



GOVERNO DO ESTADO DO RIO DE JANEIRO  
SECRETARIA DE ESTADO DE CIÊNCIA E TECNOLOGIA  
FUNDAÇÃO DE APOIO À ESCOLA TÉCNICA  
CENTRO DE EDUCAÇÃO TECNOLÓGICA DO ESTADO DO RIO DE  
JANEIRO  
FACULDADE DE EDUCAÇÃO TECNOLÓGICA DO ESTADO DO RIO  
DE JANEIRO  
FAETERJ/PETRÓPOLIS

Framework MVC - Lotus{PHP}

Guilherme Peixoto da Costa Louro

PETRÓPOLIS

Julho de 2015

# Framework MVC - Lotus{PHP}

Guilherme Peixoto da Costa Louro

Trabalho apresentado no curso de Formação em Tecnologia da Informação e Comunicação da FAETERJ – Petrópolis como requisito parcial para obtenção do grau de tecnólogo.

Orientador: Matheus Bandini

Co-orientador: Hélio José Corrêa Barbosa

PETRÓPOLIS  
Julho de 2015

Monografia de Projeto Final de Graduação sob o título “*Framework MVC - Lotus{PHP}*”, defendida por Guilherme Peixoto da Costa Louro e aprovada em Julho de 2015, em Petrópolis, Estado do Rio de Janeiro, pela banca examinadora constituída pelos professores:

---

Orientador

---

Co-orientador

---

Nome do membro da banca  
Intituição do Membro

---

Nome do membro da banca  
Intituição do Membro

# *Resumo*

Devido a forma complexa em que se encontra o desenvolvimento de sistemas web atualmente, torna-se cada vez mais importante o uso de ferramentas que facilitam a criação dos mesmos. Essas ferramentas que denominaremos de *Frameworks* são utilizadas visando o aumento de produtividade, esse aumento se deve as diversas ações que auxiliam nas principais atividades do desenvolvimento. Existe no mercado diversos *Frameworks* web que são opensource. Porém o foco desse trabalho será baseado na criação de um *Framework* próprio, onde foi escolhido um padrão de projeto, que será o modelo MVC, e uma linguagem de programação que será PHP. Serão apresentados cada passo dado para a criação do *Framework* além de explicações técnicas referentes a cada funcionalidade do sistema, a documentação completa para uso do *Framework* e algumas aplicações que já utilizam do *Framework*.

# *Abstract*

Nowadays the meta heuristics have been used for its simply of implementation and for be able to be applied in a very large range of problems with the most levels of complexity. In this work, we have study the technique named Particle Swarm Optimization (PSO). Which one, as occurs in the most of meta heuristics, the quality of application of this technique have to define some parameters that have direct influence in the performance of algorithm, like the inertia weight that prevent that the particle change its direction instatly, and the swarm learning ability and its own. Those parameters was evaluted by a group of tests largely used on literature and with the goal of observe how the algorithm behave in each group of parameters in diferents situations applied.

Furthermore, a larger study about the technique has been done evaluting algorithms that adapt their parameters during the execution, thus searching a tool more effective applied on diferents groups of problems. Algorithms that adapt the inertia weight, for instance, generally allow that the particle keep more free in the opening execution e became more restrective during the search.

Particles on its turn, do not depend olny of the inertia wight to evolve. Its learning is give by two parameters known as cognitive aceleration and social aceleration, which one, define how much the particle is influenced by the swarm and how much she must search the best by itself. (Thus, if implement more social, in other words, that give more importance to the swarm learn, it can lead to failure of all group in case it is lost.) So, it is clear that the definition of those parameters commits the final result of the technique.

Beyond those types of adaptation, still exist algorithms that updates in diferents ways the particle's velocity, according with what those particles have learn, as well as inclusion of new necessary parameters to control the adaptation.

There fore, thiw work aims to evalute various proposals around the definition of PSO parameters in a group of optimization problems without restriction, aiming identify proposals more adequate for diferents classes of problems.

# *Dedicatória*

Dedico este trabalho a sÃ£o

# *Agradecimentos*

Agradeço a ...



# *Sumário*

**Lista de Figuras**

**Lista de Tabelas**

<b>1</b>	<b>Introdução</b>	p. 10
<b>2</b>	<b>A importância de se usar um Framework</b>	p. 11
2.1	Vantagens em usar um Framework . . . . .	p. 11
2.2	Desvantagens em usar um Framework . . . . .	p. 12
<b>3</b>	<b>Experimentos de Frameworks</b>	p. 13
3.1	Cake PHP . . . . .	p. 13
3.1.1	Descricao da ferramenta . . . . .	p. 13
3.1.2	Objetivo . . . . .	p. 13
3.1.3	Características . . . . .	p. 13
3.1.4	Primeiros Passos . . . . .	p. 13
	<b>Referências</b>	p. 14

## *Lista de Figuras*

## *Lista de Tabelas*

# 1 *Introdução*

Devido à grande necessidade de entregar projetos de grande porte e com prazos consideravelmente baixos, foi percebida a necessidade de se encontrar soluções que facilitassem esse desenvolvimento.

A primeira atitude a ser tomada foi a criação de um arquivo que reunia diversas funcionalidades, afim de facilitar futuros projetos, onde processos que se repetiam diversas vezes eram colocados em funções que poderiam ser usadas em novos projetos.

Aplicações em geral precisam de um padrão mais significativo como forma estrutural de um projeto, deixando claro que a criação de um arquivo contendo todas as funções do projeto não era o melhor padrão a ser seguido. Este trabalho apresenta, como uma de suas justificativas, uma pesquisa profunda que reúne novos padrões para os processos e *Frameworks* web que poderiam ser mais úteis para um desenvolvimento ágil.

