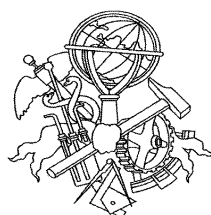


# CONTROLO DE UMA LINHA DE PRODUÇÃO

Ernesto Zhixiang Yongxin



Departamento de Engenharia Electrotécnica  
Instituto Superior de Engenharia do Porto

2010



Este relatório satisfaz, parcialmente, os requisitos que constam da Ficha de Disciplina de  
Seminário/Estágio, do 3º ano, da Licenciatura em Engenharia Electrotécnica e de  
Computadores

Candidato: Ernesto Zhixiang Yongxin, N° 1050756, 1050756@isep.ipp.pt

Orientação científica: Eng. Carlos José Ribeiro Campos, crc@isep.ipp.pt

Empresa: Indmei

Supervisão: Manuel Silva



Departamento de Engenharia Electrotécnica

Instituto Superior de Engenharia do Porto

3 de Novembro de 2010



## *Agradecimentos*

Agradeço ao meu orientador Eng. Carlos Campos por todo o seu apoio, colaboração e disponibilidade prestada ao longo deste projecto.

À minha família pelo todo o seu apoio, principalmente aos meus pais que me permitiram prosseguir os estudos apesar de todas as dificuldades.

Aos meus colegas pelo longo tempo passado juntos, pelo bom ambiente criado e pelo companheirismo nas horas difíceis.



## *Resumo*

O trabalho proposto pela empresa **Indmei**, consiste em projectar e implementar um sistema que proporcione um melhor controlo das diferentes linhas de produção. Este sistema deverá proporcionar: a diminuição da necessidade de controlo de produção e supervisão, redução do tempo de paragem da linha, aumento do controlo de qualidade, tudo isto com o objectivo de melhorar o rendimento de produção. Os dados registados pelo sistema de controlo desenvolvido são guardados numa base de dados (MySQL). Esta base de dados permite, em tempo real fornecer uma saída de dados para o operador de linha e para o supervisor de produção, através de um interface desenvolvido em linguagem *HTML* e *PHP*. Sobre os dados recolhidos é possível efectuar qualquer análise estatística. A aplicação *phpMyAdmin*, utilizada no desenvolvimento deste trabalho, permite efectuar um conjunto de ajustes dos parâmetros de controlo, alteráveis pelo supervisor da linha de produção. Como exemplo desses parâmetros são: início e fim do turno, duração de cada período, valor de referência a partir do qual conta tempo de paragem, e definição do número mínimo de peças produzidas por turno. Com a implementação deste sistema, permitiu uma diminuição de tempo de paragem da linha de produção, diminuição da necessidade de supervisão presencial, permitindo uma melhor gestão nas diferentes linhas de produção e melhor gestão dos recursos da empresa.

## *Palavras-Chave*

Arduino, Gestão de dados, MySQL, HTML, PHP, Linguagem C.





## *Abstract*

The work proposed by the company Indmei is to design and implement a system that provides better control of different production lines. This system should provide: a decreased need for supervision and control of production, reduced line downtime, increased quality control, all with the aim of improving the production yield. The data recorded by the developed control system are stored in a database (MySQL). This database allows real-time data provide an outlet for the line operator and the production supervisor, through an interface developed in HTML and PHP. About the data collected can perform any statistical analysis. The phpMyAdmin application used in the development of this work, allows performing a set of control parameter settings, changeable by the supervisor of the production line. As an example of these parameters are: beginning and end of turn, duration of each period, a benchmark from which account downtime, and definition of the minimum number of parts produced per turn. By implementing this system, allowed a reduction in downtime of the production line, reducing the need to face supervision, allowing a better management in the different production lines and improved management of company resources.

### Keywords

Arduino, Data mining, MySQL, HTML, PHP, C Language.



# Índice

<b>AGRADECIMENTOS .....</b>	<b>I</b>
<b>RESUMO .....</b>	<b>III</b>
<b>ABSTRACT .....</b>	<b>V</b>
<b>ÍNDICE .....</b>	<b>VII</b>
<b>ÍNDICE DE FIGURAS .....</b>	<b>IX</b>
<b>ACRÓNIMOS.....</b>	<b>XI</b>
<b>1. INTRODUÇÃO .....</b>	<b>1</b>
1.1. CONTEXTUALIZAÇÃO .....	1
1.2. OBJECTIVOS.....	2
1.3. CALENDARIZAÇÃO .....	3
1.4. ORGANIZAÇÃO DO RELATÓRIO .....	3
<b>2. ARQUITECTURA DO SISTEMA.....</b>	<b>5</b>
2.1. <i>HARDWARE</i> ARDUINO.....	6
2.2. PC (SERVIDOR).....	8
2.2.1. Apache .....	8
2.2.2. MySQL.....	8
2.2.3. PHP .....	9
2.2.4. phpMyAdmin .....	11
2.2.5. HTML .....	12
<b>3. APLICAÇÕES DESENVOLVIDAS .....</b>	<b>13</b>
3.1. AQUISIÇÃO DE DADOS .....	13
3.2. PROCESSAMENTO DE DADOS.....	16
3.3. INTERFACE COM O UTILIZADOR .....	23
<b>4. ANÁLISE DE RESULTADOS .....</b>	<b>25</b>
<b>5. CONCLUSÕES E TRABALHO FUTURO.....</b>	<b>31</b>
<b>REFERÊNCIAS BIBLIOGRÁFICAS.....</b>	<b>33</b>
<b>ANEXO A. ESQUEMA DO <i>HARDWARE</i> ARDUINO .....</b>	<b>37</b>

<b>ANEXO B. INSTALAÇÃO DO SERVIDOR APACHE, INTREPETADOR PHP, SISTEMA DE GESTÃO DE BASES DE DADOS (MYSQL), NO SISTEMA OPERATIVO WINDOWS.....</b>	<b>38</b>
---	-----------

## *Índice de Figuras*

Figura 1	Calenderização .....	3
Figura 2	Esquema geral do sistema .....	6
Figura 3	Arduino .....	7
Figura 4	PHP .....	10
Figura 5	Fluxograma do programa no Arduino .....	14
Figura 6	Fluxograma da função “preencher” .....	19
Figura 7	Fluxograma da função “verificar” .....	21
Figura 8	Fluxograma da função “main” .....	22
Figura 9	Interface do <i>Output</i> .....	23
Figura 10	Fluxograma da visualização dos registos .....	24
Figura 11	Login .....	25
Figura 12	Tabela com as definições das outras tabelas .....	26
Figura 13	Tabela com as definições dos turnos.....	27
Figura 14	Tabela com as definições do controlo do posto.....	28
Figura 15	Tabela com os registos do posto.....	30



## *Acrónimos*

API	–	Application Programming Interface
CPU	–	Central Processing Unit
CSS	–	Cascading Style Sheet
FDTI	–	Future Technology Devices International
GPL	–	Gnu Public License
HTML	–	HyperText Markup Language
HTTP	–	HyperText Transfer Protocol
IDE	–	Integrated Development Environment
IMAP	–	Internet Message Access Protocol
IP	–	Internet Protocol
LDAP	–	Lightweight Directory Access Protocol
NNTP	–	Network News Transfer Protocol
PC	–	Personal Computer
PDF	–	Portable Document Format
PHP	–	PHP Hypertext Pre-processor
POP3	–	Post Office Protocol 3
POSIX	–	Portable Operating System Interface for Unix
RS232	–	Recommended Standard 232

- SNMP – Single Network Management Protocol
- SQL – Structured Query Language
- TCP – Transmission Control Protocol
- USB – Universal Serial Bus
- XHTML – eXtensible HyperText Markup Language







# 1. INTRODUÇÃO

Neste capítulo será feita uma contextualização do trabalho elaborado no âmbito do estágio curricular, onde serão apresentados os requisitos do sistema, apresentados os objectivos que se pretendem atingir, assim como a calendarização e a organização do relatório.

## 1.1. CONTEXTUALIZAÇÃO

Hoje em dia, para ter sucesso, um dos requisitos necessários é ter uma boa gestão, ou seja, saber coordenar os recursos humanos e os recursos de uma organização para definir e prosseguir os objectivos da empresa, sendo o controlo de produção, uma das vertentes da gestão com extrema importância.

A proposta de estágio apresentada pela empresa, incide fundamentalmente sobre a necessidade de aumentar a gestão do controlo de produção, nomeadamente quando o supervisor da linha se encontra ausente. A solução para este problema irá partir do desenvolvimento de sistema, hardware e software, que seja capaz de autonomamente gerar dados facilmente tratáveis das diferentes linhas de produção.

A **Indmei**, é uma empresa do ramo têxtil, cuja sua área de actuação se foca na produção e acabamento de meias. Nesta empresa existe diferentes linhas, desde a produção até ao

acabamento do produto, onde neste processo existe a necessidade de uma supervisão constante.

Existindo a necessidade de controlar as linhas de produção, para melhor poder gerir os recursos da empresa, surgiu a necessidade de implementar um sistema que facilitasse a gestão e controlo de produção. Perspectiva-se que na realização deste trabalho, se irá ter a oportunidade de trabalhar com sistemas de gestão de base de dados, diferentes APIs, criação de aplicações *Web* e desenvolvimento de aplicações em linguagem C. Também trabalhar com linguagens de programação *PHP*, *HTML* e comandos de *shell MySQL*, necessários para o desenvolvimento deste projecto. Em qualquer um dos aspectos científicos referidos anteriormente, vai existir a necessidade de adquirir e aprofundar conhecimentos nas respectivas áreas.

Este projecto engloba o controlo de seis postos de trabalho, nos quais é necessário, por exemplo, controlar o número de peças produzidas por bloco de tempo, tipicamente de um turno de oito horas, sendo cada bloco composto por vários períodos, tipicamente de uma hora. Controlar o tempo de paragem da linha de produção, na ordem dos minutos, de forma a otimizar os recursos. Também será desenvolvido um interface apelativo para os funcionários terem acesso aos parâmetros relativos à produção actual em tempo real, como por exemplo: o número de peças produzidas e o tempo de paragem por período e por turno.

## **1.2. OBJECTIVOS**

O principal objectivo proposto pela empresa foi o desenvolvimento de um sistema de controlo de produção dos seis postos de trabalho. Este sistema englobará o desenvolvimento ou aquisição de hardware, e o desenvolvimento de diversas aplicações que permita a recolha e tratamento de dados. Estas aplicações serão desenvolvidas recorrendo a diferentes linguagens.

Será necessário desenvolver um sistema de aquisição de dados das linhas de produção, para isso será necessário utilizar um sistema de aquisição de dados (*hardware*).

A nível de funcionalidade (aplicações) a desenvolver, estas deverão permitir gerir:

- Número de peças por período (tipicamente de uma hora)
- Total de peças por bloco (turnos de oito horas)

- Hora do início do período
- Tempo de paragem por período (da ordem dos minutos)
- Hora da última peça produzida

Um dos principais objectivos que se pretende atingir, consiste no desenvolvimento de um sistema completo que responda aos requisitos enumerados pela especificação em que resulte um sistema de baixo custo e de implementação fácil na unidade fabril.

### 1.3. CALENDARIZAÇÃO

	1 mês	1 mês	15 dias	15 dias
Estudo da arquitectura do sistema				
Implementação				
Análise de resultados				
Conclusão do relatório				

**Figura 1 Calendarização**

### 1.4. ORGANIZAÇÃO DO RELATÓRIO

No capítulo 1, é feita uma introdução e contextualização do trabalho a desenvolver a apresentado neste relatório. São apresentados os objectivos que se pretendem atingir assim como o escalonamento temporal da realização deste projecto.

No capítulo 2, são abordados as ferramentas utilizados no trabalho.

No 3º capítulo é apresentado o desenvolvimento do trabalho.

No capítulo 4, é mostrada os resultados do trabalho.

No 5º e último capítulo, são reunidas as principais conclusões e perspectivados futuros desenvolvimentos.

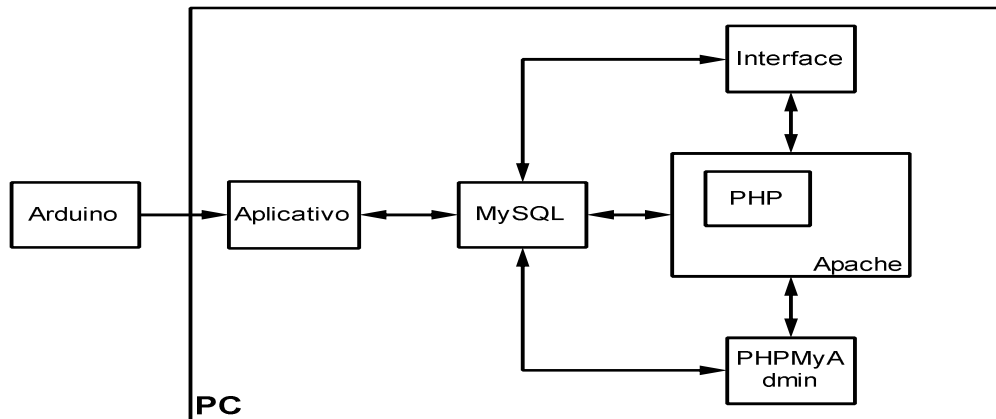


## 2. ARQUITECTURA DO SISTEMA

Neste capítulo, será descrito o *hardware* utilizado e as várias aplicações desenvolvidas referindo as suas principais características. Serão referidos os critérios de escolha das ferramentas de desenvolvimento, que recaíram essencialmente sobre facto de serem gratuitas (*open-source*), e apresentarem bastante robustez. Este aspecto permite baixar o custo de desenvolvimento, mantendo assegurada a manutenção do sistema.

Para aquisição de dados, optou-se por utilizar um sistema Arduino, embora não fosse tão interessante a nível académico, do que desenvolver uma própria placa para aquisição de dados. Tendo em conta o preço acessível, facilidade de aquisição e caso ocorram problemas não haver dependência de terceiros. Entende-se que é o ideal para a empresa em questão.

Na figura seguinte é apresentado um diagrama, onde se pode ver a arquitectura geral do sistema a implementar. Este diagrama foi desenhado na fase de planeamento e concessão do projecto. Sendo mais tarde validado na fase de implementação e na fase análise de resultados.



**Figura 2 Esquema geral do sistema**

O Arduino faz aquisição de dados, que são posteriormente enviados para o *pc*. No *pc* o Aplicativo recebe os dados do Arduino, lê os dados do controlo do MySQL e faz a actualização dos registos de informação. No servidor Apache temos um interpretador PHP, que torna possível a utilização de um *interface* com o utilizador e permite o acesso à base de dados com phpMyAdmin. Estas operações tornam-se possíveis graças a ambas utilizarem API PHP do MySQL.

## 2.1. **HARDWARE ARDUINO**

O *hardware* Arduino,[2] é uma plataforma *open-source* de computação física baseada numa simples placa de interface física (*input/output*). Este hardware tem um IDE próprio, também *open-source*, disponibilizado no site [www.arduino.cc](http://www.arduino.cc).

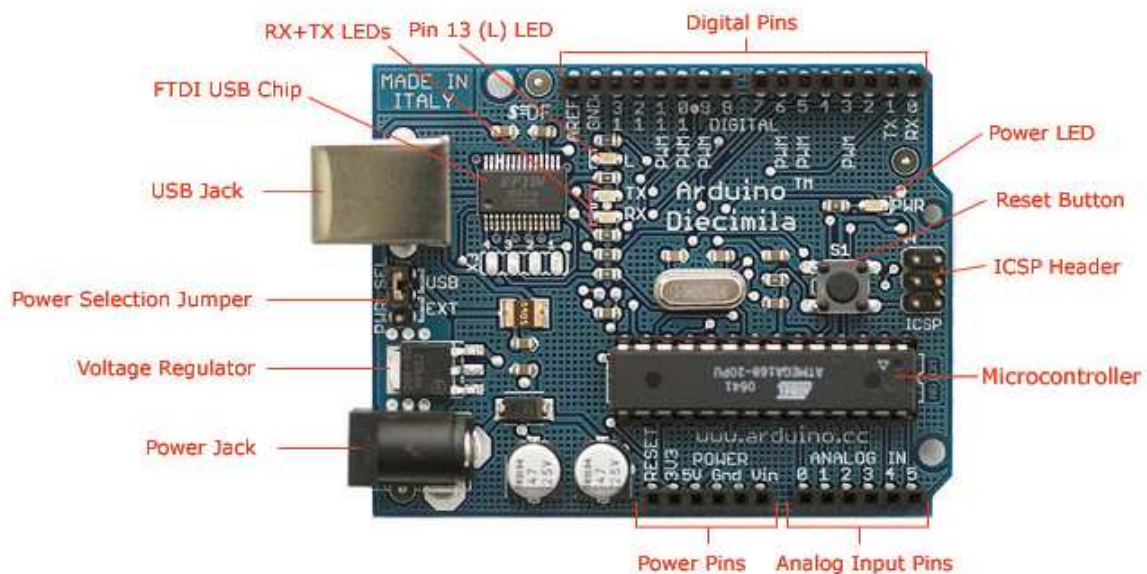
Este *hardware* pode ser utilizado em numerosas aplicações, por exemplo, para desenvolvimento de agentes interactivos autónomos, ou simplesmente ser utilizado como interface de aquisição de dados e envio para um sistema de gestão de dados (servidor).



