que a procura por sistemas Web também tem obtido um alto crescimento, principalmente pelo fato de tais aplicações possibilitarem aos seus usuários a facilidade de acessarem seus dados "em qualquer lugar e a qualquer momento".

Diante das dificuldades encontradas para se construir uma aplicação *Web (Word Wide Web)* com agilidade que seja portável aos diversos navegadores, e considerando a crescente demanda pelo desenvolvimento de aplicações para Internet, o objetivo principal deste trabalho é apresentar os resultados parciais do desenvolvimento de um framework na linguagem PHP, que vislumbra tornar possível a construção de aplicações portáveis aos principais navegadores da atualidade e em um tempo significativamente menor. Espera-se também que o framework atenda aos aspectos de adaptabilidade e extensibilidade[1].

Dessa forma, este trabalho está organizado da seguinte forma: na seção 02 é apresentado um referencial teórico sobre frameworks. Na seção 03 são apresentados alguns dos resultados obtidos até o momento com o desenvolvimento do framework. E finalmente na seção 04 são apresentadas as principais conclusões trabalho realizado.

2 FRAMEWORKS

2.1 CONCEITUAÇÃO

Devido à grande abrangência que um *framework* utilizado para o desenvolvimento de sistemas possui, este termo normalmente é definido de diversas formas e também em diferentes contextos. Seguindo está ideia, Inácio Junior (2007, p. 21) define framework como "uma arquitetura de software semicompleta, reutilizável, e que pode ser adaptada para produzir aplicações personalizadas dentro de um domínio específico".

Dentre os muitos benefícios que um framework pode agregar aos desenvolvedores no processo de implementação de software, Fayad e Schmidt (1997) descrevem quatro como principais, sendo:

1. **Modularidade:** Ocorre pelo encapsulamento e também por gerar um impacto localizado de mudanças na implementação reduzindo assim o esforço na manutenção.
2. **Reusabilidadea:** Através das interfaces estáveis proporcionadas pelos frameworks é possível definir componentes genéricos que podem ser utilizados em novas aplicações.
3. **Extensibilidade:** É a capacidade que uma aplicação possui em ser estendida para acomodar um requisito previamente não identificado.
4. **Inversão de controle:** Beneficio este que, conforme Silva (2006), ocorre da seguinte forma: tradicionalmente o desenvolvedor, tomando um conjunto de bibliotecas ou componentes, cria um programa principal que invoca esses componentes quando necessário. Ao contrário disso, nos sistemas baseados em frameworks os sistemas principais são reusados incumbindo o desenvolvedor de somente definir quais os componentes serão utilizados pelo sistema.

2.2 CLASSIFICAÇÃO DE FRAMEWORKS

Segundo Fayad e Schimidt (1997), os frameworks são classificados de acordo com o modo como os mesmos serão empregados. Desta forma, os frameworks podem ser classificados de acordo com o seu escopo nas seguintes categorias:

2.2.1 System Infrastructure Frameworks

São frameworks utilizados no desenvolvimento de sistemas operacionais e interface com o usuário. Estes frameworks têm como função principal simplificar a construção da infra-estrutura destes sistemas de forma que seja possível maximizar a portabilidade e eficiência dos mesmos.

Suavê (1999) define este tipo de framework, como frameworks de aplicação, também conhecidos como frameworks horizontais. A Figura 1 ilustra o enfoque que os frameworks de aplicação (*System infrastructure frameworks*) oferecem quanto à sua utilização e generalização.

2.2.2 Middleware Integration Frameworks

São frameworks usados para realizar a integração de aplicações distribuídas e componentes. Para Fayad e Schmidt (1997), estes frameworks são utilizados para melhorar a habilidade no

[1] Segundo Silva (2006, p. 4), "entende-se como adaptabilidade, a capacidade de uma aplicação ter suas funcionalidades adaptadas às necessidades dos usuários e por extensibilidade, a possibilidade de uma aplicação poder ser estendida para acomodar um requisito não previamente identificado."

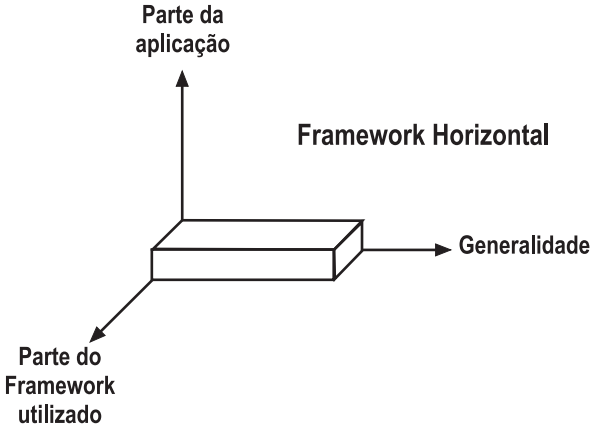

Figura 1. Frameworks de aplicação
Fonte: Suavê (1999)

desenvolvimento de software, visando a modularização, reutilização e extencibilidade da infra-estrutura de forma que tais aplicações funcionem perfeitamente em um ambiente distribuído.