No final dessas pesquisas iniciais, alguns *Frameworks* foram testados e o CakePHP passou a ser usado como padrão. O CakePHP utiliza o padrão de projeto MVC (*Model, View, Controller*) que é um modelo de arquitetura de software que tem como objetivo básico separar a lógica de negócio da aplicação.

Os *Frameworks* são sempre muito robustos e com diversos tipos de funcionalidades, e com o CakePHP não é diferente. Com uma vasta documentação e uma quantidade considerável de arquivos em seu projeto mais simples, este passou a ser um problema ao invés de solução quando se busca um total domínio em uma aplicação.

Percebeu-se então a real necessidade de criar um Framework onde se tenha total controle de todas as funcionalidades, mantendo o padrão MVC, porém criando as próprias funcionalidades, mesmo que baseado em funcionalidades de outros *Frameworks*.

Identificar o problema foi o primeiro e principal passo para se iniciar o desenvolvimento do Framework, que se encontra sempre em evolução com novas implementações que resolvam determinados problemas.

## 2 *A importância de se usar um Framework*

Neste capítulo será apresentado, com dados técnicos, a importância do uso de um *Framework* em projetos de desenvolvimento, detalhando algumas de suas vantagens e desvantagens no processo de codificação.

O *framework* é, como princípio básico, uma arquitetura "padrão" que tem como objetivo fornecer ferramentas comuns a todo tipo de projeto, utilizando os mais variados tipos de Design Pattern (Padrões de Projeto) afim de proporcionar um ambiente de desenvolvimento extremamente produtivo.

Grande parte dos *frameworks* trabalham com um padrão principal denominado MVC (Model View Controller) que tem como base trabalhar com Modelo Lógico (Model), onde acontece toda a interação com a base de dados do projeto, Visualização (View), que é a parte responsável pela exibição de dados e o Controle (Controller), que é a regra de negócios do projeto, pode-se dizer que o Controller é responsável por fazendo toda a comunicação com o Model e tratar os dados para serem exibidos pela View, resumindo, o padrão MVC separa claramente o Design do Conteúdo e de sua Lógica.

### 2.1 Vantagens em usar um Framework

- **Padronização em projetos:** A grande vantagem de um *framework* é sua padronização no desenvolvimento. Por utilizar um conjunto já definido de Classes e Métodos, a necessidade em trabalhar conforme a ferramenta possibilita ajuda a garantir um aproveitamento maior de código projetos futuros.
- **Velocidade no desenvolvimento:** O fato de se fazer uso de módulos genéricos faz com que o *framework* fique responsável por controlar o uso de funcionalidades repetitivas fazendo com que o desenvolvedor se concentre totalmente na regra de negócios de cada projeto.

- **Qualidade:** *Frameworks* em geral são testados e atualizados a todo momento, tornando cada vez mais seguro e com melhores funcionalidades.
- **Re-uso de códigos:** A padronização de projetos torna capaz o re-uso de código sem dificuldades de adaptação.
- **Segurança:** Uma das vantagens mais importantes é segurança que o *framework* pode dar ao projeto.
- **Fácil manutenção:** A separação do *framework* utilizando padrões de projetos permite uma fácil manutenção em determinada ferramenta sem que afete outras.
- **Utilitários e Bibliotecas:** Classes e métodos embutidos no *framework* afim de solucionar o problema de repetição contínua de códigos.

## 2.2 Desvantagens em usar um Framework

Esses pontos não são necessariamente uma desvantagem, porém são os principais motivos pelo qual inibem o desenvolvedor de iniciar em um *framework*.

- **Performance e peso:** A grande quantidade de arquivo e a chamada de métodos e criação de objetos nem sempre necessário para determinados projetos tornam a aplicação pesada em alguns casos.
- **Curva de aprendizado:** Ao se trabalhar com códigos de terceiros existe uma curva de aprendizado elevada e que fica dependente de uma boa documentação para conseguir atingir um bom ritmo de trabalho.
- **Conhecimento técnico:** É necessário que se tenha conhecimento em OOP (Programação Orientada à Objeto), boas práticas de programação e entenda padrões de projetos para poder utilizar o *framework* da melhor forma.

## 3 *Experimentos de Frameworks*

### 3.1 Cake PHP

#### 3.1.1 Descrição da ferramenta

O CakePHP é um projeto de código aberto mantido por uma comunidade bastante ativa de desenvolvedores PHP. Possui uma estrutura extensível para desenvolvimento, manutenção e implantação de aplicativos. Utiliza o padrão de projeto MVC (*Model-View-Controller*) e ORM (*Object-relational mapping*) com os paradigmas das convenções sobre configurações.

#### 3.1.2 Objetivo

CakePHP tem como objetivo principal a simplificação do processo de desenvolvimento e construção de aplicações web, utilizando um núcleo onde organiza o banco de dados e alguns recursos que reduzem a codificação pelo desenvolvedor. Alguns desses recursos são a validação embutida, ACLs (*lista de controle de acesso*), segurança, manipulação de sessão e cache de Views e sanitização de dados.

#### 3.1.3 Características

#### 3.1.4 Primeiros Passos

## *Referências*

[Kennedy e Eberhart 1995]KENNEDY, J.; EBERHART, R. Particle swarm optimization.  
In: IEEE. *Neural Networks, 1995. Proceedings., IEEE International Conference on*. [S.l.],  
1995. v. 4, p. 1942–1948.