Como se trata de *hardware open-source*, o esquemático da placa é disponibilizado, pode ser montado por qualquer pessoa sem ser necessário pagar a licença aos fabricantes, ou então pode ser obtido já montado. Para além disso, apresenta características que o tornam diferente de outras placas de desenvolvimento existentes no mercado:

- Tem um ambiente de desenvolvimento multi-plataforma (pode correr em Windows, Linux e Macintosh);
- Pode ser programado através da porta USB, ao invés da antiga porta série utilizada em outros sistemas, o que já não se encontra disponível em alguns computadores recentes;
- Pode ser obtido por um preço bastante acessível, e enquadrável nos custos de desenvolvimento (~20€).

Na figura seguinte, é apresentada uma imagem do *hardware* Arduino.



**Figura 3** Arduino

Como se pode observar pela figura anterior, a placa tem 2 circuitos integrados principais: um micro controlador Atmega328, que é o núcleo da placa, e pode ser facilmente substituído por outro com menos capacidade para tarefas mais simples, que necessitem de menos recursos, ou em caso de avaria; e o FT232RL que faz a conversão do protocolo RS232 para USB. A porta USB disponível pode ser utilizada para comunicação ou

programação do micro controlador. Também, facilmente se pode ter acesso ao um conjunto de pinos, que poderão funcionar como entrada ou saída de dados.

## **2.2. PC (SERVIDOR)**

Será necessário como parte integrante do sistema, o recurso a um *pc (personal computer)*, onde estará instalado e configurado um servidor *Web*, uma base de dados *MySQL* e a aplicação desenvolvida que fará a aquisição de dados e introdução dos dados no servidor *MySQL* de acordo com a informação recebida do *Arduíno*.

No anexo B, são apresentados os passos de instalação de cada um dos *softwares* utilizados no desenvolvimento deste trabalho.

### **2.2.1. APACHE**

O servidor *Web* (Apache)[13], tem uma historia de desenvolvimento que remonta a meados de 1994.

Com o aparecimento das comunidades entusiastas de *open-source*, a evolução deu-se rapidamente, o que fez com que se tornasse no servidor *open-source* mais bem sucedido.

As suas funcionalidades são mantidas através de uma estrutura de módulos, tornando-o mais flexível e seguro, sendo possível activar ou desactivar os módulos conforme as necessidades, permitindo inclusive ao utilizador escrever os seus próprios módulos utilizando o API do software.

É disponibilizado para várias plataformas, como Windows, Novell Netware, OS/2 e diversos sistemas do padrão POSIX(Unix, Linux, FreeBSD, etc).

### **2.2.2. MySQL**

O *MySQL* é um sistema de gestão de base de dados, *open-source*, que devido à sua robustez, rápido desempenho, alta confiabilidade e de fácil uso, se tornou um dos programas mais populares e mais utilizados a nível de gestão de dados. O *MySQL* é utilizado por grandes e conceituadas empresas a nível mundial, como Yahoo, Google, Nokia, Youtube, etc.[7]

Apesar de recentemente uma outra grande empresa de base de dados, Oracle, ter adquirido a Sun Microsystems, e que esta por sua vez teria adquirido a empresa de *MySQL* a cerca de 2 anos atrás. O *MySQL* continua ser *open-source* e de dupla licença, ou seja, pode ser utilizado sob licença GPL – GNU General Public License ou sob licença comercial para que possa ser usada numa aplicação comercial.

As principais características que o tornam atractivo para além de ser *open-source* são devido a sua:

Portabilidade – escrito em C e C++, testado com vários compiladores diferentes, funciona em várias plataformas, utiliza GNU *Automake*, *Autoconf*, e *Libtool* para portabilidade; suporta *multi-threads* para funcionar com vários CPUs; tem APIs para C, C++, Eiffel, Java, Perl, PHP, Python, Ruby e Tcl; fornece mecanismos de armazenamento transaccional e não transaccional; sistema de alocação de memória muito rápido e baseado em thread; *joins* muito rápidas usando uma *multi-join* de leitura única optimizada, etc.

Segurança – um sistemas de privilégios e senhas que é muito flexível, seguro e que permite verificação baseada em estações/máquinas. Senhas são seguras porque todo o tráfico de senhas é criptografado quando se conecta ao servidor.

Escalabilidade e limites – suporta enorme base de dados, suporta até 64 índices por tabela, cada índice pode ser composto de 1 até 16 colunas ou partes de colunas, o tamanho máximo do índice até 1000 *bytes*, tamanho máximo da tabela é tão grande que é apenas limitado pelo sistema operativo.

Conectividade – clientes pode conectar através dos *sockets* do TCP/IP a partir da qualquer plataforma, em sistemas UNIX pode se conectar através *sockets* do UNIX, podem ser construídos clientes a partir várias linguagens, utilizando os APIs

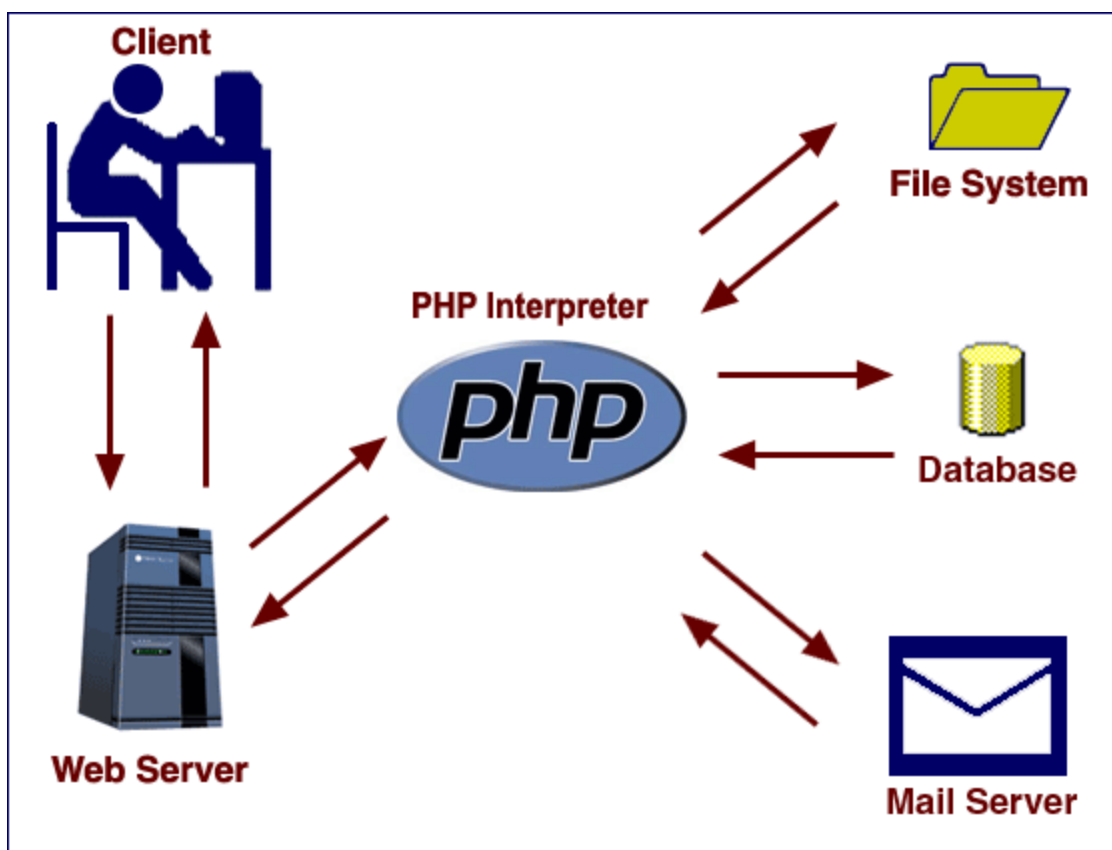
No capítulo 4, análise de resultados, é apresentada a estrutura da base de dados e é feita uma descrição mais detalhada da funcionalidade de cada tabela.

### **2.2.3. PHP**

O *PHP* é uma linguagem que apenas requer um *PHP parser*, i.e. um interpretador, não necessitando de ser compilado.[11] Os seus scripts são executados do lado do servidor para gerar páginas dinâmicas, mas não estão limitados apenas à criação de código *HTML*,

podendo gerar imagens, ficheiros no formato *PDF*, animações em Flash, e etc. Uma das características mais interessantes e poderosas é sua capacidade de trabalhar com uma grande variedade de sistemas de gestão de bases de dados, incluindo o *MySQL*, já referido anteriormente e usado neste trabalho.

Em seguida apresenta-se uma imagem, onde se pode ver a interacção dos diferentes sistemas descritos anteriormente.



**Figura 4 PHP**

Como se pode ver pela figura, existe uma interacção entre o cliente e o servidor que por sua vez pode recorrer ao interpretador PHP para processamento de dados.

A linguagem de programação *PHP* é multi-plataforma, é suportada por vários servidores, e com suporte para comunicação com outros serviços utilizando protocolos tais como LDAP, IMAP, SNMP, NNTP, POP3, HTTP, entre outros. Tudo isto torna o *PHP* numa linguagem bastante poderosa e flexível, dando ao programador muita liberdade na escolha do sistema operativo e do servidor. Com todos estes recursos e outros não mencionados aqui, tornou-o numa linguagem muito popular na *Web*, especialmente na área de *e-commerce* (vendas on-line).

#### 2.2.4. PHPMYADMIN

Como dito no capítulo anterior, a grande arma do *PHP* é a possibilidade de trabalhar com base de dados. O projecto *phpMyAdmin* nasceu da ideia de criar uma interface *Web* para gestão MySQL utilizando *PHP*. Apesar de chamar *phpMyAdmin* e consistir num código PHP, também contém código de XHTML, CSS, Javascript, ou seja, todos de linguagem de programação para aplicativos *Web*, providenciando um interface completo para administrar bases de dados a partir da *Web*. [12] Sendo *open-source* desde inicio, tem contado com inúmeros programadores e tradutores ao longo da sua existência, para o seu desenvolvimento, tendo actualmente o suporte de mais de 50 línguas. Ainda que não seja perfeita, contém a maioria das funcionalidades básicas e alguns extras na manipulação de base de dados. Das funcionalidades básicas pode-se enumerar:

- Criação, remoção, renomeação e alteração dos atributos de base de dados
- Criação, remoção, renomeação, reprodução, reparação e optimização das tabelas
- Manutenção das estruturas da tabelas, incluindo os índices
- Inserção, remoção e mudança dos dados
- *Upload* de dados em binário
- Pesquisa de dados (tabelas ou base de dados)
- Importar dados
- Exportar dados e estruturas em vários formatos, com compressão
- Multi-utilizador e instalação multi-servidor com configuração baseado em *Web*

Funcionalidades de administração consiste em:

- Criação e atribuição de privilégios a utilizadores
- Privilégio de verificação de base de dados
- Verificar a configuração do servidor

### 2.2.5. HTML

Foi inventado em 1990 por um cientista chamado Tim Berners-Lee. Com intuito de interligar computadores do laboratório e outras instituições de pesquisa para facilitar a troca de documentos científicos, de forma universal e facilmente legível entre cientistas e investigadores. Parágrafos, listas, *headers*, títulos (os elementos principais do *HTML*) eram ideais para este tipo de documentação.[10]

Ao inventar o *HTML* ele lançou as fundações da Internet tal como a conhecemos actualmente. O Tim Berners-Lee nunca imaginou que tivesse esse sucesso quando o desenvolveu.

O problema foi que entretanto, a *Web* ficou literalmente cheia de sites feitos com essas "criatividades" em *HTML*, que o puxaram para uma finalidade que não a original. Para acomodar os mais variados pedidos, as *tags* de apresentação (cor, fonte e alinhamento) foram usadas e abusadas, quando o principal propósito da linguagem era estruturar informação.

Neste momento está-se a gerar consenso para a necessidade de voltar um pouco atrás, preparando ao mesmo tempo o futuro. Um exemplo é a separação do conteúdo com a apresentação do documento. Usando *XHTML* para o conteúdo e deixando a apresentação do documento a cargo de *Cascading Style Sheets (CSS)*.

Esta linguagem (*XHTML*) foi desenvolvida e aprovada com a recomendação do *World Wide Web Consortium (W3C)* em 2000, e a sucessora do *HTML 4.0*. O *XHTML*, não é nada mais que *HTML* estruturado em *XML (eXtensible Markup Language)*. Em suma, hoje em dia temos um *HTML* flexível, portátil e com um formato prático para dar forma aos documentos de Internet que pode ser combinado com outras linguagens como *Javascript*, *Flash* e *Java* tornando o documento muito mais interactivo.

### **3. APLICAÇÕES DESENVOLVIDAS**

A realização deste projecto foi escalonada em três fases. Numa primeira fase foi desenvolvida uma aplicação para o *hardware Arduino*, para fazer a aquisição e o envio de dados para o PC. Na segunda fase, no PC, foi desenvolvido o software de ligação à porta USB, recepção de dados, processamento, e gravação na base de dados (MySQL). Na terceira e última fase foi criado um interface para visualização em tempo real dos dados relativos às diferentes linhas de produção.

A última tarefa deste projecto, será a implementação deste sistema na empresa, que será realizada após a realização de todos os teste e apresentação à comunidade científica deste projecto.

#### **3.1. AQUISIÇÃO DE DADOS**

Inicialmente foi colocada a hipótese de desenvolvimento que consistia em utilizar o *hardware Arduino* para a aquisição de dados e um módulo Ethernet adicional para o envio de dados para o PC. No entanto, e tendo em mente a redução de custos de implementação, optou-se por usar a ligação USB disponível no *hardware Arduino* para comunicação com o PC.

Na aquisição de dados utilizou-se o mecanismo de *polling* (verificação periódica do estado das entradas e saídas do *hardware*) para verificação do estado das entradas, em alternativa ao uso de interrupções externas disponíveis no hardware Arduino.