2.2.3 Enterprise Application Frameworks

São frameworks focados para o desenvolvimento de aplicações comerciais e para a área de negócios. Conforme Fayad e Shmidt (1997), são frameworks caros para se desenvolver bem como também para se comprar.

Entretanto, podem fornecer um importante retorno sobre o investimento tendo em vista que possuem a capacidade de construir aplicações para os usuários finais. Estes tipos de frameworks também podem ser classificados como frameworks de domínio ou até frameworks verticais (SUAVÊ, 1999).

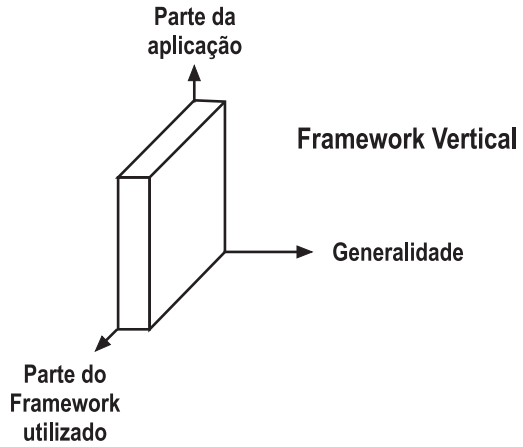

Figura 2 Frameworks de domínio
Fonte: Suavê (1999)

Na Figura 2 é ilustrada a estrutura dos frameworks de domínio (*Enterprise application frameworks*), que, contrariamente aos frameworks de aplicação, focam mais na parte da aplicação e do framework que está sendo utilizado do que na generalidade.

2.2.4 WhiteBox, BlackBox e GrayBox

Independentemente do escopo em que um framework está inserido, estes também podem ser classificados pela sua forma de reuso em 3 categorias (FAYAD; SCHMIDT, 1997).

Os frameworks caixa-branca, ora descritos *whitebox*, são os frameworks que se caracterizarão pela presente existência da orientação a objetos e ligação dinâmica. Nestes frameworks, o reuso se dá por meio da herança, onde usuários devem criar subclasses que estendem as classes abstratas e implementando então o comportamento dos métodos (MALDONADO *et al.*, 2009).

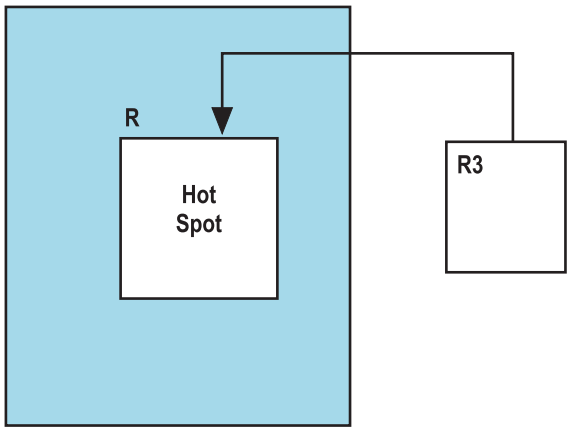

Figura 3. Framework caixa branca (*Whitebox*)
Fonte: Maldonado e colaboradores (2009)

A Figura 3 apresenta o funcionamento de um framework caixa branca cuja utilização é feita através de um único *Hot Spot* [2].

Os frameworks caixa-preta diferenciam-se dos frameworks caixa-branca por três aspectos: *menor flexibilidade*, tendo em vista que o reuso ocorre apenas na utilização das classes já existentes no framework; *maior facilidade de uso*, pois os desenvolvedores que os utilizam não necessitam interagir diretamente com o código do framework; *maior dificuldade de implementação*, isto porque, para o desenvolvimento desse tipo de framework, é necessário prever e implementar um amplo conjunto de casos de uso do domínio.

Na Figura 4 é ilustrado o funcionamento de um framework caixa-preta que, semelhantemente ao caixa branca, também possui apenas um *Hot Spot.* Entretanto pode-se observar que

[2] Conforme Maldonado e colaboradores (2009), um Hot Spot é um ponto de especialização de um domínio de aplicação, onde diferentes aplicações dentro de um mesmo domínio são distinguidas por um ou mais hot spots.

existem três alternativas presentes no framework para a implementação da responsabilidade R, necessitando, então, que o usuário escolha uma das três para obter uma aplicação especifica (MALDONADO et al., 2009).

