



SENAI – Santos Dumont

CURSO TECNICO EM REDES DE COMPUTADORES

André Lucas Maegima

Breno Henrique Borges Santos

Bruno dos Santos Mauricio

Guilherme Rios da Cunha

Leonardo Ribeiro dos Santos

**INTERFACE PARA CONTROLE REMOTO DE BRAÇO ROBÓTICO VIA SERVIDOR WEB
SEGUTO**

Professor Orientador:

Esp. Airton Cézar Zombardi

Me. Josemar Monteiro da Silva

Esp. Kleber Gelli

S.B. Wellington Carlos Joffre

São José dos Campos-SP

2016

André Lucas Maegima
Breno Henrique Borges Santos
Bruno dos Santos Mauricio
Guilherme Rios da Cunha
Leonardo Ribeiro dos Santos

**INTERFACE PARA CONTROLE REMOTO DE BRAÇO ROBOTICO VIA SERVIDOR WEB
SEGURO**

Projeto de Pesquisa ou Monografia
apresentado(a) como requisito parcial para
obtenção do título de técnico em Redes de
Computadores pela escola SENAI Santos
Dumont de São José dos Campos

Orientador: Prof. Esp. Airton Cézar Zombardi
Me. Josemar Monteiro da Silva
Esp. Kleber Gelli
S.B. Wellington Carlos Joffre

São José dos Campos-SP
2016

DEDICATÓRIA

Dedico a meu pai Luiz Carlos Hiroyuki Maegima, minha Mãe Andreia Regina de Barros Maegima, meus irmãos, a Deus, ao meu grupo e todos aqueles que nos auxiliaram até aqui.

André Lucas Maegima

DEDICATÓRIA

Dedico a minha mãe Cassiani Rufino Borges dos Santos, ao meu pai Robson Rodrigues dos Santos, a minha irmã, aos meus avós e avôs, a Deus, e por todos que ajudaram no projeto.

Breno Henrique Borges Santos

DEDICATÓRIA

Agradeço a Deus por me dar o dom da vida, dedico aos dois homens de minha vida, meu pai Adão de Souza Mauricio e meu irmão Peterson dos Santos Mauricio, meus exemplos masculinos, agradeço também a minha mãe, Roseli Inacia Mauricio, por batalhar duro sempre para dar o melhor possível a nossa família, um agradecimento especial a todos os professores que já passaram em minha vida e aqueles que ainda estão por vir, a instituição SENAI por disponibilizar o curso, agradecer aos amigos que nos incentivam, agradeço aos companheiros de curso pela caminhada, agradecimento final vai ao grupo que batalhou arduamente nesse último semestre até a conclusão do projeto, obrigado!

Bruno dos Santos Mauricio

DEDICATÓRIA

Dedico este TCC a Deus que me deu o dom da vida, a minha mãe Weslania Gonçalves Rios da Cunha, mina irmã Carolina Rios da Cunha, e minhas amigas Juliana Hediger Borges e Giulia Prado Coelho de Lemos que acompanharam e me apoiaram e durante esse projeto, e a meu falecido pai, Silas Arbolato da Cunha, que se orgulharia de ver onde cheguei

Guilherme Rios da Cunha

DEDICATÓRIA

Dedico esse trabalho ao meu pai Hailton
Aparecido dos Santos, minha mãe Giovana
Ribeiro Pereira dos Santos, familiares, amigos e
professores que nos apoiaram durante esse
projeto.

Leonardo Ribeiro dos Santos

AGRADECIMENTOS

A Deus, por iluminar nossos passos para que possamos trilhar nossos caminhos nesta nova jornada.

Aos nossos pais por nos apoiarem em cada decisão e nos incentivarem nos estudos.

Ao Diretor do SENAI – Santos Dumont, por ter proporcionado o Curso técnico de Redes de Computadores.

Ao Coordenador do Curso Técnico em Redes de Computadores, José Rogério Chavier, pela ética e pela destreza.

Aos professores Airton Cézar Zombardi, Josemar Monteiro da Silva, Kleber Gelli, Wellington Carlos Joffre, pela orientação deste estudo.

Aqueles, que direta ou indiretamente, colaboraram para o desenvolvimento desta pesquisa.