No processo de desenvolvimento, foram utilizadas 6 *switches* para simular a produção do produto dos 6 postos de trabalho. Nesta fase, surgiu uma dificuldade na leitura dos *switches*, ao aplicar o método de *polling*, o micro controlador fazia várias leituras cada vez que era pressionado o *switch*. Para evitar este efeito, efectua-se a leitura, cria-se um atraso enquanto a entrada mantém o estado, enviando posteriormente os dados para o computador através da porta USB. No envio são utilizadas as funções implementadas no *hardware* Arduino. Em seguida apresentam-se dois fluxogramas, correspondentes ao software implementado no *hardware* Arduino.[1]

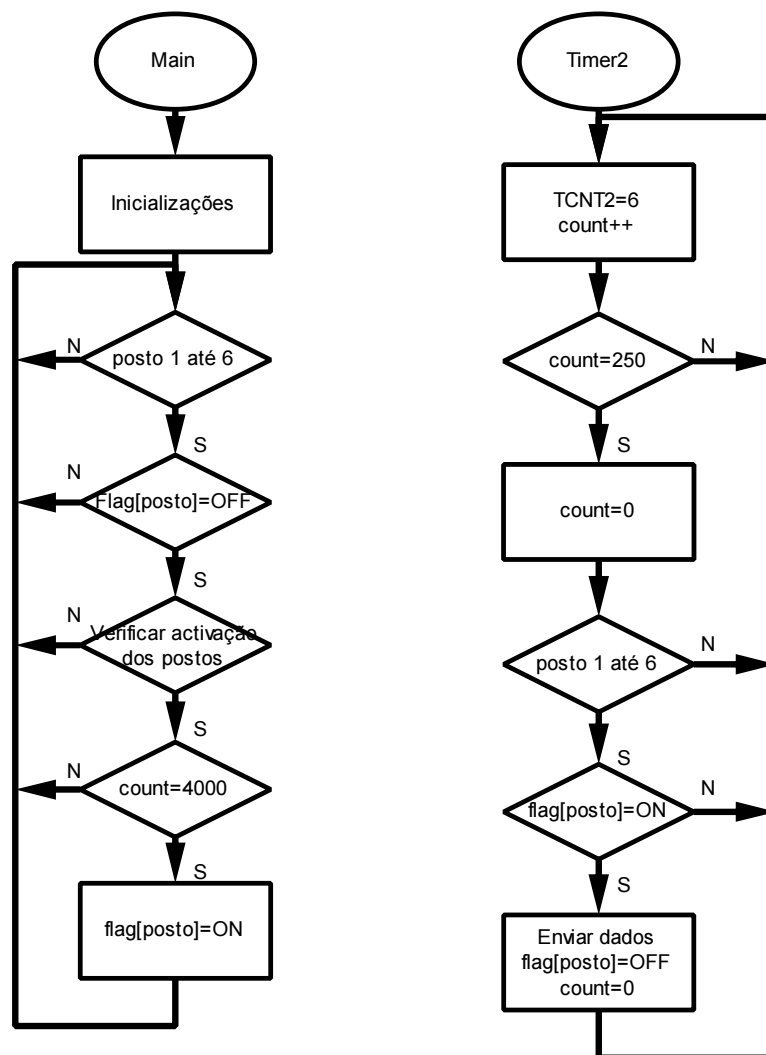


Figura 5 Fluxograma do programa no Arduino



Na figura 5 podemos observar os fluxogramas das funções “*main*” e “*timer2*”. Na função *main*, são feitas as inicializações, onde se activam os registos necessários para o *Timer2*. São declaradas um conjunto de variáveis e depois entra num ciclo (infinito) que faz a leitura das entradas. Foi associado um contador e uma *flag* para cada entrada (*switch*). A função *timer2* é chamada a cada 2 segundos e são feitas as verificações das *flags*. Caso alguma *flag* esteja activa, são enviados os dados para o PC. Após o envio, a *flag* é desactivada e o contador é reinicializado a zero.

Em seguida apresenta-se um extracto de código referente às duas funções descritas anteriormente.

#### **Função main**

```
for(j=0;j<=6;j++){
/*percorre 6 postos para verificar o estado */

    if(flag[j]==OFF && digitalRead(j+8)==HIGH){
//caso o flag do esteja a zero entrada ON

        duracao[j]++;
        if(duracao[j]==4000){
            flag[j]=ON;

        }

    }

}
```

#### **Função Timer 2**

```
if(count==125){                                //125 para 2s
    count=0;                                    //limpa o contador
    for(i=0;i<6;i++){
/*ciclo para verificar os postos com flag a 1*/

        if(flag[i]==ON){
/*mandar a info para pc e limpa a flag*/

            Serial.println(i+1);
            flag[i]=OFF;
            duracao[j]=0;

        }

    }

}
```

### 3.2. PROCESSAMENTO DE DADOS

Para o desenvolvimento da aplicação em C [8][9] que recebe os dados enviados pelo *hardware* Arduino, os vai processar e gravar na base de dados, recorreu-se ao IDE da Microsoft, Visual Studio 2005.

Nesta fase, foi necessário estudar o controlador FT232RL da FTDI,[3] Windows API[4] e C API do MySQL,[7] para compreender o seu funcionamento e conseguir conciliar as funcionalidade pretendidas.

Como foi referido anteriormente, a comunicação entre o *hardware* Arduino e o PC é feita através de uma ligação USB, o que facilita a compatibilidade de ambos, mesmo até em termos futuros, na utilização de outros sistemas operativos.

Para implementar esta funcionalidade o *hardware* Arduino possui um controlador da FTDI que faz a conversão de uma porta de comunicação serie para uma porta de comunicação USB. No caso do controlador da FTDI, o fabricante fornece o driver para aceder ao dispositivo USB como se fosse uma porta COM virtual.

Para a comunicação pela porta série é necessário definir: se a comunicação é feita de forma síncrona ou assíncrona; o número de bits usados para dados, o número de bits usados para o início e fim da trama; bem como se existe algum tipo de detecção de erros. Também é preciso definir a taxa transmissão pois esta terá que ser igual nos dois extremos da comunicação.

Como a aplicação a desenvolver é para correr no sistema operativo Windows e interagir com porta COM virtual, assim como, com o MySQL, foi necessário utilizar a API do Windows e do MySQL.[4][5][6]

Da API Windows foram utilizadas as funções para configurar os parâmetros da porta COM virtual e o modo como é feita a leitura dos dados.

Para obtenção e envio dos dados, recorreu-se à API MySQL em linguagem C.

Na aplicação desenvolvida encontram-se duas funções principais que gerem todo o processo, como a criação e actualização dos registos na base de dados.

Em seguida é apresentado um extracto do código que cria registo da função “verificar”.

```
nr_bloco= ptr.controlo->duracao_bloco;
id_bloco=ObterIdBloco(&ptr,hora_pc,nr_bloco);
duracao_t=duracao_turno(hora_pc);
nr_periodos=duracao_t/(d_controlo.duracao_periodo
*60);
resto[posto_id]=duracao_t%(d_controlo.duracao_per
iodo*60);

hora_fim_turno=mkttime(&d_turno[id_turno-
1].hora_fim);

for(f=0;f<nr_periodos;f++){

    limite_inferior=hora_inicio_turno+f*d_controlo.
duracao_periodo*60;

    limite_superior=hora_inicio_turno+(f+1)*d_contr
olo.duracao_periodo*60;

    (limite_inferior <= hora_pc &&
hora_pc<limite_superior)
?hora_tmp=limite_inferior : 0;
}

    (resto[posto_id]!=0 &&
(hora_fim_turno-resto[posto_id])<=hora_pc &&
hora_pc<hora_fim_turno )
?hora_tmp=hora_fim_turno-resto[posto_id] : 0;

    total=total_pecas(id_bloco,&ptr,hora_pc);

    ptrtm=localtime(&hora_tmp);
    sprintf(tmp,"%i:%i:%i",ptrtm->tm_hour,ptrtm-
>tm_min,ptrtm->tm_sec);

    sprintf(data,"%i-%i-%i",ptrtm-
>tm_year+1900,ptrtm->tm_mon+1,ptrtm->tm_mday);

    sprintf(query,"INSERT INTO %s
VALUES('%i','%i','0','%i','0','%s','00:00:00','%s
','%s')",d_tabelas[posto_id].data_info,id_turno,i
d_bloco,total,tmp,tmp,data);

WaitForSingleObject(mysql_mutex, INFINITE);
mysql_query(&mysql,query);
ReleaseMutex(mysql_mutex);
```

Em seguida é apresentado um extracto do código da actualização do registo na função “preencher”.

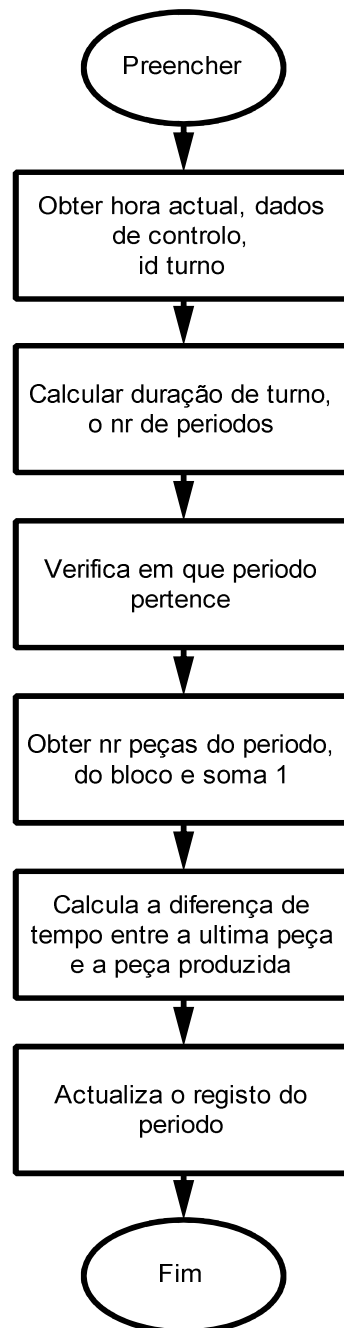
```
(H==0 && M==0 && S==0) // SE HORA DA PRIMEIRA
PEÇA FOR 00:00:00 ACTUALIZA ESSE CAMPO

? sprintf(query,
"UPDATE %s SET nr_pecas_perodo=%i,
total_pecas_bloco=%i, tempo_paragem_total=%i,
hora_primeira_pecas='%s',hora_ultima_pecas='%s'
WHERE hora_inicio_perodo='%s' AND data='%s'",
d_tabelas[posto_id].data_info,
nr_pecas_perodo,total_pecas_bloco,tempo_paragem_
total,hora_atual,hora_atual,inicio_perodo,
data)

: sprintf(query,
"UPDATE %s SET
nr_pecas_perodo=%i,total_pecas_bloco=%i,tempo_pa
ragem_total=%i,hora_ultima_pecas='%s' WHERE
hora_inicio_perodo='%s' AND data='%s'",
d_tabelas[posto_id].data_info,nr_pecas_perodo,to
tal_pecas_bloco,tempo_paragem_total,hora_atual,i
nicio_perodo, data);

WaitForSingleObject(mysql_mutex, INFINITE);
mysql_query(&mysql,query);
ReleaseMutex(mysql_mutex);
```

No fluxograma apresentado a seguir, podemos ver o funcionamento da função “preencher”.



**Figura 6 Fluxograma da função “preencher”**

Esta função só é executada quando recebe dados da porta USB, fazendo apenas actualizações na base de dados ao registo correspondente. A função “preencher” lê a hora

do computador e consulta a base de dados para determinar qual o registo onde deve escrever. É necessário saber, numa determinada altura, em que turno se encontra, para determinar em que período se está, obtendo os dados necessários para determinar e actualizar o respectivo registo. Os campos a serem actualizados são: número de peças do período, total de peças do bloco, tempo de paragem total e hora da última peça.

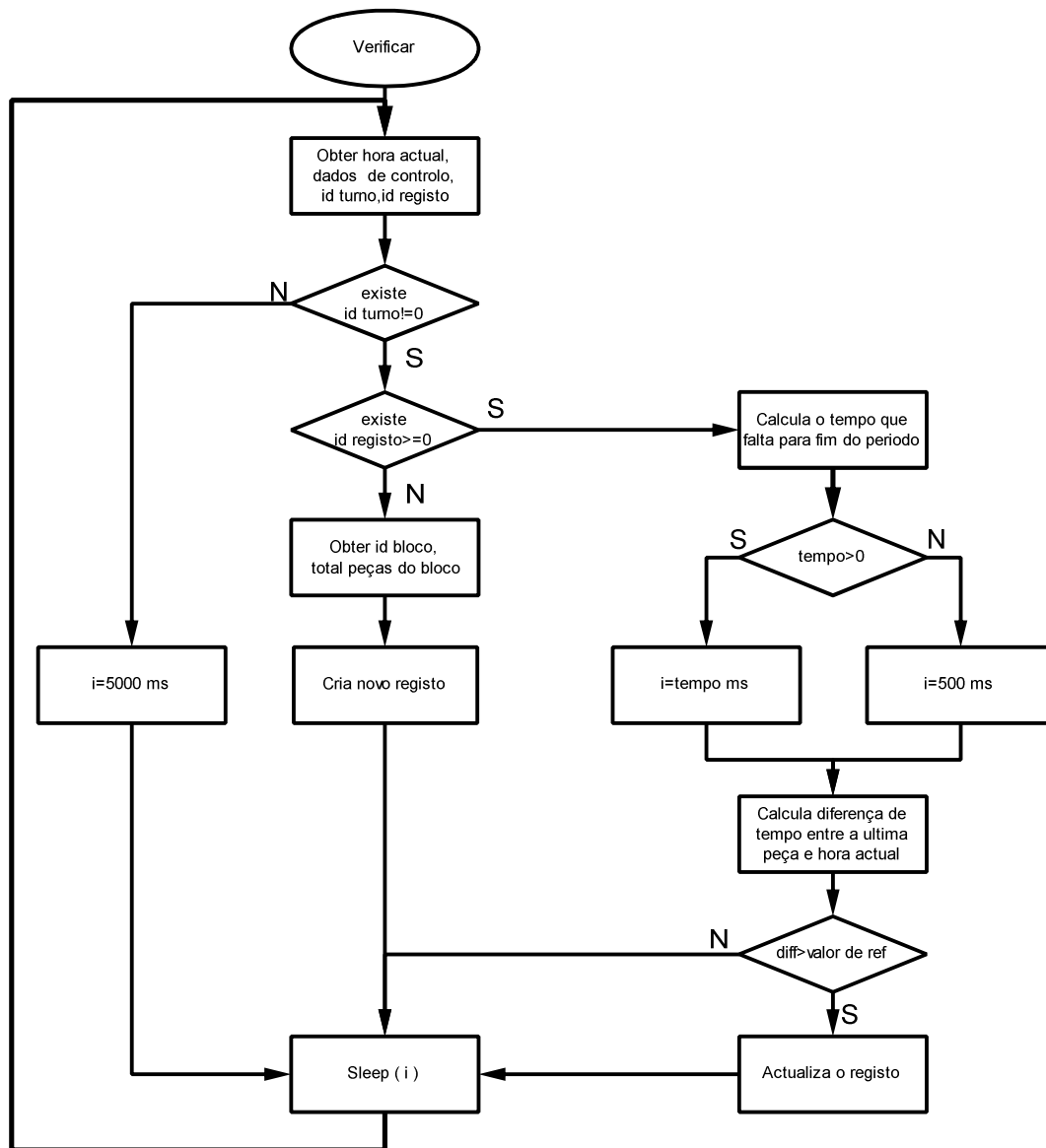
A função “verificar”, referente ao fluxograma da figura seguinte, é mais complexa do que a anterior. Esta função permite criar e actualiza registos.

De modo mais detalhado, obtém a hora do computador e informações da base de dados de acordo com essa hora. Após a obtenção dos dados, verifica se está dentro do intervalo do turno. Caso não esteja entra em modo *sleep*, que na pratica corresponde a uma verificação feita de 5 em 5 segundos, até que a hora actual pertença ao intervalo, por exemplo, temos 2 turnos, em que o primeiro turno está definido entre 06h00 a 14h00 e segundo turno entre 15h00 a 23h00, entre esses 2 turnos temos um intervalo das 14h00 a 15h00, se a hora actual for 14h30, então essa hora não pertence a nenhum dos turnos, o aplicativo faz uma verificação de 5 em 5 segundos até as 15h.

De seguida verifica a existência do registo, se não existir, é gerado um novo registo. Como a função se encontra num ciclo infinito, na próxima verificação, irá encontrar a existência desse registo, calculando então, quanto tempo resta até ao fim do período. Entra no modo *sleep* com a duração do tempo calculado, evitando assim consumo de ciclos do CPU. Ao fim desse tempo “acorda” e calcula o tempo dispendido entre a hora actual e a hora da última peça produzida, caso seja maior que o valor de referência, é então actualizado o campo do registo “tempo de paragem total”.

A diferença entre a função “preencher” e a função “verificar” reside no facto de uma só ser executada quando há uma produção de peça, ou seja, só é chamada quando recebe dados da porta USB (função preencher), e a outra fica em espera durante o tempo definido para produção das peças e depois vai automaticamente verificar se foi produzida e actualizar o registo.

Na figura seguinte é apresentado um fluxograma da função “verificar”.



**Figura 7 Fluxograma da função “verificar”**

Como podemos ver nos dois fluxogramas mostrados acima, antes de qualquer actualização, as funções adquirem dados das tabelas, e são actualizados os registos no instante em que se recebem os dados na porta USB, em vez de se guardar num *buffer* e actualizar posteriormente. Isto apresenta uma grande vantagem, uma vez que em caso de avaria, ou falha de energia, se evitem as perdas de dados.

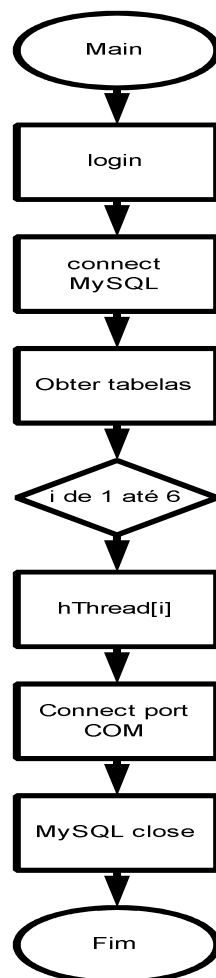
Na função “main”, apresentada no fluxograma seguinte, no início, nas inicializações são pedidos ao utilizador um conjunto de parâmetros, tais como: indicação da porta COM para

a recepção de dados, o servidor, o nome do utilizador e a respectiva palavra passe e o nome da base de dados à qual se pretende ligar para ligar ao MySQL. No capítulo 4 é apresentada uma explicação mais detalhada deste processo.