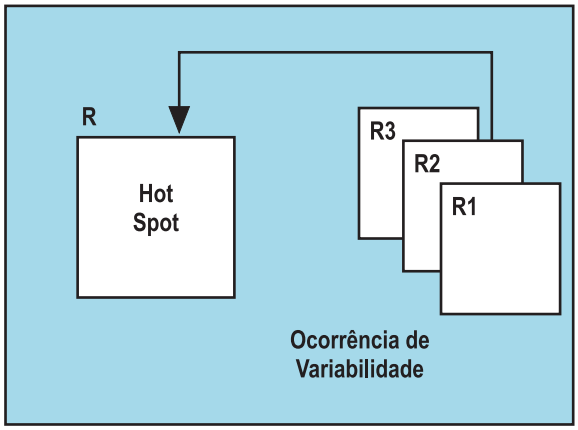

Figura 4. Frameworks caixa preta (*Blackbox*)

Fonte: Maldonado e colaboradores (2009)

Tendo em vista a discrepância existente entre os frameworks caixa-branca e caixa-preta, foi criada a classificação para os frameworks caixa-cinza (**graybox**), que classifica os frameworks que não se caracterizam como caixa-branca e nem caixa-preta.

> Esse tipo de framework permite a extensão tanto em termos de herança quanto em termos de composição, dependendo da necessidade da aplicação. Dessa forma, os frameworks caixa-cinza possuem flexibilidade e facilidade de extensão sem ter que prever muitos casos de uso e sem expor informações desnecessárias ao desenvolvedor (SILVA, 2006).

Algo muito comum no desenvolvimento de frameworks caixa-preto é que à dificuldade em se identificar e conhecer todo o domínio da aplicação faz com que, na grande maioria das vezes, a primeira versão destes frameworks seja caixa-branca, de forma que com a contínua melhora passe a ser caixa-cinza e posteriormente a isso caixa-preta.

2.3 VANTAGENS E DESVANTAGENS

Como foi visto até o momento, a utilização ou construção de um framework pode trazer inúmeras vantagens para seus usuários, tais como: redução de custos, maior reuso, melhor manutenabilidade, estabilização do código, extensibilidade, inversão de controle dentre outras supracitadas.

Entretanto, segundo (SUAVE, 1999; FAYAD; SCHMIDT, 1997), a utilização de frameworks pode acarretar em algumas desvantagens, tais como: a tarefa de se construir um framework é difícil e demanda muito planejamento para que o reuso seja alcançado; os benefícios da utilização de um framework são realizados a longo prazo; o custo de um treinamento e também o tempo necessário para se utilizar um framework de forma adequada são altos; uma boa documentação de manutenção e apoio são necessários (MALDONADO *et al.*, 2009).

Na Figura 5 é apresentando de uma forma simplificada uma comparação entre o custo e o beneficio da construção de frameworks.

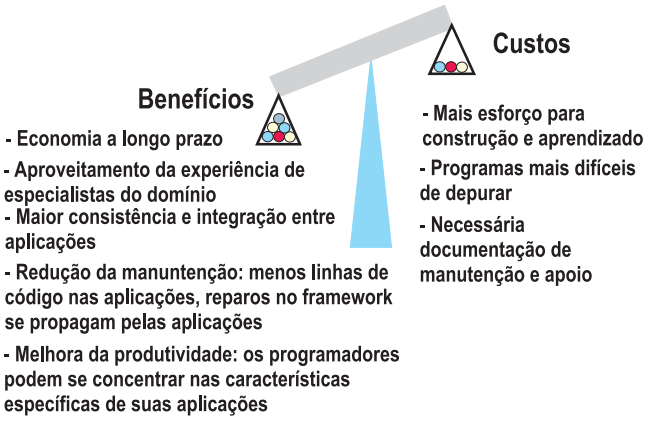

Figura 5. Custos e Benefícios na construção de um framework

Fonte: Maldonado e colaboradores (2009)

3 DESENVOLVIMENTO

3.1 DESENVOLVIMENTO DO FRAMEWORK

O framework ora desenvolvido, trata-se de um framework de aplicação, podendo ser classificado como um framework do tipo caixa-cinza. Isto tendo em vista que, em sua atual situação, estão presentes características como: a generalidade e a facilidade em estendê-lo, no sentido de que, para sua utilização, não é necessário que o usuário tenha contato direto com o código do framework.

O framework consiste em estrutura reutilizável cujo foco é permitir que usuários desenvolvam aplicações Web com maior agilidade sem que seja necessária a manipulação de códigos HTML (*HyperText Markup Language*) e CSS (*Cascading Style Sheets*).

3.1.1 Estrutura de Diretórios do Framework

Algo importante a ser ressaltado sobre o funcionamento do framework é que o mesmo está diretamente vinculado à estrutura em que as classes, arquivos de estilo, arquivos de configuração e também a forma que os arquivos desenvolvidos pelos usuários do framework estão organizados.

A organização e estruturação dos diretórios do framework, além de possuir suma importância para seu funcionamento, visam, também, fornecer aos usuários a possibilidade de desenvolverem aplicações seguindo padrões de desenvolvimento como MVC[3], separando a regra de negócios de uma aplicação dá interface da mesma.

