PROPOSTA DE UMA ARQUITETURA DE SOFTWARE E FUNCIONALIDADES PARA IMPLEMENTAÇÃO DE UM AMBIENTE INTEGRADO DE DESENVOLVIMENTO PARA A LINGUAGEM PHP

Aline Martins CHAVES, Gabriel da SILVA*
Centro Federal de Educação Tecnológica de Bambuí-MG

RESUMO

A concepção de sistemas para Web confiáveis e de alta qualidade, requer, fundamentalmente, a adoção de uma linguagem de programação e de uma ferramenta de desenvolvimento adequadas. O presente trabalho propõe o estudo dos principais IDE existentes para a linguagem PHP e suas metodologias de implementação. Alguns IDE e editores, livres ou proprietários, implementados sob os paradigmas de programação estruturada ou orientada a objetos, são apresentados, expondo as principais características de cada um. Em seguida, testes foram realizados a fim de identificar as vantagens e desvantagens de cada um destes. As principais características que um IDE deve possuir e uma proposta de arquitetura para implementação de um IDE são apresentadas.

Palavras-chave: Ambiente de Desenvolvimento Integrado, IDE, software, PHP.

1. INTRODUÇÃO

A utilização de um Ambiente de Desenvolvimento Integrado (do inglês *Integrated Development Enviroment* - IDE) adequado para o desenvolvimento de software é de suma importância em um desenvolvimento de sucesso, principalmente quando o desenvolvimento é focado para Web, que geralmente apresenta uma maior complexidade que os sistemas *stand alone*.

Uma das linguagens mais utilizadas atualmente para o desenvolvimento de sites Web é a PHP. Entretanto, pela inexistência de um IDE eficiente para uso desta linguagem, um desenvolvedor PHP faz uso de uma série de ferramentas no processo de desenvolvimento de software. A alternância entre um aplicativo e outro costuma diminuir o rendimento do programador. Uma solução eficiente para este problema é encontrada quando todas, ou pelo menos a maioria das ferramentas necessárias, encontram-se num mesmo ambiente.

Atualmente, existem editores e IDE para PHP, mas estes não são completos. Cada um deles trata o desenvolvimento de aplicações de forma particular, porém análoga.

Este trabalho tem como objetivo analisar de forma exploratória alguns ambientes de desenvolvimento de software para a linguagem PHP, do ponto de vista do desenvolvedor, bem como da metodologia de desenvolvimento, a fim de propor uma série de requisitos funcionais e não funcionais para construção de um IDE. Também uma arquitetura de software para implementação do mesmo é proposta.

A escolha de um único ambiente não tem a intenção de limitar as opções dos desenvolvedores e usuários, mas servir como uma referência comum a todos. Este trabalho está inserido no âmbito do projeto "IDE4PHP – Metodologia e Implementação para Desenvolvimento de um IDE para a linguagem PHP", desenvolvido no Centro Federal de Educação Tecnológica de Bambuí, Minas Gerais.

2. DESENVOLVIMENTO DE SOFTWARE

No que diz respeito à construção do software, para Borges e Viana Preto (2006), "um paradigma de programação fornece e determina a visão que o desenvolvedor possui sobre a estruturação e execução do programa". Com relação ao licenciamento de software, atualmente existem duas grandes correntes, que são os software livres – SL e os software proprietários – SP. SL é qualquer software cuja licença garanta ao seu usuário liberdades relacionadas ao uso, alteração e redistribuição. Como a grande maioria de software proprietários não possui código aberto, não é possível personalizar o programa, verificar a

* gabrields@cefetbambui.edu.br

qualidade do código, realizar melhorias ou corrigir erros.

O presente trabalho buscou avaliar IDE cujo paradigma de desenvolvimento permita, inicialmente, a expansão ou modificação de suas funcionalidas, bem como uma maior facilidade na realização desta tarefa, isto é, IDE desenvolvidos sob o paradigma da orientação a objetos e distribuídos como software livre. A seguir são apresentados conceitos que permitem um maior entendimento das próximas seções.