Feita a conexão, a aplicação vai obter os nomes das tabelas de controlo e um conjunto adicional de informação, consultando a tabela de definições. O supervisor pode alterar os nomes das tabelas, ou guardar dados em novas tabelas, sem necessitar de mudar o código fonte da aplicação.

São criadas 6 *threads* que executam a função “verificar”, um para cada posto de trabalho, a função “*main*” fica à espera de novos dados, executando a função “preencher” sempre que recebe novos dados.

Na figura seguinte apresenta-se um fluxograma da função principal “*main*”.



**Figura 8** Fluxograma da função “*main*”



### 3.3. INTERFACE COM O UTILIZADOR

No desenvolvimento do *interface* com o utilizador, utilizou-se a linguagem *HTML* para formatação da apresentação e a linguagem *PHP* para obter e mostrar os dados processados ao utilizador.

Na figura seguinte é apresentada uma imagem do interface com o utilizador.

interface1 - Mozilla Firefox

http://localhost/interface1.php

Links

File

Tools

Titulo

2010-11-2

POSTO 1

peças

tempo

0

0

5

1

2

1

TOTAL

7

5

POSTO 2

peças

tempo

0

0

4

3

3

1

TOTAL

7

12

POSTO 3

peças

tempo

0

0

4

2

3

1

TOTAL

7

12

POSTO 4

peças

tempo

0

0

6

4

3

2

TOTAL

9

12

POSTO 5

peças

tempo

0

0

6

9

3

2

TOTAL

9

12

POSTO 6

peças

tempo

0

0

9

4

TOTAL

9

12

Done

Js

J

F

SL

I

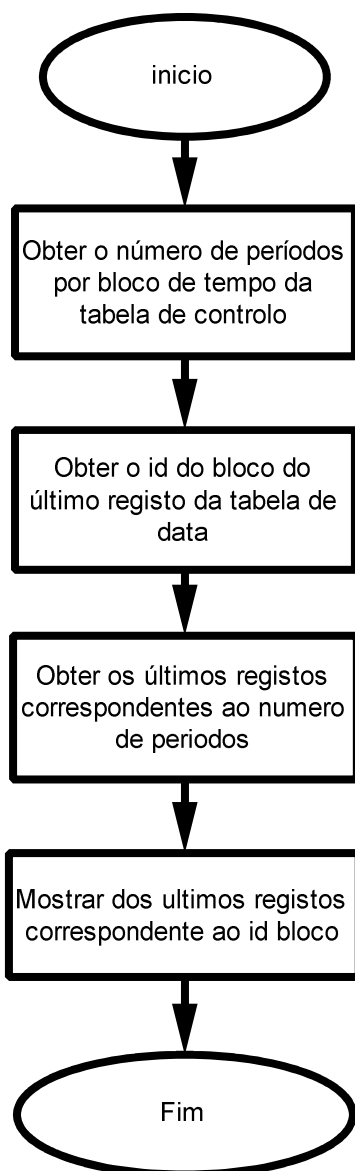
Css

zotero

Figura 9 Interface do *Output*

Como se pode ver na imagem anterior, temos 6 campos que contêm informações correspondentes a 6 postos de trabalho. Em cada campo: tem a descrição do posto, total das peças do bloco, referência do tempo de paragem permitido, para além disso mostra peças produzidas e tempo de paragem por período. A visualização do número de registos que contém informação do período é dinâmica, isto é, o supervisor pode definir o número períodos por cada bloco, turno, independentemente, e o *interface* ajustar-se-á.

O funcionamento da visualização é descrita no fluxograma seguinte.



**Figura 10 Fluxograma da visualização dos registos**

O refrescamento da informação apresentada ao utilizador é feito recorrendo a funções PHP. O intervalo de refrescamento pode ser parametrizado pelo utilizador.

## 4. ANÁLISE DE RESULTADOS

Na execução da aplicação, basta indicar os dados de login. Deve-se indicar o número da porta COM virtual, isso flexibiliza a aplicação, evitando a mudança no código fonte. A placa de aquisição de dados ligado num computador pode estar definido como porta COM4 e em outro computador pode já não ser a mesma.

Os restantes dados que devem ser indicados são referentes à conexão ao MySQL, o servidor onde está alojado o MySQL, utilizador, palavra-passe e a base de dados em que queremos trabalhar. Em seguida apresenta-se uma imagem do arranque da aplicação.

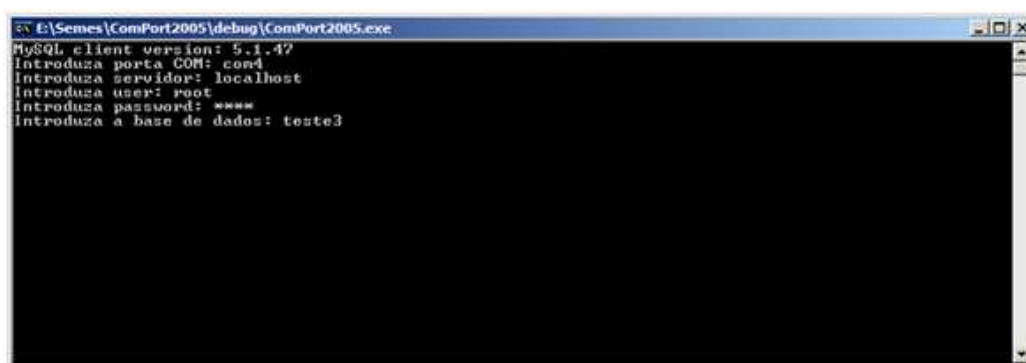


Figura 11 Login

Em seguida vamos analisar a estrutura da base de dados criada, recorrendo à aplicação phpMyAdmin.

A base de dados é constituída por 14 tabelas. Para cada posto contém uma tabela de controlo e tabela de informação; uma para definições de turnos dos postos e uma que contém os nomes das tabelas relativas a postos de trabalho, pode ver se de seguida uma pequena descrição dos campos das tabelas.

Para não alterar os nomes das tabelas no código fonte, foi criada outra tabela contendo os nomes das tabelas de controlo e data contidas na base de dados. Se este passo não fosse tomado, no caso de ser o utilizador a especificar os nomes das tabelas, aplicação corria o risco de não saber identificar correctamente a tabela pretendida.

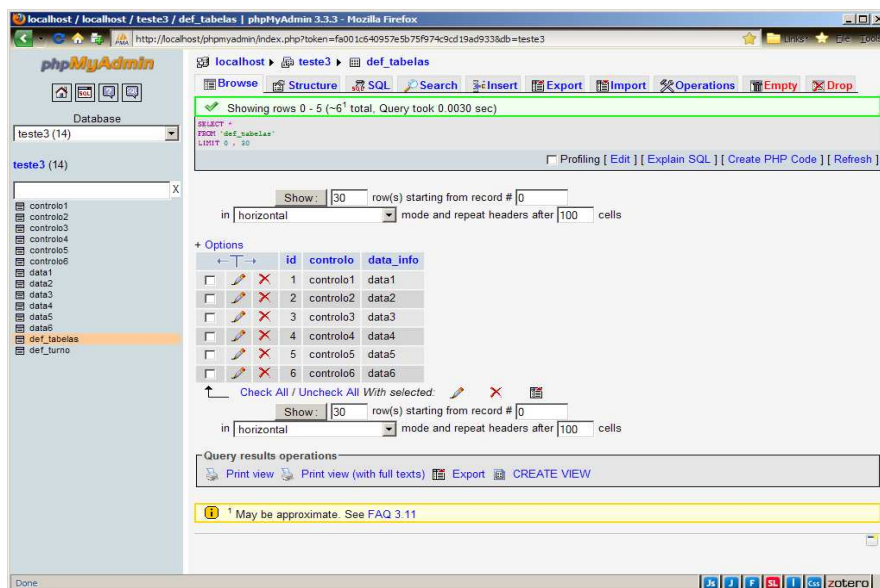
Na figura 12, vê-se a definição das tabelas de controlo e de dados. É nessa tabela que a aplicação desenvolvida vai ler os nomes das tabelas com que vai trabalhar, ou seja, os nomes das tabelas de controlo e de dados têm de estar de acordo com os nomes definidos na tabela def\_tabelas.

### Tabela de definições de tabelas (def\_tabelas):

Id – identificação posto

Controlo – contem o nome de tabela de controlo do posto

Data – contem o nome da tabela de informação do posto



**Figura 12** Tabela com as definições das outras tabelas

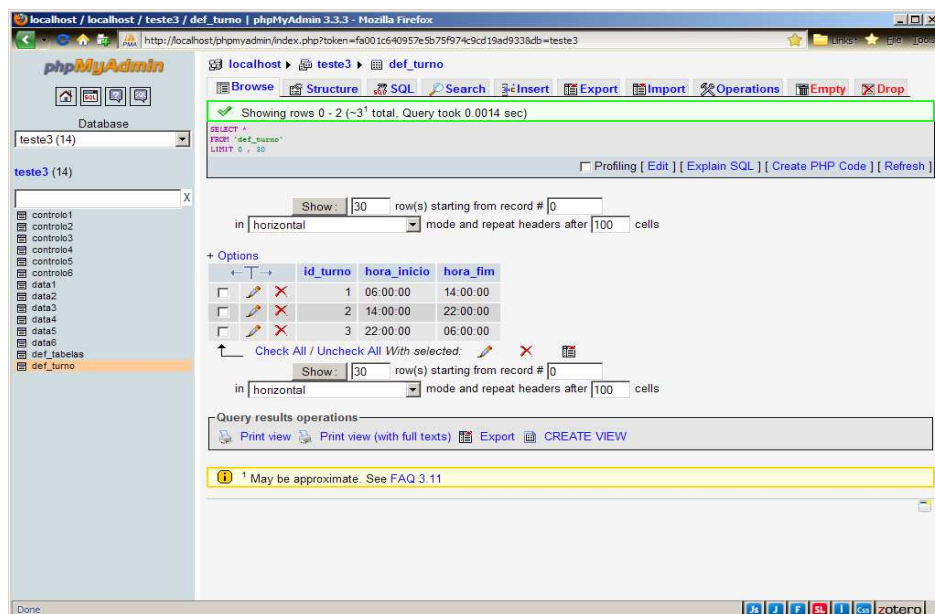
Na imagem seguinte vemos uma tabela com as definições do turno, é com os dados contidos nessa tabela que a aplicação verifica se a hora actual pertence a que turno, se não pertencer a aplicação não cria nem actualiza os registos.

### Tabela definições de turnos (def\_turnos):

Id turno – identificação do turno

Hora\_inicio – hora do inicio

Hora\_fim – hora do fim



The screenshot shows the phpMyAdmin interface for the 'teste3' database. The 'def\_turno' table is selected, and its contents are displayed. The table has three columns: 'id\_turno', 'hora\_inicio', and 'hora\_fim'. There are three rows of data representing different shifts.

id_turno	hora_inicio	hora_fim
1	06:00:00	14:00:00
2	14:00:00	22:00:00
3	22:00:00	06:00:00

**Figura 13 Tabela com as definições dos turnos**

Na figura 14 vê-se os dados de controlo de um dos postos, esses dados pode ser alterados pelo supervisor da produção.

## Tabela de controlo:

Descrição – contém uma pequena descrição do posto

Duração do período – duração do período em minutos

Duração do bloco – define o número de períodos por bloco

Ref\_cont\_tempo\_paragem – nesse campo é definido um intervalo de tempo (segundos) entre as peças produzida para qual o tempo começa a ser contabilizado

Ref\_tempo\_paragem – tempo (minutos) máximo permitido de paragem por período

Ref\_produção – número mínimo de peças que deve ser produzida por período

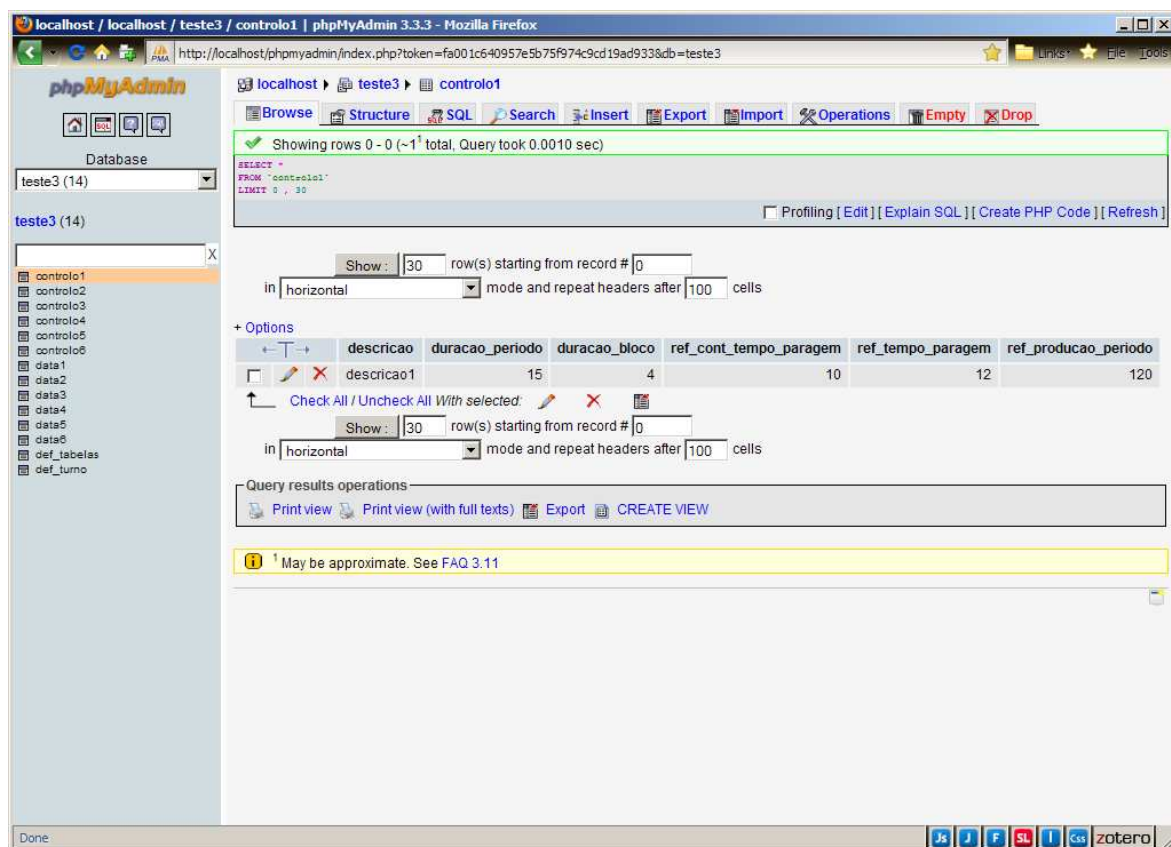


Figura 14 Tabela com as definições do controlo do posto

Na imagem seguinte apresenta-se uma tabela com as informações dos registos que foram introduzidos pela aplicação desenvolvida, essas informações são: id do turno, id do bloco, número de peças por período, total de peças por bloco, tempo de paragem total por período, hora do início do período, hora da primeira peça, hora da última peça e a data.