A estrutura de diretórios do framework encontra-se organizada da seguinte forma (Quadro 1):

Quadro 1. Estrutura de diretório do Framework

DIRETÓRIO	DESCRIÇÃO
/app	Diretório onde ficará o código da aplicação.
/controllers	Arquivos que controlam as regras de negócio da aplicação
/views	Arquivos de interface.
Config.inc.php	Arquivo de configuração.
Menu.php	Arquivo onde deve ser criado o Menu da aplicação.
DeskTop.php	DeskTop da aplicação.
/LFClass	Código-fonte do Framework.
/style	Arquivos CSS do Framework.
/js	Javascripts do Framework.
/docs	Documentação do Framework.
Load.inc.php	Arquivo responsável por carregar as classes do Framework

Deve-se observar que os arquivos criados para o controle das regras de negócio de um determinado caso de uso (diretório /controllers) e a interface deste mesmo caso de uso (diretório / views) devem possui o mesmo nome. Exemplo: "/controllers/**CadCliente.php**" e "/views/**CadCliente.php**".

3.1.2 Diagrama de Classes

Uma das melhores e mais atuais formas de se documentar o projeto de uma aplicação é por meio de diagramas de classes, que descrevem as classes que formam a estrutura de uma aplicação e também o relacionamento existente entre as mesmas.

Conforme Shalloway e Trott (2004), a UML (*Unified Modeling Language*) é uma linguagem de visual (isto é, uma notação de desenho com semântica) utilizada para criar modelos de programas, os quais podem ser entendidos como uma representação diagramática dos programas. Esta linguagem de modelagem vem sendo utilizada como padrão de fato e oferece suporte para a construção dos diagramas de classes. Neste trabalho utilizamos, em função da sua notória aceitação tanto no meio acadêmico como no meio comercial, a UML como linguagem para modelagem do diagrama de classes do framework proposto.

A figura 6 apresenta o diagrama de classe do framework desenvolvido em seu estado atual. Neste diagrama pode-se verificar que todas as classes do framework estão relacionadas por meio de herança, que por sua vez, se dá a partir da classe principal *Object*.

Herança pode ser entendida como o compartilhamento de propriedades entre classes de forma que uma classe mais especializada (por ex. Radio) herde todas as propriedades de uma classe mais geral (por ex. Object).

Pode-se observar no diagrama abaixo, que visando facilitar ao máximo o trabalho dos desenvolvedores, bem como também fazer com que não seja necessária a manipulação de código HTML para o desenvolvimento de aplicações Web foram desenvolvidas classes para algumas das tags[4] mais utilizadas como: <b>, <p>,
, <table> entre outros.

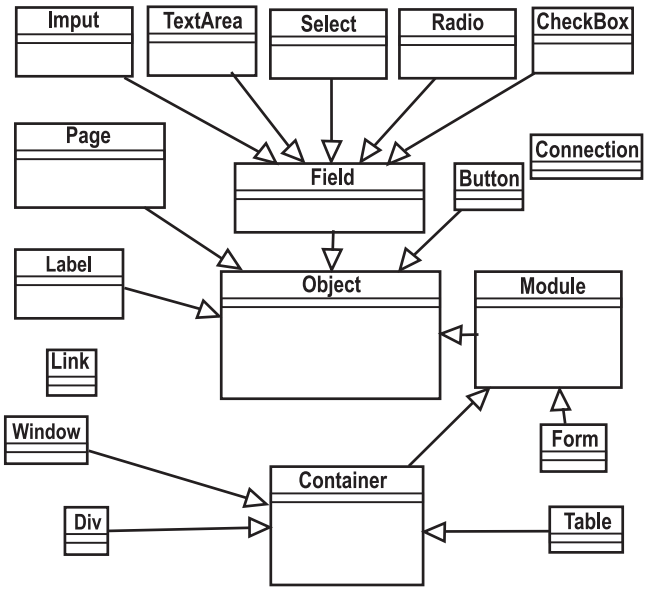

Figura 6. Diagrama de Classes do Framework

3.1.3 Exemplo de Utilização e Vantagens do Framework

Conforme apresentando anteriormente, para a utilização do framework não é necessário que seus usuários interajam diretamente com o código do mesmo, sendo necessário apenas realizar a estruturação dos módulos do sistema que se pretende desenvolver, informar os parâmetros necessários para que o framework se

[3] Conforme Minetto (2007), MVC é um acrônimo para Model, View, Controller (Modelo, Visão e Controlador) que se baseia na ideia de separar o desenvolvimento de uma aplicação em três partes ou camadas, ou seja, o Model onde é realizado o gerenciamento dos dados, a View onde se gerencia a saída gráfica (interface com usuário) e o Controller que interpreta as entradas do usuário, comandando a Visão e o Modelo para se alterarem de forma apropriada.

[4] Tags são normalmente especificas em pares, para delimitar um texto que deve sofrer alguma formatação. São identificadas pelos sinais de maior e menor (< > ou </ >), onde dentro dos sinais são especificados os comandos propriamente ditos. Importante ressaltar que a / entre os sinais determina que a tag está finalizando a marcação de um texto (RAMALHO, 1996).

comunique com o banco de dados e também realizar a codificação dos módulos através dos métodos já disponíveis no framework.

