

**FUNDAÇÃO EDSON QUEIROZ**

**UNIVERSIDADE DE FORTALEZA – UNIFOR**

**CENTRO DE CIÊNCIAS TECNOLÓGICAS – CCT**

**CURSO ENGENHARIA DE COMPUTAÇÃO**

**Projeto de Pesquisa**

**TCC I**

Proposta de Arquitetura de Assembler Online

Judah Holanda Correia Lima 1010500-5

Fevereiro - 2015

Judah Holanda Correia Lima

Proposta de Arquitetura de Assembler Online

Projeto de pesquisa a ser apresentado e submetido à avaliação para elaboração de Trabalho de Conclusão de Curso – TCC do Curso de Engenharia de Computação do Centro de Ciências Tecnológicas da Universidade de Fortaleza.

Orientador:

Fortaleza – Ceará 2013

Sumário

Resumo 4

Introdução 5

Tema 7

Delimitação do Tema 7

Objeto 7

Problema 7

Hipótese 7

Objetivos 7

Objetivo Geral 7

Objetivos Específicos 7

Justificativa 8

Discussão Teórica 8

Metodologia 8

Provável Sumário 9

Cronograma 10

Bibliografia 10

# Resumo

O presente estudo explora o potencial de um Assembler multiplataforma online. Além de um sistema de armazenamento de códigos e documentação online este projeto tem como objetivo ser uma rede social de compartilhamento, venda e troca de códigos online, bem como o auxílio ao reuso de códigos, impedindo a criação de códigos clones. O projeto do sistema tem como tarefa principal facilitar a programação, leitura e compartilhamento de códigos, tendo um padrão de codificação que deve ser utilizado por seu usuários para que esse objetivo possa ser alcançado. Este trabalho é um estudo em progresso do potencial que esse site possa ter bem como sua versão inicial e simplificada. O objetivo final desse sistema é criar uma rede de compartilhamento de códigos comerciais e acadêmicos que facilite o uso de códigos de terceiros, tanto como possibilidade programar em qualquer lugar, pois é um serviço web. E evidentemente por sua linguagem ser desacoplada de arquitetura este sistema possibilita uma migração de plataforma simples.

# Introdução

Atualmente é possível perceber que softwares que era um dia programas estão se tornando serviços web. Isto possibilita o uso de diversas plataformas em qualquer lugar que possua conexão com a internet e facilita trabalhos colaborativos.

Para a criação de um projeto de software/firmware comercial é necessário a instalação de diversos programas, dentre eles IDE’s, programas de documentação, gerenciamento, controle de versão, entre outros. E cada um deles é direcionado para um sistema operacional específico e algumas vezes é necessário utilizar dois sistemas distintos, pois um ou mais programas não estão disponíveis para a plataforma utilizada no inicio do projeto, bem como o tempo gasto para instalação. Depois de toda esta maratona ainda existe a pesquisa de frameworks e códigos para serem reutilizados neste projeto, pois não existe um local que todos os códigos do mundo estão armazenados e de fácil acesso. Finalmente temos a integração destes códigos com o projeto que isso implica em fazer tanto a integração com a IDE como a plataforma utilizada e isso algumas vezes implica na modificação do código que deseja-se integrar ou encapsula-lo em outro que faca a compatibilidade entre os dois e isso implica em ler um código que geralmente não esta seguindo o mesmo padrão de codificação e documentação isto quando segue algum ou tem documentação.

Uma IDE online além de tornar desnecessário a instalação ela permite o uso em qualquer lugar que internet esteja disponível, independente de plataforma ou sistema operacional. E esse é um dos motivos pelos quais serviços Web tem sido tamanho sucesso e tem aumentado tanto ao longo dos anos, programas que antes precisavam ser instalados na maquina agora se tornam online, sendo necessário apenas um browser. Onde não é necessário se preocupar nem mesmo com atualização de software.

Integrando todas essas ferramentas e códigos em um mesmo serviço ira tornar a criação de projetos muito mais rápida e pratica além de aumentar significativamente o reuso de código.

Contudo, mesmo fazendo tudo isso cada família de processador tem sua própria linguagem de maquina e consequentemente um Assembly próprio o que dificulta muito a migração de uma plataforma para outra.

Este estudo propõe uma nova abordagem, criando um Assembly em um nível mais alto que possa se transformar no Assembly especifico de cada processador criando um nível a mais, onde será possível a partir do código genérico gerar um especifico e vice-e-versa.

Ainda assim existe um outro problema a ser resolvido, ao se aprender algo novo, normalmente o mais difícil de se assimilar é a terminologia. Então mesmo que a ideia fundamental seja simples, quando é descrita com palavras não familiares isso pode dificultar bastante um assunto (Assembly 101). Como o Assembly utiliza mnemônicos isso demanda um estudo e aprendizado de cada instrução o que torna a linguagem bem mais complexa, linguagem de alto nível como C/C++ e Java tem suas funções escrita de forma extensa diferentemente de uma abreviação/sigla como no caso dos mnemônicos, por isso usualmente se aprende primeiro linguagens que alto nível, pois elas estão mais próximas da linguagem humana. Então propõe-se que as instruções seja escritas de forma extensa, facilitando a leitura do código e o aprendizado da linguagem.