**Tabela de informações:**

Id turno – identificação do turno, serve para identificar o registo

Id bloco – identificação do bloco, serve para identificar o registo

Nr\_peças\_periodo – número de peças por período

Total\_peças\_bloco – total de peças dos períodos constituintes do bloco

Tempo\_paragem\_total – tempo de paragem (segundos) por período

Hora\_inicio\_periodo – regista o início do período, serve para identificar o registo

Hora\_primeira\_peça – hora da primeira peça

Hora\_ultima\_peça – hora da última peça, serve para cálculo de tempo de paragem entre a produção das peças

Data – a data do registo, serve para identificar o registo

Showing rows 0 - 29 (~74 total, Query took 0.0018 sec)

SELECT \* FROM `teste3`.`data1` LIMIT 0, 30

Page number: 1

	id_turno	id_bloco	nr_pecas_perodo	total_pecas_bloco	tempo_paragem_total	hora_inicio_perodo	hora_primeira_pecas	hora_ultima_pecas	data
<input type="checkbox"/>	2	1	0	0	889	21:45:00	00:00:00	21:59:59	2010-10-23
<input type="checkbox"/>	3	1	3	3	859	22:00:00	22:01:49	22:14:59	2010-10-23
<input type="checkbox"/>	3	1	1	4	879	22:15:00	22:28:47	22:29:59	2010-10-23
<input type="checkbox"/>	3	1	0	4	889	22:30:00	00:00:00	22:44:59	2010-10-23
<input type="checkbox"/>	3	1	1	5	879	22:45:00	22:47:07	22:59:59	2010-10-23
<input type="checkbox"/>	3	2	0	0	889	23:00:00	00:00:00	23:14:59	2010-10-23
<input type="checkbox"/>	3	2	0	0	889	23:15:00	00:00:00	23:29:59	2010-10-23
<input type="checkbox"/>	3	2	0	0	889	23:30:00	00:00:00	23:44:59	2010-10-23
<input type="checkbox"/>	3	2	0	0	889	23:45:00	00:00:00	23:59:59	2010-10-23
<input type="checkbox"/>	3	3	0	0	889	00:00:00	00:00:00	00:14:59	2010-10-24
<input type="checkbox"/>	3	3	0	0	889	00:15:00	00:00:00	00:29:59	2010-10-24
<input type="checkbox"/>	3	3	0	0	889	00:30:00	00:00:00	00:44:59	2010-10-24
<input type="checkbox"/>	3	3	0	0	889	00:45:00	00:00:00	00:59:59	2010-10-24
<input type="checkbox"/>	3	4	0	0	889	01:00:00	00:00:00	01:14:59	2010-10-24
<input type="checkbox"/>	3	4	0	0	889	01:15:00	00:00:00	01:29:59	2010-10-24
<input type="checkbox"/>	3	4	0	0	889	01:30:00	00:00:00	01:44:59	2010-10-24
<input type="checkbox"/>	3	4	0	0	889	01:45:00	00:00:00	01:59:59	2010-10-24
<input type="checkbox"/>	3	5	0	0	889	02:00:00	00:00:00	02:14:59	2010-10-24
<input type="checkbox"/>	3	5	0	0	889	02:15:00	00:00:00	02:29:59	2010-10-24
<input type="checkbox"/>	3	5	0	0	889	02:30:00	00:00:00	02:44:59	2010-10-24
<input type="checkbox"/>	3	5	0	0	889	02:45:00	00:00:00	02:59:59	2010-10-24
<input type="checkbox"/>	3	6	0	0	889	03:00:00	00:00:00	03:14:59	2010-10-24

Figura 15 Tabela com os registos do posto

Na imagem apresentada, pode-se observar a criação de vários registos, correspondentes a turnos diferentes, e em cada turno a períodos diferentes. Para cada registo tem-se acesso aos dados que foram descritos anteriormente.

A saída de dados é dado pelo *interface*, pode-se ver no capítulo 3.2 figura 9, desenvolvido em *HTML* e *PHP*.



## **5. CONCLUSÕES E TRABALHO FUTURO**

Com a conclusão deste trabalho, conseguimos obter um sistema de baixo custo, e que agrega todos os requisitos inicialmente propostos.

Com a implementação do sistema de controlo de produção em tempo real, irão obter-se vantagens a nível de gestão por parte da administração, motivação por parte dos funcionários e até poderá servir no controlo de qualidade.

Para administração, permite ter informações sobre a performance dos funcionários, fluxo do trabalho, assegurar o ritmo de trabalho dos operários, otimizar os recursos, um controlo mais detalhado e permitindo fazer uma análise estatística dos dados recolhidos

Poderá servir de motivação para funcionários mais ambiciosos, uma vez que têm a informação disponibilizada na hora, e para tentar manter ou melhorar o nível de trabalho.

Esse controlo poderá dissuadir os funcionários de terem uma produção desequilibrada, evitando que os funcionários estejam inactivos durante largos períodos de tempo e muito activos em outros períodos, diminuindo assim a probabilidade de produção de peças com defeito.

Na realização do trabalho, surgiram dificuldades nomeadamente com as APIs, no caso do Windows, a diferença de estilo de programação e a falta de conhecimentos em MySQL.

Estas dificuldades foram ultrapassadas durante a execução do trabalho, com o devido estudo das matérias.

Uma das características do aplicativo desenvolvido, é a sua capacidade de actualização em tempo real na base de dados, evitando assim as perdas de dados em caso de avaria ou falha de energia.

Em futuros desenvolvimentos poder-se-á implementar:

- Controlo de erro na comunicação entre a placa de aquisição de dados e computador.
- Obtenção dos parâmetros de configuração de um ficheiro de texto, como o número da porta COM, *server*, *user*, *password* e base de dados
- A criação automática de *backups* da base de dados
- Interface para alterar os parâmetros do controlo em vez de phpMyAdmin
- Efectuar testes reais
- Implementação do sistema na unidade fabril.

## ***Referências Bibliográficas***

- [1] EVANS, Brian W. – *Arduino Programming Notebook*. 1ª ed. Brian W. Evans Ed., 2007. [Consult. 18 de Outubro 2010]. Disponível em WWW:<url:[http://www.arduino.cc/playground/uploads/Main/arduino\\_notebook\\_v1-1.pdf](http://www.arduino.cc/playground/uploads/Main/arduino_notebook_v1-1.pdf)>
- [2] BANZI, Massimo – *Getting Started with Arduino*. 1ª ed. O'Reilly Media, Inc., 2008. ISBN: 978-0-596-15551-3
- [3] DATASHEET – *FT232R*, Versão 2.07, [Consult. 18 de Outubro 2010]. Disponível em WWW:<url:[http://www.ftdichip.com/Support/Documents/DataSheets/ICs/DS\\_FT232R.pdf](http://www.ftdichip.com/Support/Documents/DataSheets/ICs/DS_FT232R.pdf)>
- [4] BRAIN, Marshall e Ronald D. Reeves – *Win32 System Services: The Heart of Windows 98 and Windows 2000*, 3ª ed. Prentice Hall, 2000. ISBN: 0-13-022557-6
- [5] NEWMAN, Chris – *Sams Teach Yourself MySQL in 10 Minutes*, 1ª ed. Sams, 2006. ISBN: 978-0672328633
- [6] DUBOIS, Paul – *MySQL*, 4ª ed. Addison-Wesley Professional, 2008. ISBN: 978-0672329388
- [7] AXMARK, David e Michael Monty Widenius – *MySQL Reference Manual 5.5*, [Consult. 18 de Outubro 2010]. Disponível em WWW:<url:<http://http://downloads.mysql.com/docs/refman-5.5-en.html-chapter.zip>>
- [8] KERNIGHAN, Brian W. e Dennis M. Ritchie – *C Programming Language*, 2ª ed. Prentice Hall, 1988. ISBN: 978-0131103627
- [9] BANAHAN, Mike, Declan Brady e Mark Doran – *The C Book: Featuring the ANSI C Standard*, 2ª ed. Addison-Wesley Pub, 1991. ISBN: 978-0201544336. [Consult.

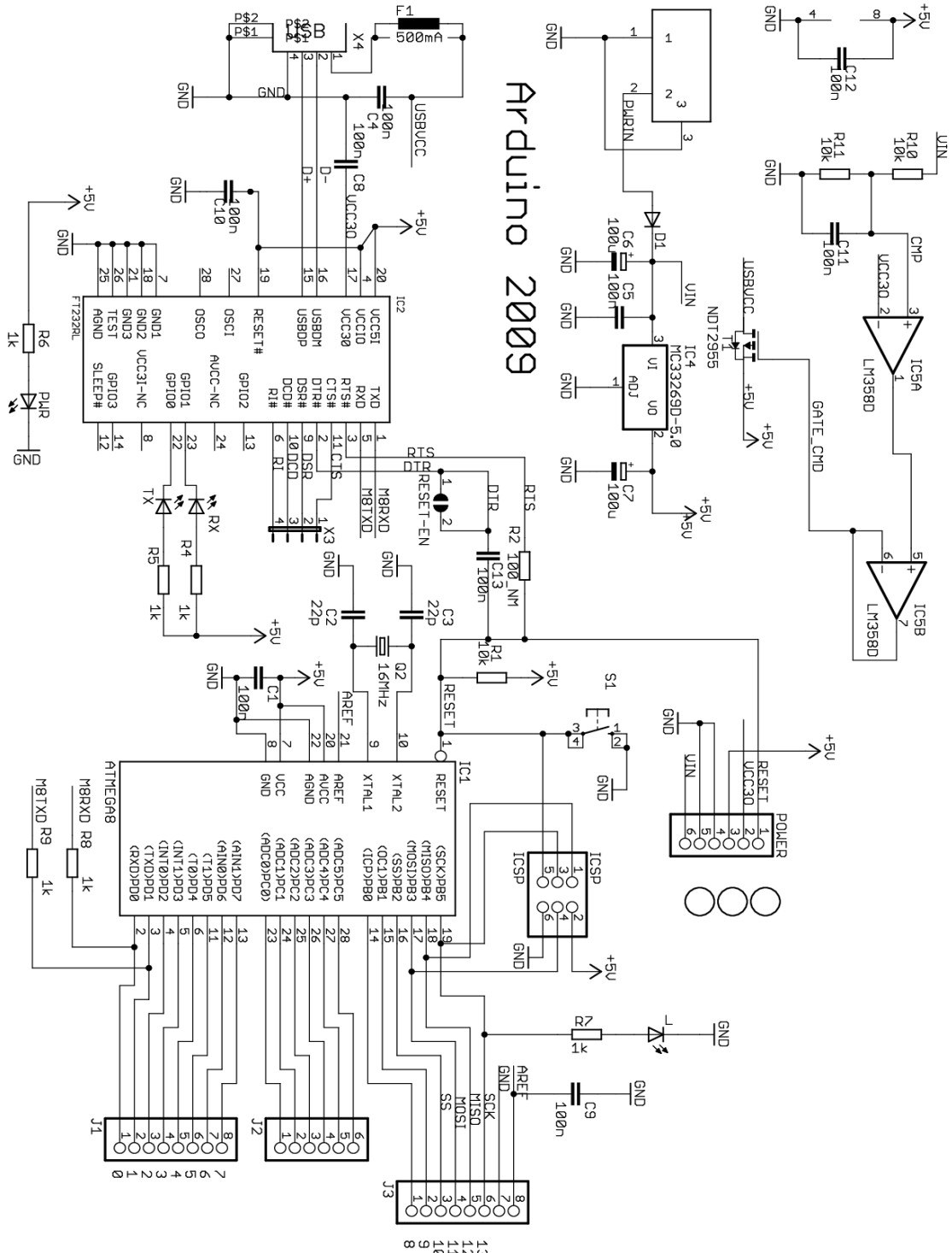
28 de Outubro 2010]. Disponível em  
WWW:<url:http://publications.gbdirect.co.uk/c\_book/>

- [10] W3C – *A history of HTML*, [Consult. 28 de Outubro de 2010]. Disponível em  
WWW:<url:http://www.w3.org/People/Raggett/book4/ch02.html>
  
- [11] ACHOUR, Mehdi e Friedhelm Betz – *PHP Manual*, [Consult. 28 de Outubro de 2010]. Disponível em  
WWW:<url:http://pt.php.net/get/php\_manual\_pt\_BR.chm/from/this/mirror>
  
- [12] DELISLE, Marc – *Mastering phpMyAdmin for Effective MySQL Management 2e*, 2<sup>a</sup> ed, Packt Publishing, 2006. ISBN: 978-1847191601
  
- [13] APACHE – *HTTP Server Project*, [Consult. 29 de Outubro de 2010]. Disponível em  
WWW:<url:http:// http://httpd.apache.org/ABOUT\_APACHE.html>





# Anexo A. Esquema do *Hardware* Arduino



## Anexo B. Instalação do Servidor Apache, Intrepetador PHP, sistema de Gestão de Bases de Dados (MySQL), no Sistema Operativo Windows.



Este anexo contém pequenas indicações na instalação do Apache, o PHP, o MySQL e phpMyAdmin no Windows sem utilização dos pacotes como WampServer, XAMPP. A vantagem de instalar tudo separado é poder configurar-se com mais liberdade, ao contrário dos pacotes mencionados anteriormente, que impõem certas limitações.

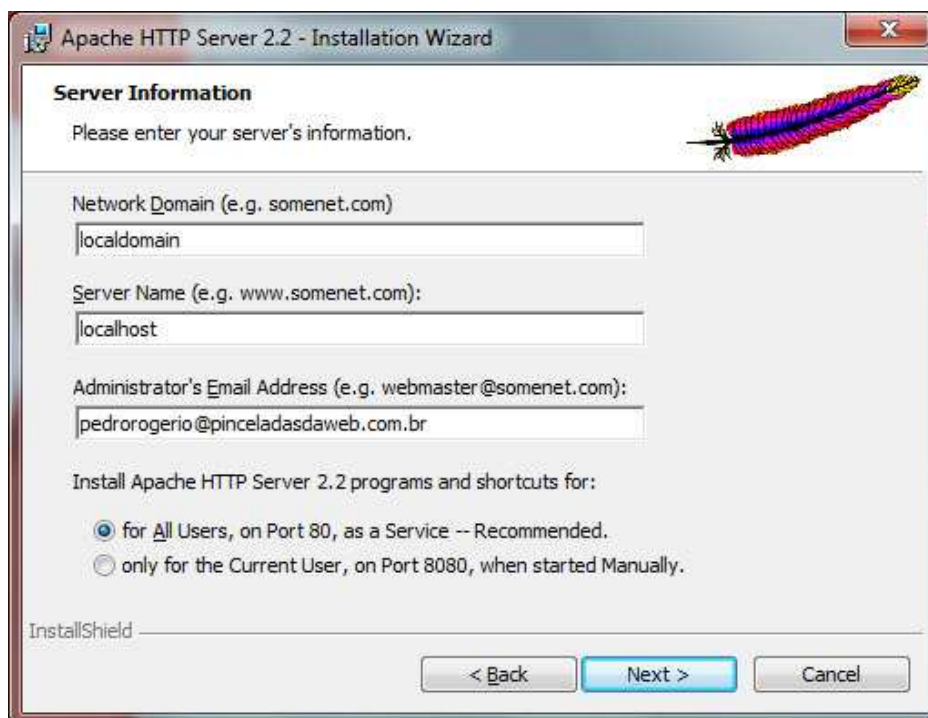


## 1: Instalação do Apache 2.2

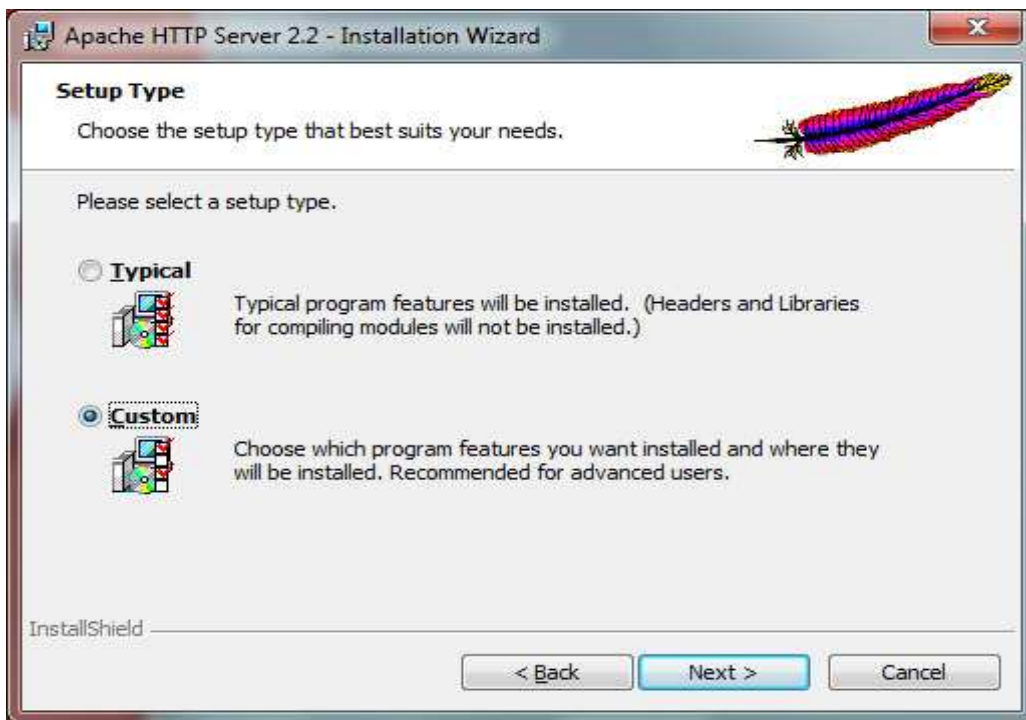
Executar o *setup* do Apache, e aparecerá as seguintes janelas:



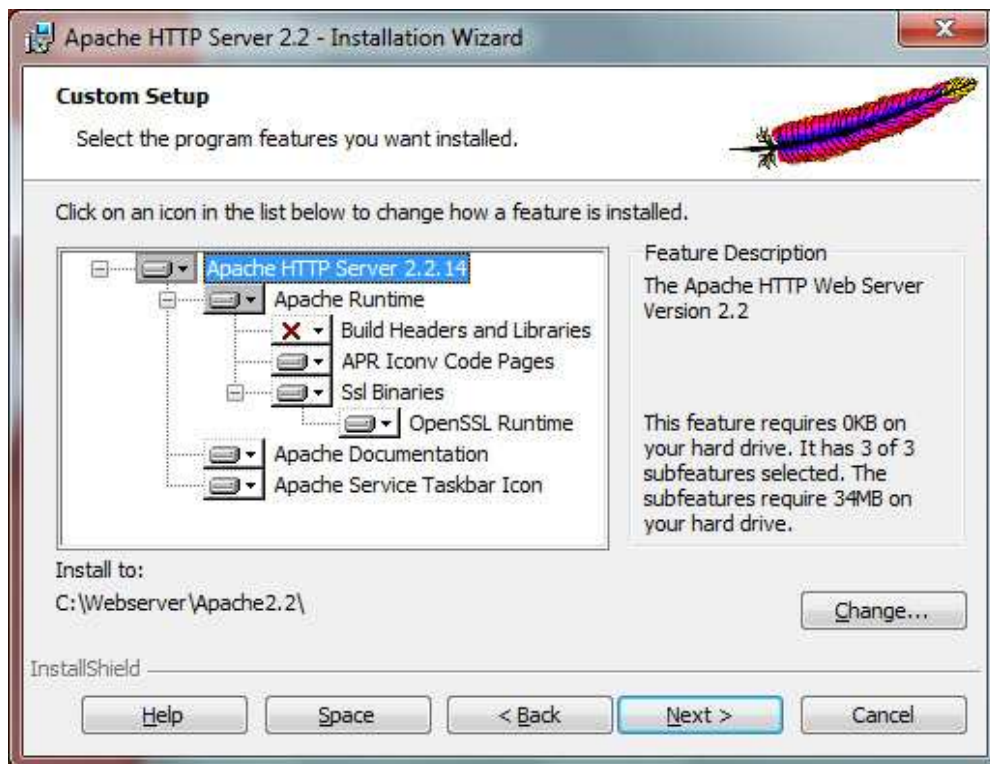
Como podemos ver nas seguintes imagens, temos de preencher alguns campos, com o nome da rede e do servidor, o *e-mail* e o porto de acesso.



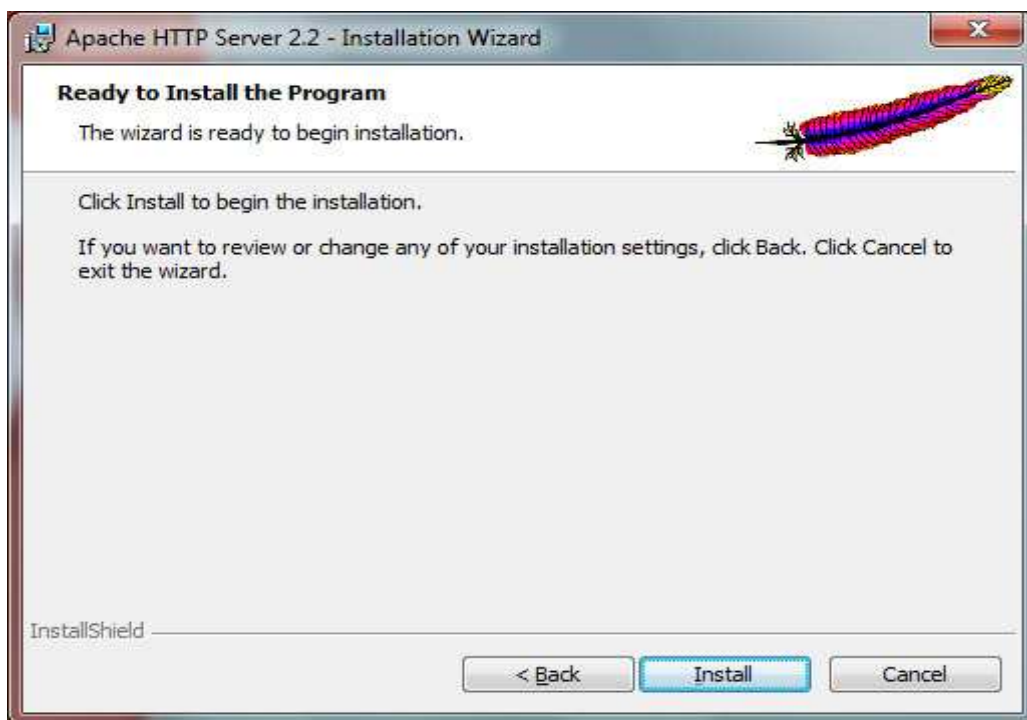
O passo seguinte é escolher uma instalação típica ou personalizada.



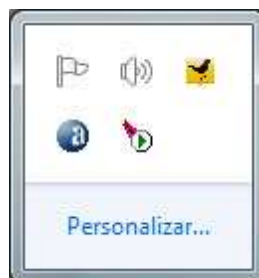
Na janela da instalação personalizada, apresenta várias opções que podemos escolher para instalação.



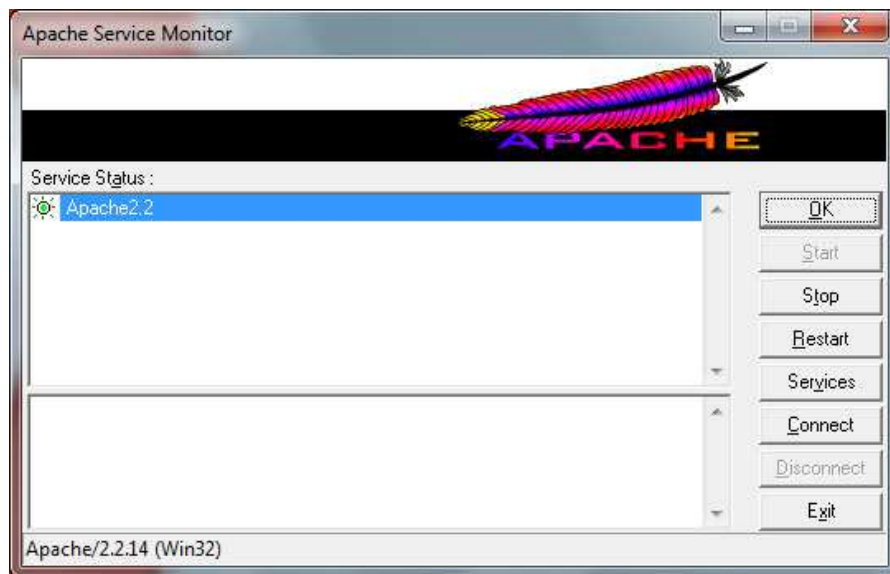
De seguida é só carregar no botão de Install.



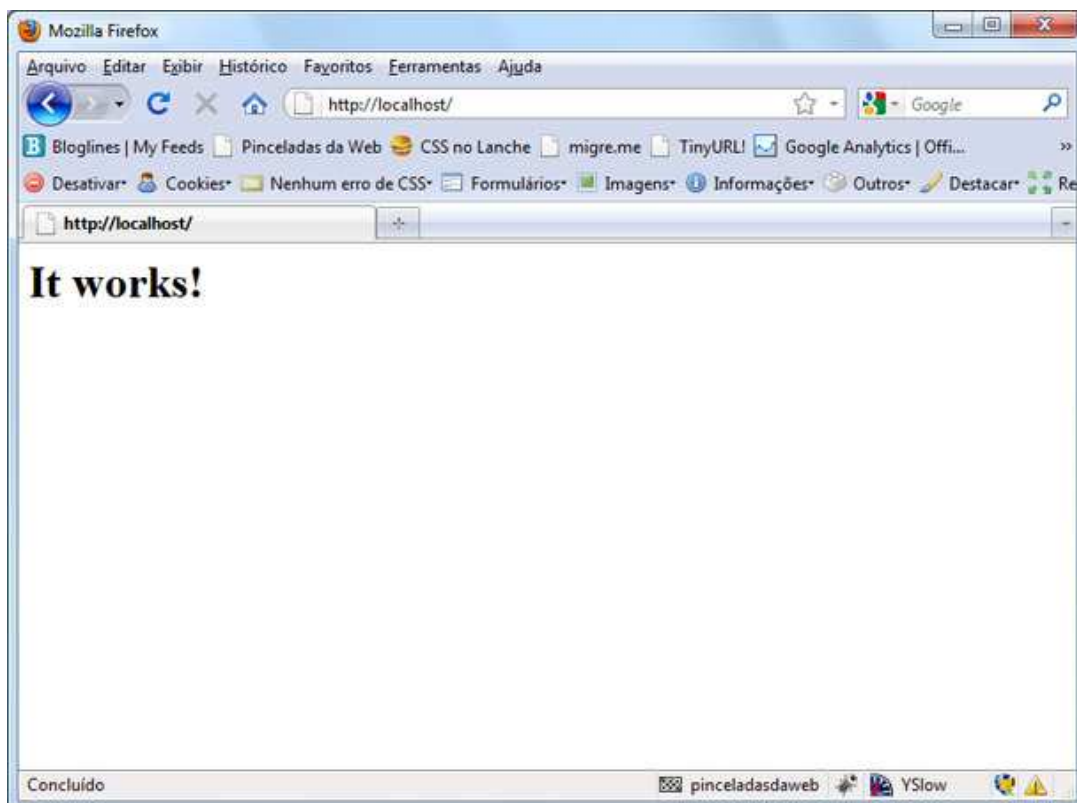
Após o término da instalação, teremos um ícone no *System Tray*, o Apache Monitor.



Através dele temos várias opções do Apache.



Finalmente, para testar se a instalação correu bem, podemos escrever *http://localhost* no *browser* à nossa escolha e aparecerá a seguinte janela se a instalação foi feita correctamente.



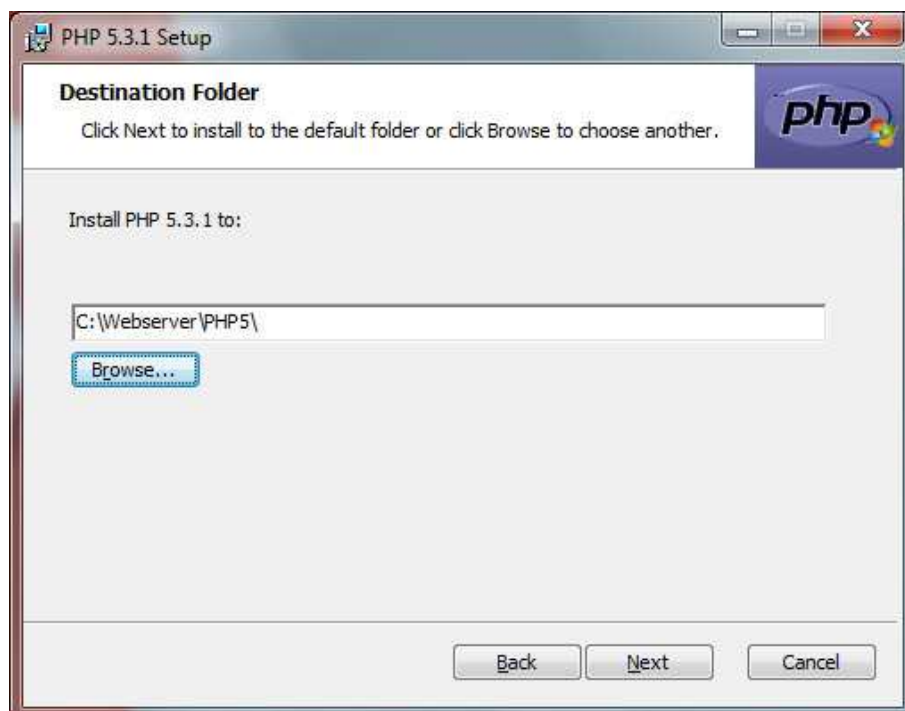
Para alojar as nossas próprias páginas, basta colocar os ficheiros das páginas no directório C:\WebServer\Apache2.2\htdocs.

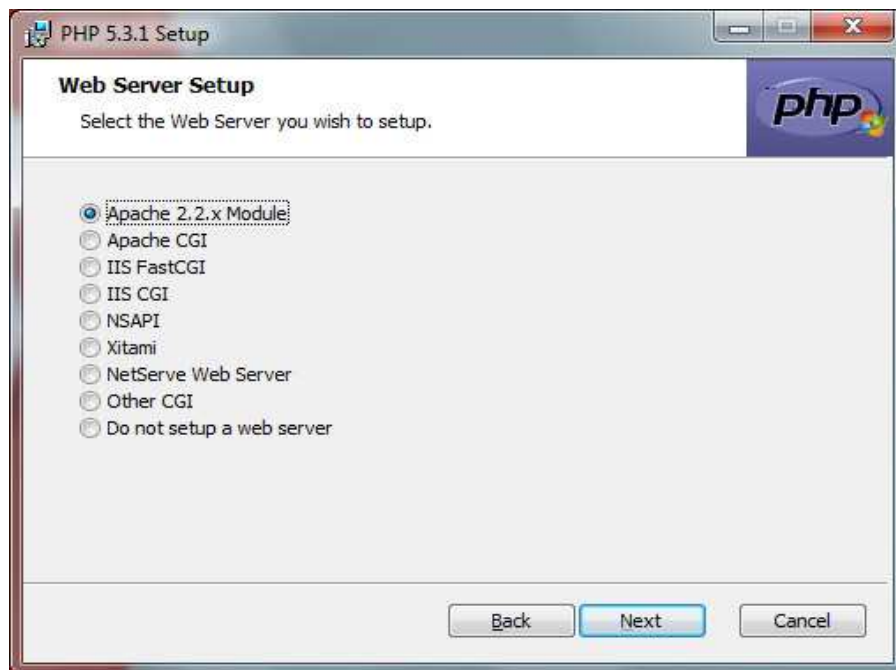
## 2: Instalação e Configuração do PHP 5.3.1

Na instalação do PHP, teremos estas janelas.

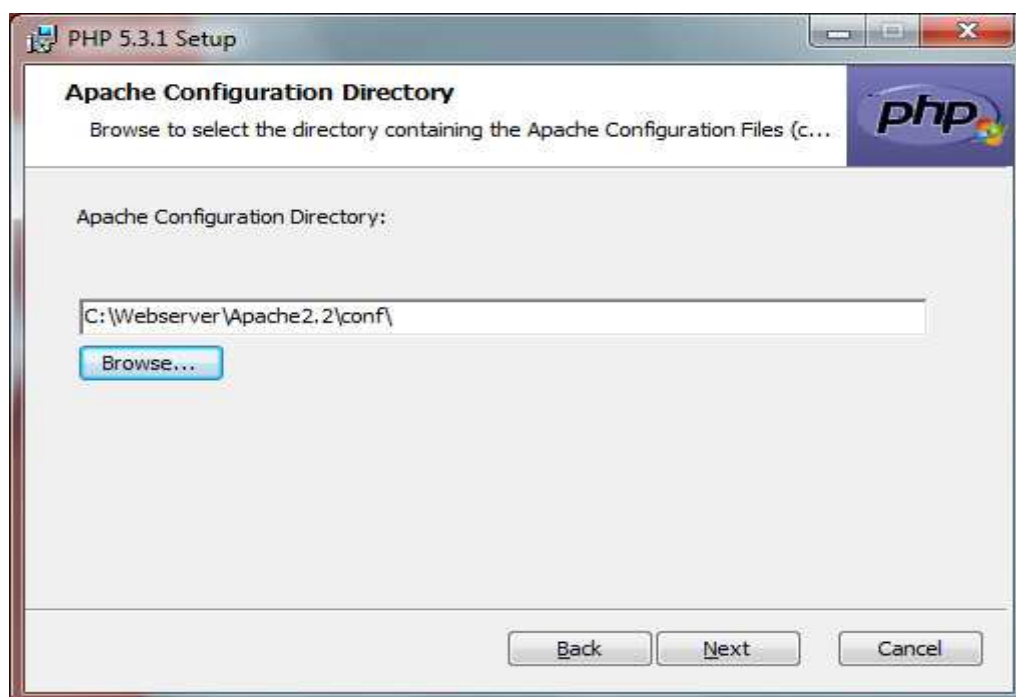


Nas 2 janelas seguir podemos indicar o local de instalação e tipo de servidor utilizado.