3.1.3.1 Criação e Acesso dos Módulos

Antes de qualquer coisa, é importante apresentar informações sobre a utilização e sobre a estrutura do framework: cada arquivo criado pelo usuário é tratado pelo framework como um modulo (classe *Module*). Isto porque, visando à agilidade no desenvolvimento todo modulo criado já possui por padrão o objeto *$module* sem que seja necessário se criar uma instância do mesmo para iniciar o desenvolvimento.

Para o melhor entendimento de como pode ser realizado o desenvolvimento de uma aplicação e quais as verdadeiras vantagens que o framework pode trazer ao desenvolvedor, dentro do diretório /app/views do framework, criaremos, como exemplo, um arquivo cadCliente.php, que, por sua vez, será tratado pelo framework como o modulo "cadCliente".

Por padrão, sempre que é aberta uma aplicação desenvolvida pelo framework, o primeiro módulo que será apresentado será o modulo DeskTop, ou seja, o arquivo DeskTop.php apresentado anteriormente na estrutura dos diretórios do framework. Para que possamos acessar nosso módulo cadCliente, é necessário que seja criado um menu dentro do arquivo Menu.php.

Seguindo o exemplo supracitado, para criamos um Menu que acesse no modulo cadCliente, basta que, dentro do arquivo Menu.php, seja utilizado o método addMenu através do objeto *$page* (já definido por padrão no framework) conforme mostrado abaixo:

```
$page->addMenu("menu","Cadastro","cadCliente")
```

O resultado da execução desta sequência de instruções pode ser observado na Figura 7.

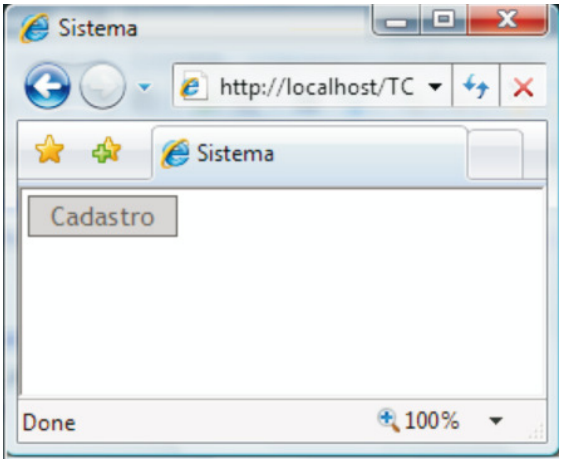

Figura 7. Menu do Sistema

Assim como os demais métodos presentes no framework, sempre que é adicionado um objeto, neste caso o "menu", o mesmo pode ser referenciado sem que seja instanciado pelo usuário. Ou seja, caso desejássemos criar sub-menus para nossos menu anterior basta executar o método addMenu() através do objeto "menu"criado anteriormente, conforme mostrado abaixo;

```
$page->addMenu("menu","Cadastro);
$menu->addMenu ("cad1","Cadastro de Clientes",
"cadCliente");
$menu->addMenu("cad2","Cadastro de Cidade");
```

O resultado da execução desta seqüência de instruções pode ser observado na Figura 8.

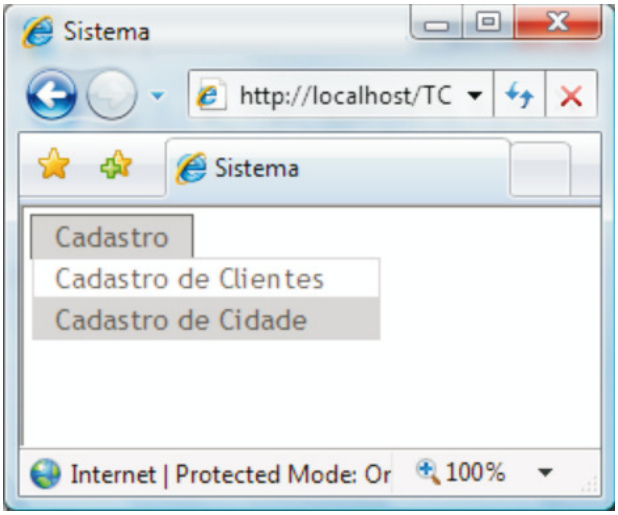

Figura 8. Menu e Sub-menu do Sistema

Deve-se observar que na segunda chamada do método *addMenu()* do código anterior, é passado como terceiro parâmetro o módulo cadCliente. Desta forma ao clicar neste item do menu o framework irá carregar no browser o módulo cadCliente. Para os demais itens do menu (linhas 1 e 3), onde não foram informados nenhum módulo, a seleção dos mesmos não resultará na apresentação de módulo algum.

3.1.3.2 Configuração e Manipulação de Banco de Dados

Para que seja realizada a comunicação como o banco de dados é necessário apenas que o usuário através do arquivo Config.inc.php informe dados como: Nome do *Host*[5], nome do banco de dados, nome do usuário e o password do usuário, conforme apresentado na Figura 9.