2.1. IDE

De acordo com Sebesta (2000, p. 46), IDE pode ser identificado como um ambiente de desenvolvimento integrado que reúne características e ferramentas que dão apoio ao desenvolvimento de software, com o objetivo de agilizar o processo. Geralmente, IDE apresenta a técnica RAD (do inglês Rapid Application Development), que consiste em permitir que os desenvolvedores tenham um aproveitamento maior, desenvolvendo códigos com mais rapidez e facilidade. É integrado porque envolve pelo menos, editor, compilador e depurador.

2.2. Linguagem PHP

O presente artigo fundamenta-se na linguagem PHP (PHP.NET). Esta é uma linguagem para programar scripts do lado do servidor, incorporada à *HyperText Markup Language* – HTML, permite a criação de sites dinâmicos, possibilitando uma interação com o usuário. Esta linguagem trabalha em conformidade com a estrutura cliente servidor, na qual, o servidor é responsável por interpretar os *scripts* que compõem o documento solicitado, transformá-lo em código HTML e enviar o resultado ao cliente que fez a solicitação.

O PHP pode ser usado tanto para produção de *software* para *Web* quanto para aplicações *desktop*. Caso o desenvolvedor deseje usar alguns recursos avançados do PHP em aplicações do lado do cliente, poderá utilizar o PHP-GTK para desenvolver estes programas. E programas desenvolvidos desta forma serão independentes de plataforma. O PHP-GTK é uma extensão do PHP. Algumas características do PHP estão listadas a seguir:

- Gratuito e com código-fonte aberto
- Compatibilidade: O PHP pode ser utilizado na maioria dos sistemas operacionais, incluindo, Linux, Unix e Windows. Também é suportado pela maioria dos servidores Web atuais, principalmente Apache. Diversos bancos de dados são suportados pelo PHP, entre eles pode-se citar; MySQL, PostgreSQL, Sybase, Oracle, SQLServer e muitos outros.

2.3. Reuso e extensão de software

O reuso de código é um dos principais objetivos para desenvolvedores de software. É uma característica muito forte no que diz respeito a tempo e custo de desenvolvimento do sistema. O reuso pode se dar de diversas formas, se a arquitetura estiver bem organizada, cada componente computacional ou parte dele pode ser construída com vistas para reuso. Nos tópicos a seguir são apresentados alguns tipos de reuso.

2.3.1. API

API, do inglês *Application Programing Interface*, é comumente traduzida como Interface de Programação de Aplicativos. É um conjunto de rotinas e padrões estabelecidos por um software para utilização de suas funcionalidades por programas aplicativos, isto é: programadores que não querem envolver-se em detalhes da implementação do software, mas apenas usar seus serviços.

Segundo Goodman (2001) é freqüente dar à expressão API um sentido mais restrito e específico do que este, considerando API apenas as interfaces formadas por funções, reservando outros nomes para interfaces que façam uso de tecnologias mais complexas, como frameworks e componentes, que serão discutidos no próximo tópico.

2.3.2. Biblioteca

Uma das grandes vantagens da programação orientada a objetos é a utilização de bibliotecas de classes. Estas bibliotecas lembram as bibliotecas de código (procedimentos e funções), utilizadas na programação modular (estruturada). As bibliotecas de classes permitem uma capacidade muito maior de compartilhamento e reutilização de código, pois é possível criar-se subclasses para atender novas necessidades, em função das classes.

2.3.3. Framework

Pode-se dizer que um *framework* se baseia nas principais características da programação orientada a objetos: abstração de dados, polimorfismo e herança. Essas características são responsáveis por tornar o *framework* possível de ser feito, tendo como objetivo a reutilização de código, pois permitem a criação de classes genéricas. Os benefícios do uso de *frameworks*, além de reuso, da facilidade de manutenção e da inversão de controle, são a capacidade de extensão e a modularidade. A modularidade é atingida quando os detalhes de implementação são encapsulados pelo framework em classes fixas. Devido ao *framework* disponibilizar "esqueletos de métodos, permitindo que as classes possam ser estendidas". A extensão de um *framework* é uma característica fundamental para economizar tempo de programação.