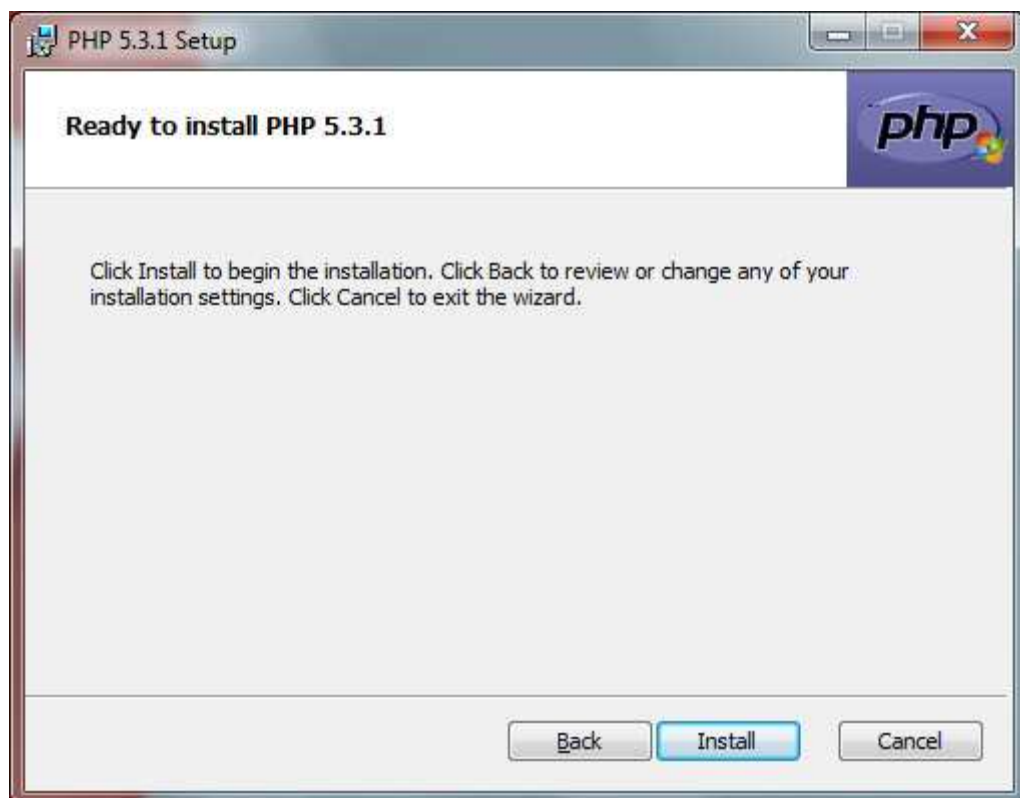
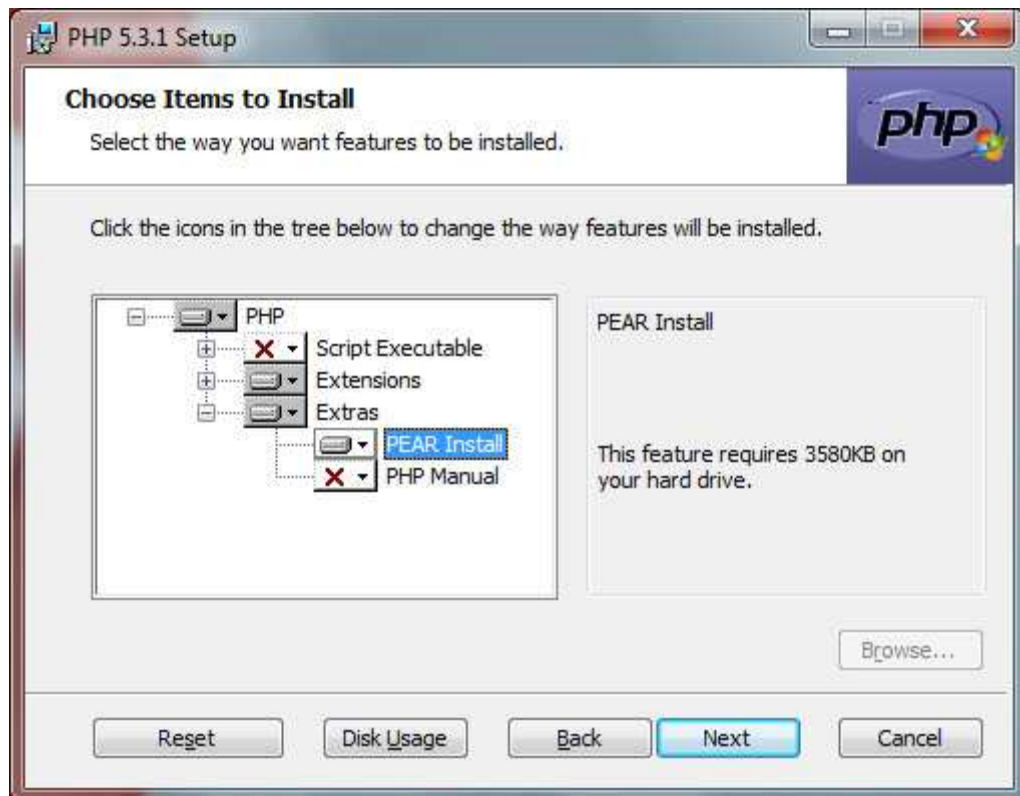


De seguida devemos indicar o local com ficheiro de configuração do Apache, para que a configuração seja automatizada.



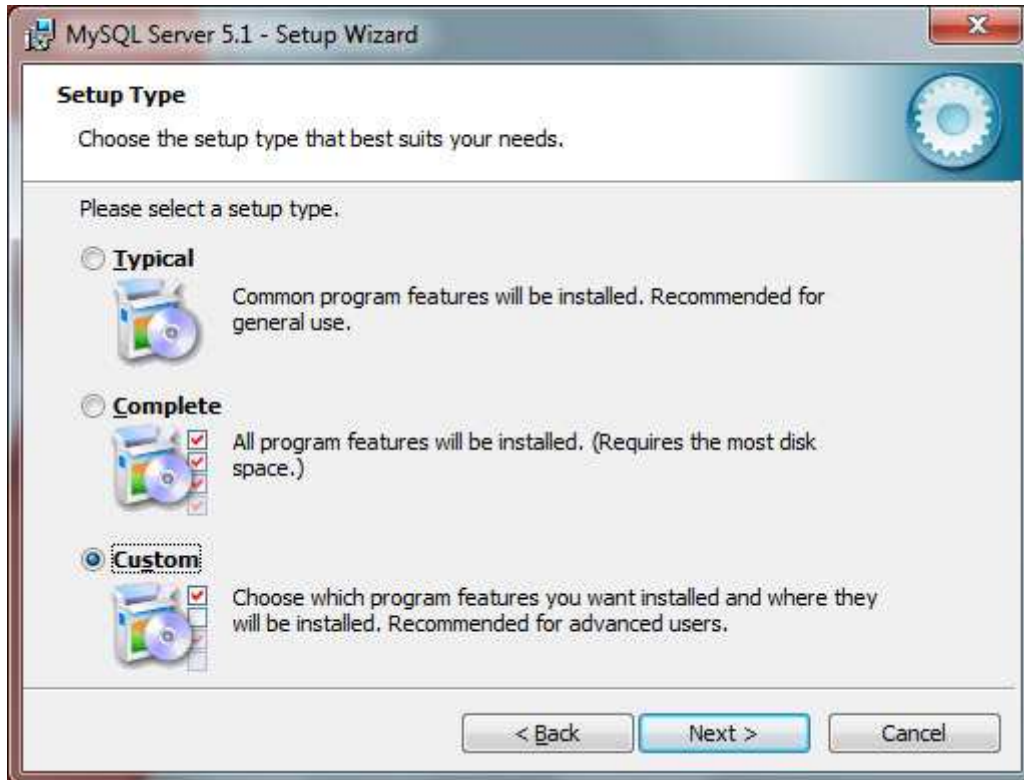


O próximo passo é escolher as extensões que queremos instalar juntamente com PHP e instalar.

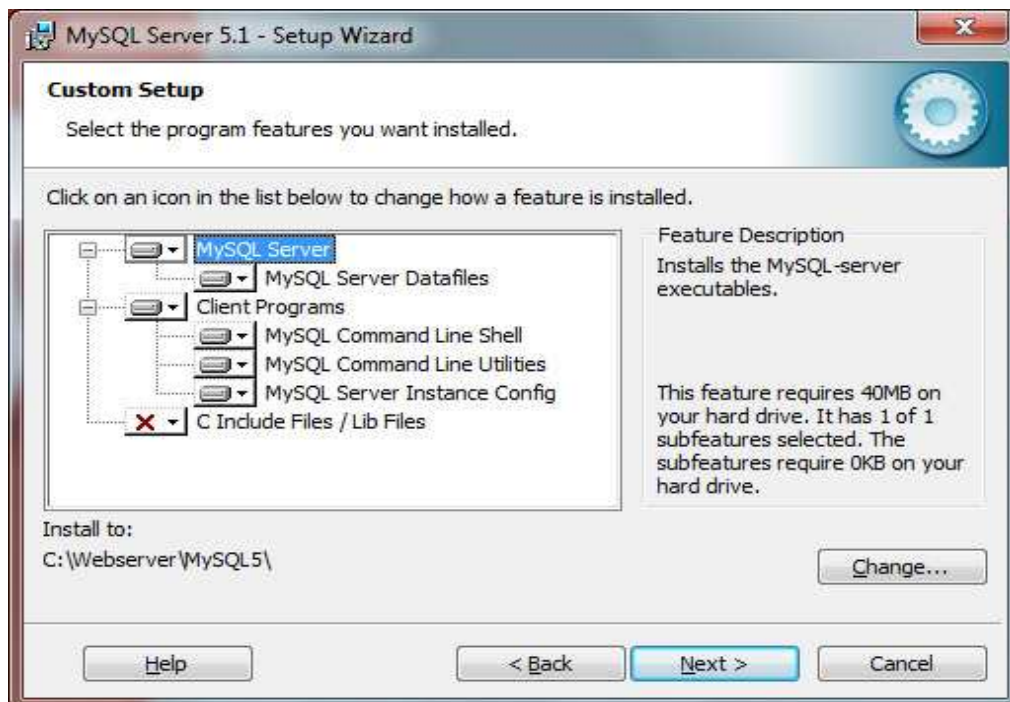


### Passo 3: Instalação e Configuração do MySQL 5.1

Depois de executar o ficheiro de instalação, devemos escolher o tipo de instalação que queremos como mostra a seguinte janela.



Escolher o local e as extensões da instalação.





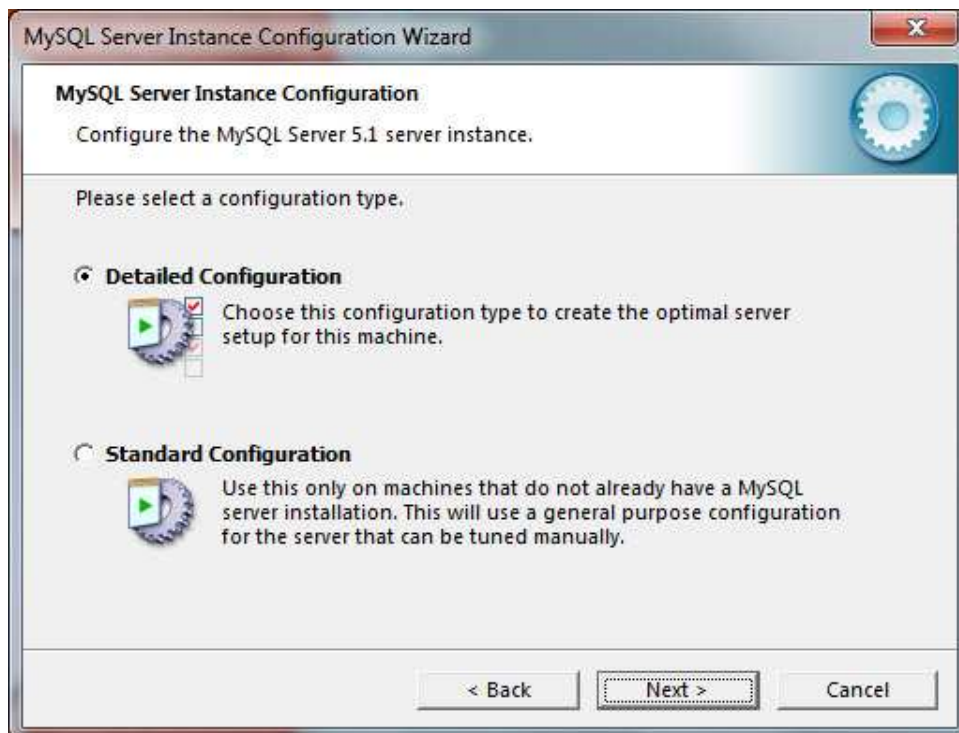
Estando tudo Ok, basta clicar em *Install*:



Após o final da instalação, vamos efectuar as configurações do MySQL, basta seguir os passos que serão mostrados aqui:



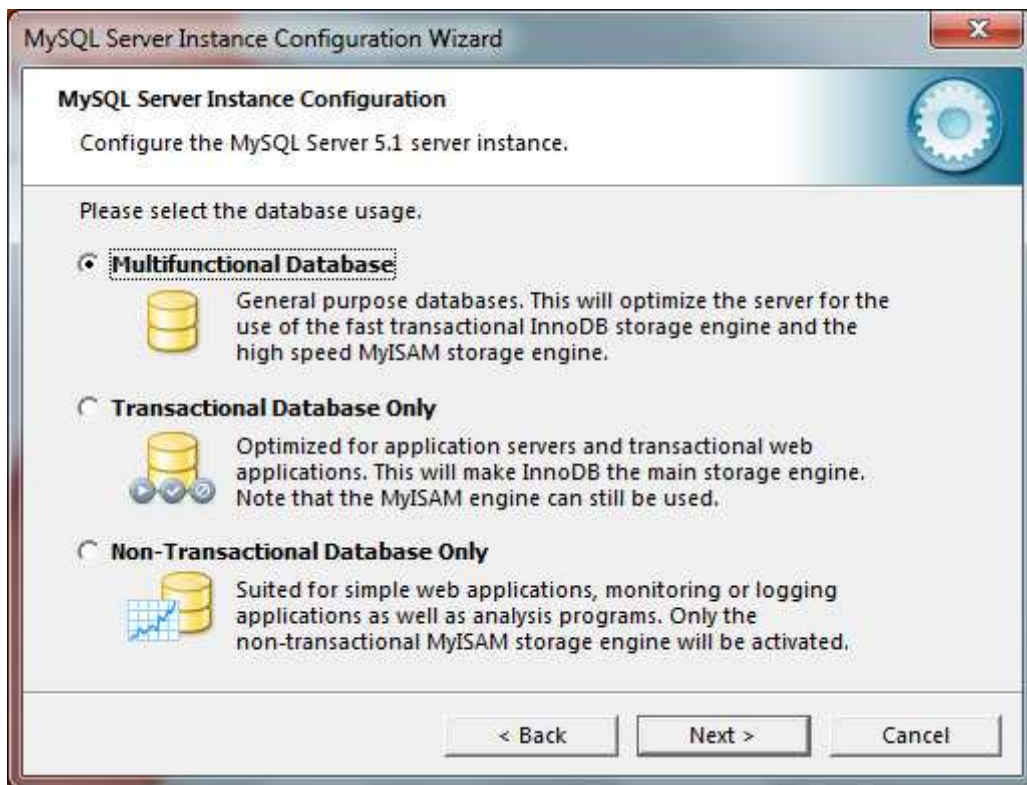
A seguir escolha configuração detalhada:



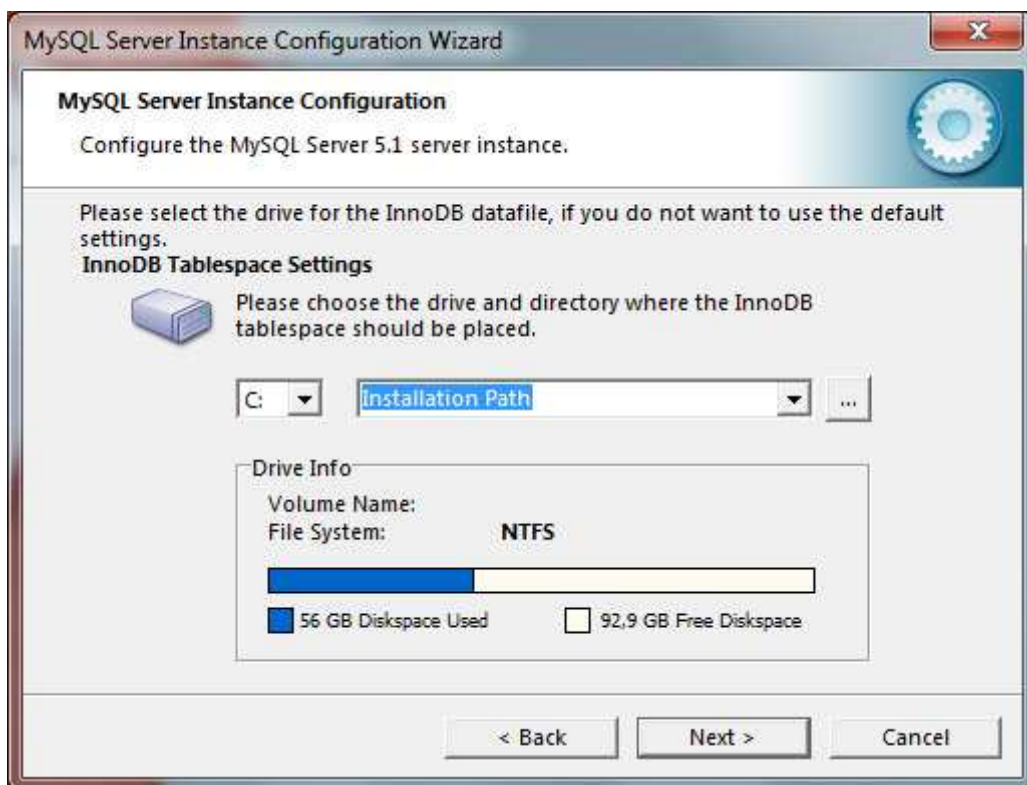
Escolha *Developer Machine*:



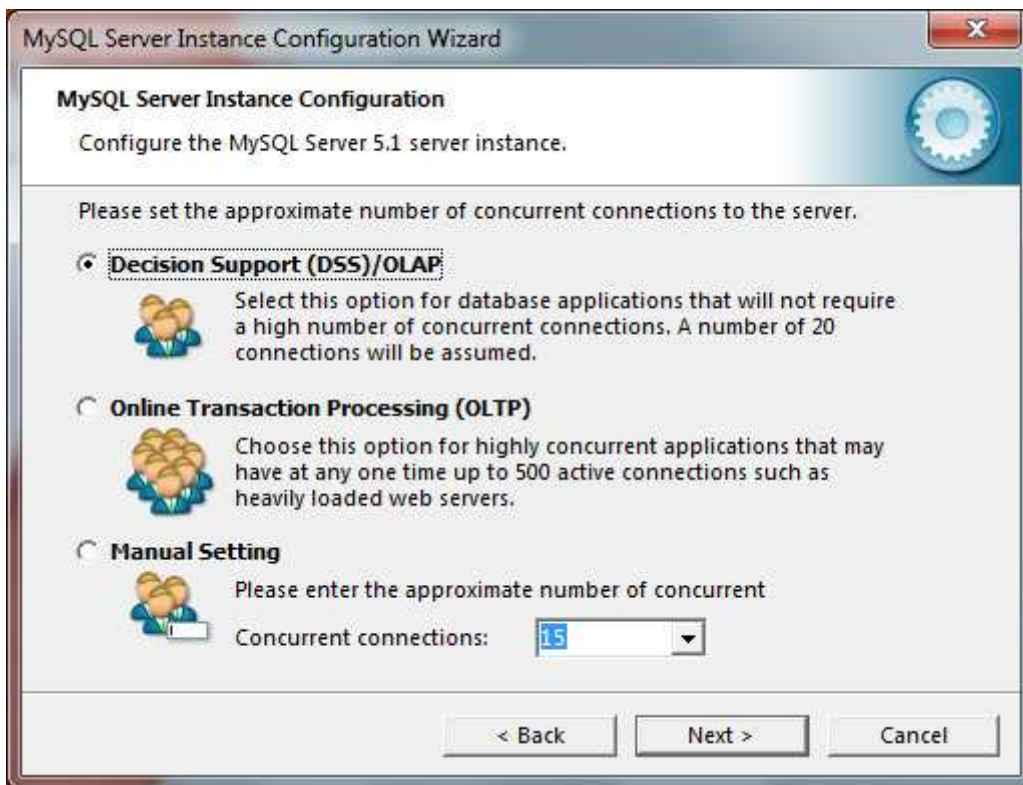
Escolha *Multifunctional Database*:



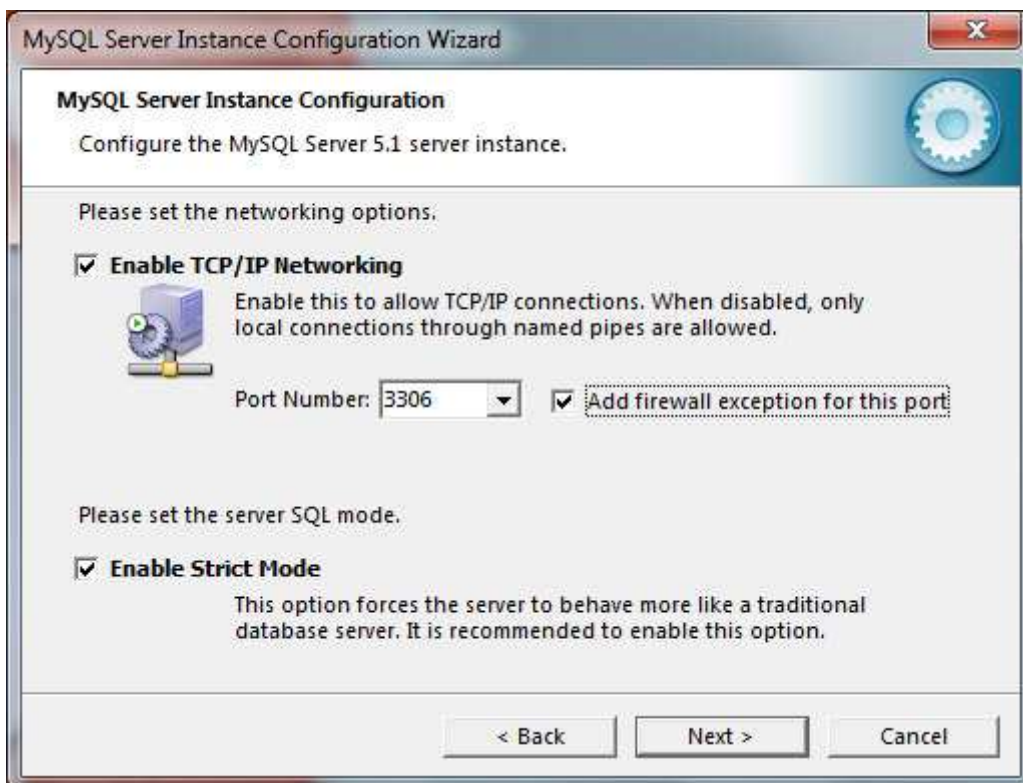
No passo ao que se refere a instalação do InnoDB, deixe como o caminho *Installation Patch*:



Selecionar *Decision Support (DSS)/OLAP*:

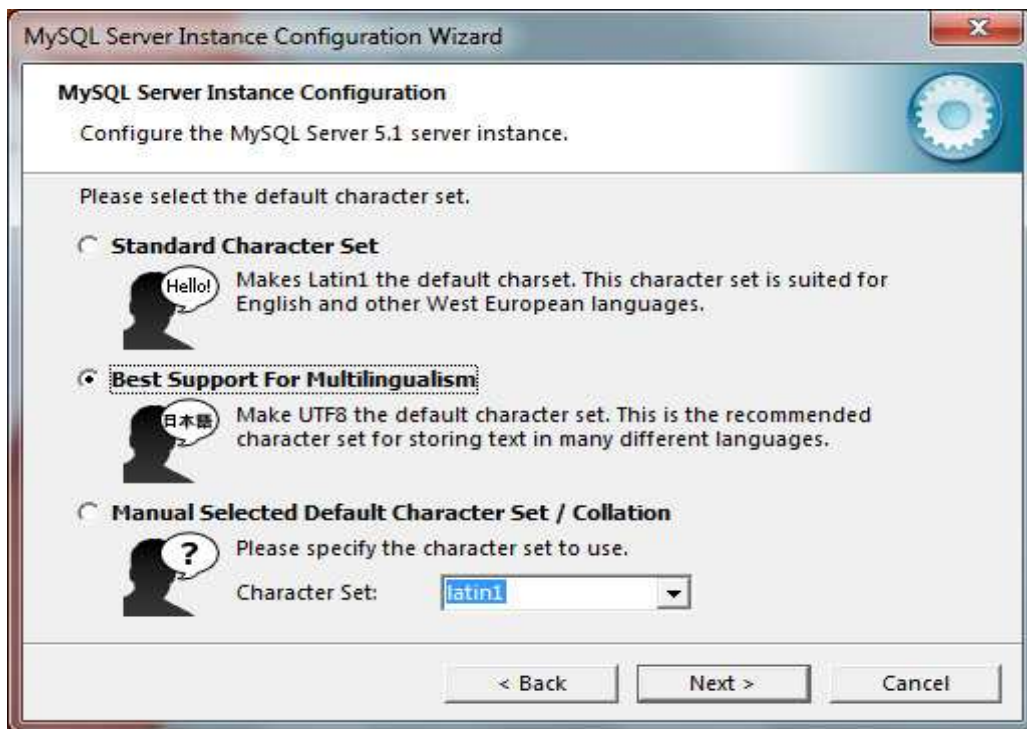


No passo seguinte seleccionar *Add firewall exception for this port*:





Selecione *Best Support for Multilingualism*:



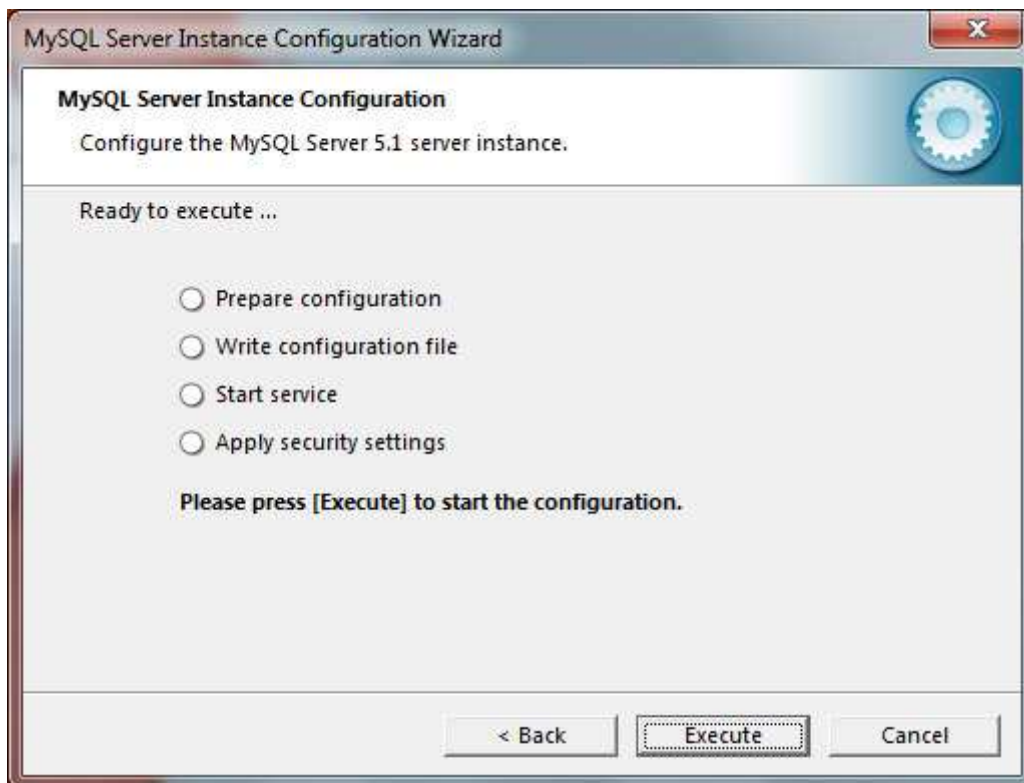
A próxima opção é de incluir o caminho do MySQL na variável de ambiente “Path” do Windows. Deixe-a selecionada pois assim você pode chamar os executáveis do MySQL a partir de qualquer *prompt*, em qualquer pasta:



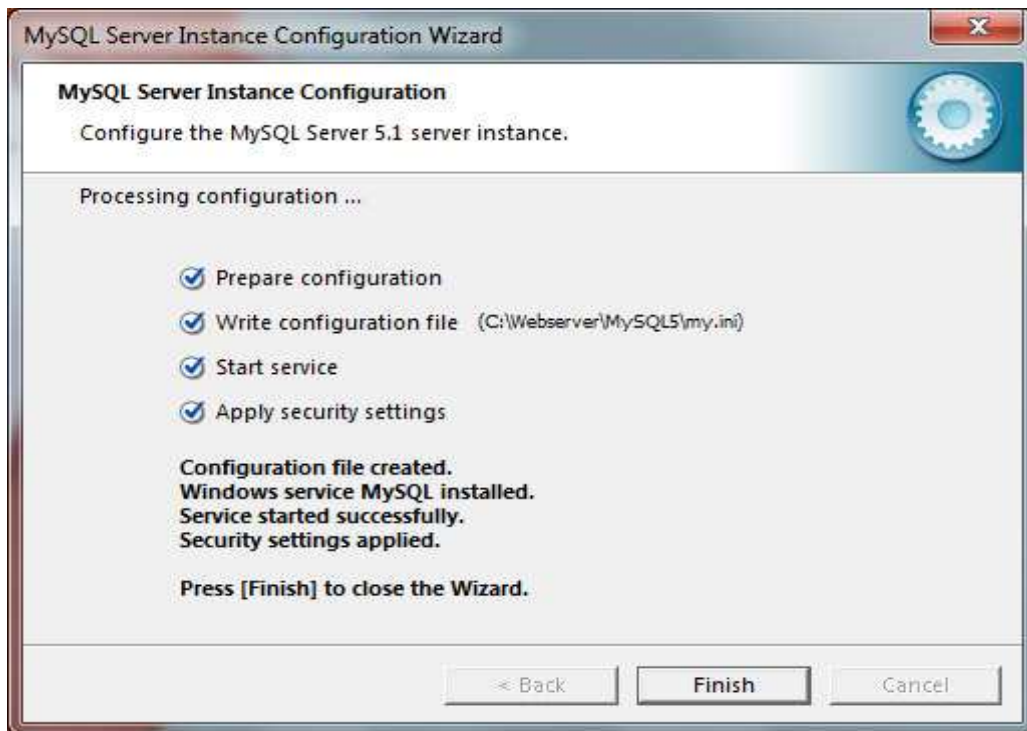
No próximo passo deve-se definir a senha de root, ele é o administrador do MySQL.



No próximo passo basta você clicar em execute para que o MySQL dê início ao processo de configuração:



Caso tudo tenha corrido OK, você verá um ecrã como essa, caso tenha algum problema, efectue as configurações novamente:



#### Passo 4: Instalação do PhpMyAdmin

Após efectuar o download do PhpMyAdmin, descompacte o mesmo para directório *htdocs* da pasta de instalação do Apache e renomeie a pasta para **phpmyadmin**. Para aceder o PhpMyAdmin basta introduzir o seguinte: *http://localhost/phpmyadmin/* no browser, e entrar como utilizador root e senha que foi configurado na instalação do MySQL.

Para login automático, teremos de mudar o nome do ficheiro *config.sample.inc.php* para *config.inc.php* contido na pasta *phpmyadmin*, e editar o ficheiro, substituindo a linha:

```
$cfg['Servers'][$i]['auth_type'] = 'cookie'
```

por:

```
1.$cfg['Servers'][$i]['user'] = 'root';
2.$cfg['Servers'][$i]['password']=' '; //Digite aqui a senha de root
3.$cfg['Servers'][$i]['auth_type'] = 'config';
```