[5] Em sistemas de informação Host é qualquer máquina ligada à rede, seja ela, um roteador, computador pessoal (loscalhost) e servidores.

```php
<?php

// definições de conexão do banco
define('DB_NAME', '   ');          // nome do banco de dados
define('DB_USER', '   ');          // nome do usuário
define('DB_PASSWORD', '   ');      // password do usuário
define('DB_HOST', '   ');          // nome do host
define('DB_CHARSET', '   ');       // charset utilizado no banco
define('DB_COLLATE', '   ');       // collection utilizada no banco

?>
```

Figura 9. Arquivo Config.inc.php

Atualmente o framework comunica-se apenas com MySQL, que é um gerenciador de banco de dados padrão SQL[6] e que se tornou muito popular entre os desenvolvedores, pois, além de ser gratuito, encontra-se disponível para várias plataformas como Windows, Linux, Solaris, FreeBSD, OpenBSD e MacOS (ALVES, 2005).

Após realizadas tais configurações, o framework já permite que, a partir de qualquer parte e momento da aplicação, o usuário, por meio do objeto $conn, realize operações com o banco de dados definido no arquivo Config.inc.php. Isso, por meio dos métodos execute() e executeUpdate(). Em ambos os métodos é necessário que seja passada uma instrução SQL sendo que a diferença entre ambos se da pelo fato de que o método execute() retorna um Array com os dados obtidos.

Para exemplificar como os dados são retornados através do método execute(), imaginemos que, em um banco de dados que esteja se comunicando com o framework, exista um tabela chamada CLIENTE que possua os seguintes registros:

Nome	Idade
José	22
Maria	13

Ao executar a linha de código apresentada abaixo:

```
$dados = $conn->execute("SELECT * FROM CLIENTE");
```

A variável $dados passará a possuir o seguinte valor:

```
Array
(
   [0] => Array
      (
         [NOME] => José
         [IDADE] => 22
      )
   [1] => Array
      (
         [NOME] => Maria
         [IDADE] => 13
      )
)
```

3.1.3.3 Desenvolvimento de um Caso de Uso

Depois de realizada a criação do menu para que se possa ter acesso aos módulos, e a configuração para que seja realizada a comunicação com o banco de dados, para se construir uma aplicação Web basta realizar a codificação da mesma, o que, por sua vez, torna-se mais fácil com a utilização do framework, conforme pode ser visto abaixo:

Seguindo os exemplos realizados acima, será utilizado o arquivo cadCliente.php para exemplificar a facilidade e vantagens que a utilização do framework fornece para o desenvolvimento de aplicações Web.

Em primeiro lugar, será criado a interface com o usuário, ou seja, a visão do caso de uso. Para isso dentro do arquivo cadCliente.php criado dentro da pasta views (/views/cadCliente.php) será criado um formulário a partir do objeto $module, conforme dito anteriormente.

Normalmente, em uma aplicação Web desenvolvida em PHP e HTML, o código necessário para se criar um formulário neste nosso contexto seria necessário criar o código abaixo:

```
<form method="post" action="cadCliente.php"name="form">
```

[6] SQL (Structured Query Language) é uma linguagem padronizada utilizada para a definição e manipulação de bancos de dados relacionais.

```
... Obs: “Em um código HTML, nesta parte do código são inseridos todos os campos do formulário” ...
</form>
```

Utilizando o framework se torna dispensável escrever códigos HTML iguais ao descrito acima. É necessário apenas utilizar o método addForm(), passando por parâmetro o nome do formulário, conforme o código abaixo:

```
$module->addForm(“form”);
```

Após a criação do *form*[7] será criada a janela que será apresentada ao usuário. Algo importante a se observar é que assim como realizado na criação do menu anterior sempre que é criado um objeto através do framework, este passa ser um objeto, de forma que o mesmo através de seus métodos pode então adicionar outros objetos em seu contexto.

Assim para inserir uma janela na tela e dentro do formulário (no caso “form”) será utilizado o método addWindow(), passando por parâmetro o nome da janela, título da título, largura e altura, conforme pode ser visto abaixo:

```
$form->addWindow(“win”, “Cadastro de Clientes”, 500, 300);
```

O resultado destas duas linhas de código implementadas no arquivo cadCliente.php pode ser observado na Figura 9.

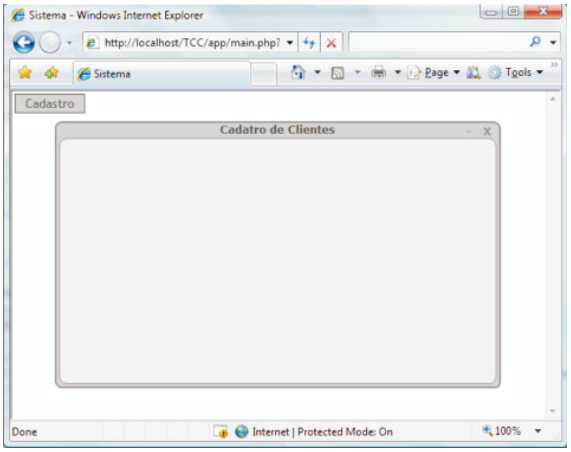

Figura 9. Window do Framework

Ao criar a janela, pode se notar algumas características como o posicionamento no centro da tela por default (o que por sua vez pode ser alterado através no método setPosition()), a possibilidade de movimentar a mesma na tela dentro do *browser* e também o botão “X” no canto superior da janela que serve para fechar o módulo atual e voltar para o último módulo aberto, neste nosso caso o módulo Desktop.