Como vários em países tem se um costume de se programar na língua oficial do pais este é outro problema que normalmente impede que pessoas que não falem a língua não consigam utilizar o código escrito em tal língua. Então para contornar este problema o sistema terá como regra a codificação em inglês, devido ser uma das línguas mais faladas no mundo, uma das mais fáceis de se aprender e ter se tornado o padrão na maioria dos projetos.

Mesmo com isso ainda é necessário delimitar um padrão para nomenclatura de variáveis e métodos, pois mesmo tendo uma linguagem padrão isso não garante que os códigos serão de fácil leitura. Então um padrão de nomenclatura também deve ser adotado. Bem como de chaveamento e padrões de projetos.

# Tema

Proposta de uma Arquitetura de Assembler Online.

## Delimitação do Tema

Utilizar um conjunto de instruções ainda mais reduzido que o RISC convencional que contenha todas as operações básicas que qualquer microprocessador possua, permitindo assim um conjunto de instruções multiplataforma.

Criar um conjunto de instruções fáceis de serem lidas.

Desenhar um banco de dados para que se possa ter um banco de códigos com controle de versão básico onde o programador não tenha que ficar se preocupando com programas externos para este fim.

# Objeto

## Problema

O processamento do montador é viável pra a linguagem PHP?

O Sistema tem uma abordagem eficiente?

Pode ser usando em sistemas embarcados, dispositivos móveis?

## Hipótese

A velocidade de processamento em linguagem PHP por um Servidor em nuvem é suficientemente rápida para o Montador proposto.

A arquitetura reduzida RISC proposta é suficientemente rápida para boa parte das aplicações atuais, de tal forma que a perda de performance devido a ausência de outras instruções é justificada pela sua flexibilidade.

Devido o aumento da capacidade de processamento estar tornando processadores para dispositivos tão poderosos que já se justifica utilizar Java esta arquitetura é viável para sistemas embarcados e dispositivos móveis.

# Objetivos

## Objetivo Geral

Potencializar o reuso de códigos.

## Objetivos Específicos

Criar conjunto de instruções, determinar padrão de nomenclatura a ser seguido pelo seus usuários, utilizar controle de versão e armazenamento no próprio sistema e criar sistema básico (Easembly).

# Justificativa

Mesmo já existindo linguagens multiplataforma em auto nível, não é possível utilizá-la em processadores com um poder de processamento mais modesto.

Mnemônicos dificultam a leitura e entendimento dos códigos.

Conjuntos de instruções diferentes para cada família de processadores dificulta a migração de um processador para outro.

A utilização de uma plataforma web permite o programador codificar em qualquer lugar que possua internet em uma infinidade de dispositivos sem a necessidade de instalar diversas ferramentas.

# Discussão Teórica

A ideia deste conjunto reduzido de instruções vem do sucesso da arquitetura RISC nos processadores modernos. Para tornar este conjunto de instruções multiarquitetura é necessário reduzir o numero de instruções ao máximo para que que se tenha certeza que este conjunto vai estar presente em qualquer processador.

O armazenamento de códigos, nada mais é do que banco de dados de Strings com metadados para gerenciamento dos mesmos.

# Metodologia

*Devido a principal produto deste estudo ser a arquitetura deste sistema e não o sistema/software em si, será adotada uma metodologia ágil para este projeto.*

# Provável Sumário

INTRODUÇÃO

Estrutura da monografia

1. Arquitetura do sistema
   1. Geral
   2. Cliente
   3. Servidor
   4. Banco de Dados

2. Conjunto de Instruções

2.1. Conjunto de Instruções do processador #1

2.2. Conjunto de Instruções do processador #2

2.3. Conjunto de Instruções comuns a ambos processadores

2.3.1. Conjunto de Instruções básicas

2.4. Tipo de Instruções

2.5. Exemplos de Execução

3. Armazenamento

3.1. Banco de Dados

3.2. Controle de Versão

3.3. Informações de conta de usuário e permissões

4.Arquitetura do website Easembly

4.1.

CONCLUSÃO

REFERÊNCIAS BIBLIOGRÁFICAS

# Cronograma

Etapa I: revisão da bibliografia.

Etapa II: desenvolvimento do protótipo

Etapa III: análise do material coletado.

Etapa IV: redação do trabalho.

Etapa V: revisão e apresentação do trabalho final.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Jan/14 | Fev/14 | Mar/14 | Abr/14 | Mai/14 |
| Etapa I |  |  |  |  |  |
| Etapa II |  |  |  |  |  |
| Etapa III |  |  |  |  |  |
| Etapa IV |  |  |  |  |  |
| Etapa V |  |  |  |  |  |

# Bibliografia

[*Descrição de todo o material (livros, periódicos, sites, relatórios, etc) utilizado no trabalho.* ***Seguir as normas da ABNT***]

**10. DE ACORDO** (assinaturas dos professores que avaliaram o projeto)

Professor Orientador: Nome:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Ass: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Professor da mesma área: Nome:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Ass: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_