2.3.4. Plugin

Segundo Lima *et.al.*(2004), a idéia de *plugin* está associada à extensão das funcionalidades da ferramenta, ou seja, "plugar" um pedaço de software ao que já existe, o qual irá possibilitar ao desenvolvedor realizar atividades além das que já eram possíveis. Com isso, é possível adicionar recursos a um ambiente, criar novos ambientes de programação para outras linguagens ou para um propósito específico. Um *plugin* pode ser desenvolvido e distribuído separadamente, pois, é a menor unidade de função de uma plataforma. Usualmente, uma ferramenta pequena pode ser escrita em apenas um *plugin*, porém pode ter funcionalidades divididas. Pode-se citar como exemplo, o Eclipse, onde tudo são *plugins*, o próprio Eclipse é constituído de um conjunto de *plugins* inter-relacionados (ECLIPSE, 2007).

3. AVALIAÇÃO DOS IDE

Foram realizados experimentos com seis ambientes de desenvolvimento existentes para a programação para a Web com suporte à linguagem PHP, a saber: Eclipse com plugin PHP –PHPEclipse (ECLIPSE, 2007), Delphi for PHP (CODE GEAR, 2007), Zend Studio (ZEND,2007), Dreamweaver (MACROMEDIA, 2007), PHPEditor (PHPEDITOR, 2007) e Anúbis (ESTEVARENGO,2007). Estes foram definidos por dois motivos: os cinco primeiros, após pesquisas realizadas a fim de se descobrir quais os mais utilizados pelos desenvolvedores; o Anúbis, por estar sendo utilizado em outro trabalho também integrante do mesmo projeto do qual o presente trabalho faz parte.

O objetivo destas avaliações foi o de encontrar as vantagens e desvantagens do uso de cada um destes ambientes, a fim de elencar quais as principais características devem ser contempladas na construção de um novo IDE para desenvolvimento com a linguagem PHP. Os ambientes foram instalados e uma aplicação teste foi desenvolvida em todos. Para a utilização dos mesmos foram utilizados tutoriais e documentação oficial, quando disponível. As avaliações também se basearam em entrevistas realizadas em fóruns de discussão, com desenvolvedores que as utilizam no seu dia-a-dia. A Tabela 1 apresenta os resultados da avaliação dos ambientes e algumas características consideradas importantes para os autores.

Tabela 1 – Características dos IDE/Editores

Característica	Ambientes					
	PHP ECLIPSE	**ZEND Studio**	**DELPHI for PHP**	**DreamWeaver**	**PHP Editor**	**ANÚBIS**
Proprietário, livre ou gratuito?	Gratuito	Proprietário	Proprietário	Proprietário	Gratuito	Livre
Código-fonte aberto	Não	Não	Não	Não	Não	Sim
Possui API	Sim	Não	Não	Não	Não	Não
Documentação	Sim	Sim	Sim	Sim	Sim	Não
Capacidade de extensão	Sim	Sim	Sim	Sim	Não	Sim

Algumas observações que destacaram alguns dos ambientes avaliados dizem respeito à facilidade de desenho e à necessidade de construção de pouco código. Entretanto, é importante destacar que nenhum dos IDE possui ferramentas que ofereçam num mesmo ambiente, facilidade de desenho e implementação de código.

4. RESULTADOS E DISCUSSÕES

As análises basearam-se nas facilidades oferecidas pelos ambientes, buscando algo mais produtivo, com recursos interessantes, não tendo uma análise profunda sobre suas funcionalidades.

O ambiente Eclipse possui facilidades de uso e possui API de extensão, através de sua API ele pode ser estendido a qualquer momento.