Conforme se pode observar, a partir do momento que será criado o formulário passando “*form*” (nome dado ao formulário) passa-se a acessar os métodos através de $form, da mesma forma, ao adicionar a janela passando o nome de “*win*”, todos os objetos pertencentes à janela serão adicionados através do objeto $win.

Desta forma, para inserir os campos dentro janela, basta utilizar o método addInput(), passando por parâmetro o nome do input, o tipo de dado que será informado no campo, largura, posição Y e posição X que será apresentado dentro da janela, conforme código abaixo:

```
$win->addInput(“nome”, “texto”, 200, 15, 10);
```

Algo importante que deve ser destacado sobre o método addInput(), é o segundo parâmetro (Tipo de Texto) que serve para definir qual será o tipo do texto a ser inserido no campo. Atualmente são configurados os seguintes tipos: **Text**: permite a inserção de qualquer caractere; **Number**: permite inserção apenas de números e cria uma máscara; **CEP**: permite inserção apenas de números e cria uma máscara (99999-999); **CPF**: permite inserção apenas de números e cria uma máscara (999.999.999-99); **CNPJ**: permite inserção apenas de números e cria uma máscara (99.999.999/9999-99); **Date**: permite inserção apenas de números e cria uma máscara (99/99/9999); **Hour**: permite inserção apenas de números e cria uma máscara (99:99); **Datehour**: permite inserção apenas de números e cria uma máscara (99/99/9999 99:99); **Phone**: permite inserção apenas de números e cria uma máscara ((99) 9999-9999); **Moeda**: permite inserção apenas de números e cria uma máscara de moeda com duas casas decimais.

Para definir um label[8] que indique qual informação será inserida no campo em questão, bata utilizar o método setLabel(), passando por parâmetro o label em questão, conforme código abaixo:

```
$nome->setLabel(“Nome”);
```

Assim, basta inserirmos os demais campos na tela, conforme mostrado abaixo:

```
$win->addInput(“nome”,”text”, 250, 15, 10);
$nome->setLabel(“Nome”);
$win->addInput(“idade”,”number”, 50, 15, 270);
$idade->setLabel(“Idade”);
$win->addInput(“cpf”,”cpf”, 150, 15, 330);
$cpf->setLabel(“CPF”);
```

Para inserir os botões utiliza-se o método addSubmit() passando o nome do botão, a legenda, a posição Y e posição X,conforme abaixo:

[7] Ferramenta do HTML que permite a criação de documentos interativos dentro do WWW. Permite que os dados informados pelo usuário sejam submetidos e processados.

[8] Label é um tag HTML que é utilizada juntamente com os campos do formulário e que possui a finalidade de definir um rótulo para tais elementos.

```
$win->addSubmit("btn_salvar","Salvar", 45, 280);
$win->addSubmit("btn_novo","Novo", 45, 380);
```

O resultado da execução destes códigos pode ser visto na Figura 10.

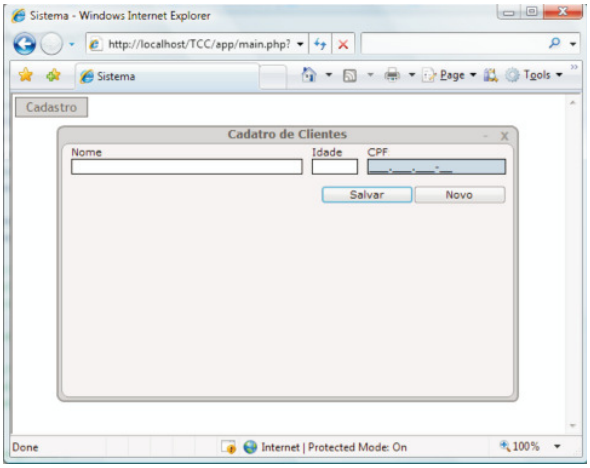

Figura 10. Tela do caso de uso, Janela e campos

Além dos campos e botões já inseridos, através do framework pode-se também facilmente criar tabelas, através do método addTable(), passando por parâmetro o nome da tabela, número de colunas e largura, altura, posição Y e posição X.

```
$win->addTable("table",4, 486, 205, 70, 1);
```

Após criada a tabela, pode-se então adicionar os *label's* em cada coluna através do método addLabel(), onde são passadas a linha, a coluna e o valor que será inserido na linha e coluna em questão, segue o código abaixo:

```
$table->addLabel(1,1,"Código");
$table->addLabel(1,2,"Nome");
$table->addLabel(1,3,"Idade");
$table->addLabel(1,4,"CPF");
```

O resultado da execução destes códigos é apresentado na Figura 11.