O Anúbis não possui nenhuma documentação útil ao desenvolvedor, é uma ferramenta para criação de aplicações *desktop*, sendo que o foco maior é em torno de aplicações *Web*.

O Zend Studio possui um ótimo analisador de código, um interpretador que mostra a saída do código sem ser preciso ir ao *browser*, mas é proprietário, o que dificulta para desenvolvedores.

No PHPEclipse encontram-se estas mesmas características do Zend, porém, ele é um produto livre e de código aberto.

Dos IDE avaliados o Eclipse apresenta-se como o mais completo e interessante, além de possuir uma documentação extensa. Os *plugins* do eclipse possuem todas as permissões do ambiente principal, a API oferece aos *plugins* acesso ao ambiente e é suficientemente completa. Contudo, expandir o IDE Eclipse pode ser trabalhoso, mas o próprio ambiente oferece ferramentas que dão apoio a isso, como também suporte de várias comunidades.

A partir das comparações feitas, é possível escolher qual o ambiente oferece maiores recursos para se desenvolver um IDE, a fim de, proporcionar facilidades e melhorias para o desenvolvimento de *software*.

Diante disso, foi possível identificar as facilidades que cada IDE possui, bem como suas características. Deste modo são apresentados requisitos funcionais e não funcionais de um IDE para PHP, como também uma arquitetura de *software* deste IDE.

4.1. Características desejáveis do IDE

Um IDE deve possuir uma API de extensão, pois, é através da API que o *plugin* é implementado; uma boa documentação, para que haja meios para viabilizar a construção de todas as funcionalidades; suporte a outros desenvolvedores; código-fonte aberto, para maior facilidade, no desenvolvimento de *plugins*; aceitação do ambiente, quanto maior o número de usuários do ambiente, maior será a aceitação do projeto. A localização e o tipo de suportes oferecidos para os IDE devem ser observados, o suporte pode ser oferecido tanto por comunidades de usuários no Brasil quanto por comunidades estrangeiras. Para facilitar o entendimento de algumas características dos IDE.

Em conformidade com o que se busca neste trabalho, um IDE/RAD que permita extensão e alta produtividade deve possuir alguns recursos que são indispensáveis, dentre eles:

- Gratuito e open source;
- Multiplataforma;
- Possuir interface (API);
- Suporte a outras tecnologias, como HTML. JavaScript, Flash, ActiveX e outras que forem necessárias;
- Suporte a linguagem PHP-GTK, para possibilitar criação de aplicações stand alone;
- Integração com banco de dados;
- Checagem de sintaxe em tempo real;
- Coloração de sintaxe;
- Auto complemento de código;
- Ajuda rápida ao selecionar palavra chave e mover mouse sobre a mesma;
- Um bom *debugger*;
- Organizador de código;
- *Browser* integrado;
- Uma documentação que ajude o desenvolvedor;

Após caracterizar o IDE, foi possível apresentar uma proposta de arquitetura de *software* básica para o desenvolvimento do IDE definido.

4.2. Proposta da Arquitetura de Software

O IDE definido deve ser desenvolvido sobre o paradigma de orientação a objetos, em uma plataforma como a do Eclipse, ou seja, possuir um *kernel*, que ofereça, através da API de extensão, possibilidade de criação e acoplamento de *plugins*, utilizando reuso de *software*, como *frameworks* e bibliotecas. Este IDE deve ser livre e *open source*, possibilitando o desenvolvimento de aplicações tanto para Web quanto para *stand alone*.
Considerando a plataforma Eclipse e o ambiente PHPEclipse, que já possui muitos recursos importantes, sugere-se o desenvolvimento de *plugins* e acoplamento dos mesmos a este ambinete. Como proposta, desenvolver os seguintes plugins:

- Plugin para construção de aplicações stand alone;
- Plugin para suporte a linguagem PHP-GTK;
- Plugin editor visual para interface.

Após definida a arquitetura básica do IDE e os requisitos funcionais e não funcionais, é possível escolher entre fazer aplicações para Web ou stand alone. Este IDE terá todas as funcionalidades presentes no PHPEclipse, como também as funcionalidades adicionais, observadas em outros ambientes.

5. CONCLUSÃO

Cabe, por fim, ressaltar que este artigo colaborou com o estudo de um IDE, com o intuito de propor melhores facilidades no desenvolvimento de *software*. Alguns IDE e editores existentes para linguagem PHP foram analisados, as avaliações foram satisfatórias e permitiram o estudo de técnicas de reusos de *software*. Neste contexto, foi escolhida a plataforma Eclipse, pelas suas vantagens, características e por oferecer facilidades de extensão, através de *plugins*. As metas alcançadas servirão de fundamento para definição de um IDE, onde será definido todas as atividades, passos e fases de desenvolvimento, além dos plugins de implementação que devem ser gerados. Cabe, salientar que este artigo já pode ser utilizado como referência, para desenvolvedores de *software*, a fim de desenvolver o ambiente aqui definido. Este trabalho contribuiu para o projeto “IDE4PHP”, não tendo nenhum custo financeiro, permitindo um estudo para que outros desenvolvedores possam refletir a partir desta pesquisa e dar continuidade à proposta do mesmo. Tem-se como sugestão, a aplicação da proposta apresentada, para adição de alguma nova funcionalidade ao *plugin* PHP para plataforma Eclipse e busca de uma nova plataforma de desenvolvimento de IDE que atenda todas as necessidades do desenvolvedor.

REFERÊNCIAS

BORGES, Karen; VIANA PRETO, Alexandre; Notas de Aula, paradigmas e linguagens de programação. Disponível em <http://www.thielke.com.br/alexandre/paradigma.htm>. Acesso em: 20 de Março de 2007.

CODE GEAR, 2007; Site oficial Disponível em < http://www.codegear.com/ /delphi/php>. Acesso em 28 de Março de 2007.

LIMA, Lucas Albertins, et al. Eclipse tools - ferramenta para auxílio à composição dinâmica de software. Campina Grande, 2005. Depto. de Sistemas e Computação DSC/UFCG. Disponível em <http://www.dsc.ufcg.edu. /~pet/Artigos/ _ECLIPSETOOLS. >. Acesso em 17 de Janeiro de 2007.

CONVERSE, Tim; PARK, Joyce. PHP a bíblia. 2. ed. Rio de Janeiro: Elsevier, 2003. 868 p.

DALL’OGLIO, Pablo; PHP-GTK, Criando aplicações gráficas com PHP. 1. ed. Rio de Janeiro: Novatec, 2004. 127p.

ECLIPSE, 2007; Site oficial. Disponível em <http://www.eclipse.org>. Último acesso em 15 Março de 2007.

ESTEVARENGO, L. F. Z.. Comunicação pessoal via software de comunicação MS Messenger em 02 de Junho de 2007.

GOODMAN, Danny. Java script a bíblia. 1. ed. Rio de Janeiro: Elsevier Editora Ltda, 2001. 909p.

MACROMEDIA, 2007; Site Oficial. Disponível em <http//www.macromedia/softwaresoftware/ dreamweaver>. Acesso em 30 de Janeiro de 2007.

PHPEDITOR, 2007. Site oficial. Disponível em <http://paginas.terra.com.br/informatica/php_editor/> em 25 de Janeiro de 2007.

PHP.NET, Inc. PHP. Hipertext Preprocessor. 1995–2007. Disponível em <http://www.php.net>. Acesso em: 25 de Janeiro de 2007.

SEBESTA, Robert W. Conceitos de linguagem de programação. 4.ed. Rio de Janeiro: Alta Books, 2000.

ZEND, 2007. Zend Studio. Disponível em http://www.zend.com/products/zend_studio?hpb=4 >. Acesso em 20 de Março de 2008.