Assim, toda a parte de visão (toda interface com usuário) é finalizada com apenas 15 linhas de código fonte, o que, por sua vez, torna-se praticamente inviável de se fazer quando se programa diretamente em HTML e CSS, isso,, é claro deixando o código legível.

Feito isso, o próximo passo para se concluir o caso de uso ora exemplificado é realizar a código que tratará da regra de negócio, que, por sua vez, pode ser desenvolvido separadamente em um arquivo com o mesmo nome de onde foi programada a

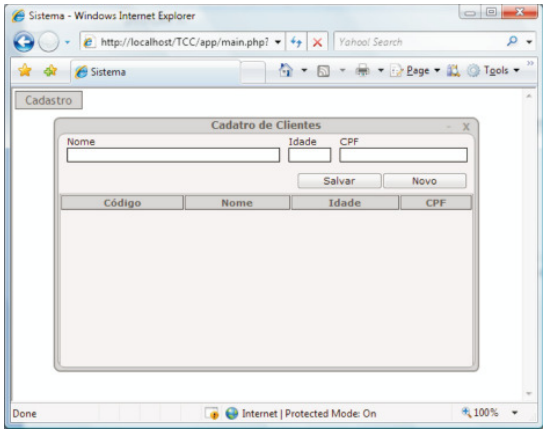

Figura 11. Tela do Caso de Uso completa

visão, só que na pasta de controllers, neste caso, "/controllers/cadCliente.php".

4 CONSIDERAÇÕES FINAIS

Com base no que foi apresentado ao longo deste trabalho, pode-se concluir que a construção e também a utilização de frameworks é uma solução útil quando se espera agregar a um software aspectos como o reuso, estruturação, flexibilidade e manutenabilidade além de agilidade no desenvolvimento. No entanto, como um passo futuro desta pesquisa deverão ser realizados experimentos para comprovar a viabilidade do uso do framework proposto.

Também se pode destacar que o framework proposto neste trabalho classifica-se como um framework de aplicação do tipo caixa-cinza, uma vez que apresenta características tais como: generalidade que permite desenvolver diferentes aplicações e facilidade de uso tendo em vista que, para sua utilização, não é necessário que o usuário tenha contato direto com o código do framework.

Deve-se observar também a necessidade de algumas alterações para que o framework possa então ser de classificados como um framework caixa-preta. Dentro deste contexto, como continuidade deste trabalho, serão realizadas alterações para que o framework se torne ainda mais flexível em sua forma de desenvolvimento (ou seja, se transforme em um framework caixa-preta) bem como em aspectos como comunicação com novos gerenciadores de banco de dados.

REFERÊNCIAS

ALVES, William Pereira. **Delphi 2005:** Aplicações de Banco de Dados com InterBase 7.5 e MySql 4.0.23. São Paulo, SP: Érica, 2005.

FAYAD, Mohamed; SCHIMIDT, Douglas C. **Object-Oriented Application Frameworks. Communications of the ACM**, v. 40, n. 10, Oct. 1997. Disponível em: <http://www.cse.wustl.edu/~schmidt/CACM-frameworks.html>. Acesso em: 07 maio 2009.

INÁCIO JUNIOR, Valter dos Reis. **Um Framework para Desenvolvimento de Interfaces Multimodais em Aplicações de Computação Ubíqua**. 2007. 119Fls. Dissertação (Mestrado em Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação – Universidade de São Paulo, São Carlos, São Paulo. Disponível em: <http://www.teses.usp.br> Acesso em: 01 maio 2009.

MALDONADO, José Carlos et al. **Padrões e Frameworks de Sofware**. São Paulo, SP: Universidade de São Paulo, [S. d.]. (Notas Didáticas). Disponível em: <http://www.icmc.sc.usp.br/~rtvb/apostila.pdf>. Acesso em: 19 maio 2009.

MINETTO, Elton Luis. **Frameworks para Desenvolvimento em PHP**. São Paulo, SP: Novatec, 2007.

RAMALHO, José Antonio Alves. **HTML Dinâmico**. São Paulo, SP: Berkeley Brasil, 1999.

SAUVÊ, Jacques Philippe. **Frameworks**. Paraíba, Campina Grande: Universidade Federal da Paraíba, 1999. (Notas de Aula). Disponível em: <http://www.dsc.ufcg.edu.br/~jacques/cursos/map/html/frame/oque.htm>. Acesso em: 19 maio 2009.

SHALLOWAY, Alan; TROTT, James R. **Explicando Padrões de Projeto:** Uma Nova Perspectiva em Projeto Orientado a Objeto. [S. l.]: Bookman, 2004.

SILVA, Elaine Quintino da. **Um Framework baseado em componentes para o desenvolvimento de aplicações Web e um processo de instanciação associado**. 2006. 167Fls. Tese (Doutorado em Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação – Universidade de São Paulo, São Carlos, São Paulo. Disponível em: <http://www.teses.usp.br>. Acesso em: 30 abr. 2009.

Recebido em: 30 Junho 2009
Aceito em: 09 Novembro 2008