Todos os caminhos estão errados quando
você não sabe aonde quer chegar.

William Shakespeare

RESUMO

Devido ao crescente número de acidentes em empresas, houve a necessidade da substituição do homem pela máquina em trabalhos braçais ou trabalhos que envolvem o movimento repetitivo. Entretanto somente com essa substituição o número de acidentes com ou sem vítimas nessas entidades não diminuíram, isso porque alguns trabalhos ainda precisam que o funcionário opere uma máquina de perto. Para isso deve se desenvolver métodos para o controle de máquinas com a mínima interação física do trabalhador com o equipamento.

Esse métodos visam diminuir o número de acidentes e acidentados, uma vez que cada ocorrência gera um alto custo tanto em tempo como em recursos à empresa, além disso, esses acontecimentos geram um custo imensurável para o trabalhador e sua família.

A tecnologia vem avançando cada dia mais, em vista das pesquisas em livros e monografias fica claro que ela se torna cada vez mais essencial atualmente, porém junto a ela podem vir os riscos à segurança, portanto essas tecnologias devem avançar juntamente com a preservação da integridade dos seres humanos.

O método de abordagem utilizado para a pesquisa é o cartesiano, pois o projeto visa a melhoria na interface de controle de um braço já existente, a sua abordagem será quantitativa uma vez que ela se apoia em dados estatísticos e pesquisas de causas e consequência.

Este trabalho de conclusão de curso tem como objetivo desenvolver uma interface para controle de um braço robótico, controlado através de uma rede de computadores mais segura para proporcionar uma maior proteção ao trabalhador, com isso aumentar a produtividade das empresas e consequentemente aumentar o seu lucro.

Palavras chave: Braço Robótico. Servidor Web. Interface de Controle. Segurança.

ABSTRACT

Due to the increasing number of accidents in companies, there was a need to replace the man by the machine in manual labors or jobs that involve repetitive movement. However, with only this replacement the number of accidents with or without victims in these entities has not diminished, because some jobs still require the employee to operate a machine closely. For this, methods must be developed to control machines with minimal physical interaction between the worker and the equipment.

The technology is advancing every day more, in view of the researches in books and monographs it is clear that it becomes more and more essential nowadays, but next to it can come the risks to the security, therefore these technologies must advance along with the preservation of the integrity of human beings.

The approach used for the research is the Cartesian, since the project aims at improving the control interface of an existing arm; its approach will be quantitative since it relies on statistical data and research on causes and consequences.

These methods aim to reduce the number of accidents, since each occurrence generates a high cost both in time and in resources to the company, besides these events generate an immeasurable cost for the worker and his family.

This course completion work aims to develop an interface for controlling a robotic arm, controlled through a safer computer network to provide more protection to the work, thereby increase the productivity of companies, and consequently increase their profit.

Keywords: Robotic Arm. Web Server. Control Interface. Security.

LISTA DE FIGURAS

Figura 1 Acidentes de Trabalhos Registrados de 2007 a 2011.	22
Figura 2 Braço Robótico.	28
Figura 3 Arduino.	29
Figura 4 Ábaco.	30
Figura 5Régua de Cálculo.	31
Figura 6 Máquina de Pascal.	31
Figura 7 Oscilador HP200A.	32
Figura 8 Computador Eniac.	33
Figura 9 Computador Harvard MARK I.	34
Figura 10 Computador IBM SSEC.	34
Figura 11 IBM 7030.	35
Figura 12 IBM 360/91.	36
Figura 13 Altair 8800 IBM 360/91.	37
Figura 14 Apple I.	37
Figura 15 Esquema de Sistema Operacional.	38
Figura 16 IBM 709.	39
Figura 17 Unidade PDP-7.	41
Figura 18 Thompson (de pé) e Ritchie programando no PDP-11 em um terminal teletipo.	42
Figura 19 Caricatura de um Gnu, símbolo do Projeto GNU.	44
Figura 20 Logo do Projeto Debian.	47
Figura 21 Lançamentos do Debian.	48
Figura 22 Logo do Ubuntu.	49
Figura 23 Lançamentos do Ubuntu.	50
Figura 24 Tela Inicial do Windows 95.	52
Figura 25 Tela inicial do Windows NT.	53
Figura 26 Tela Inicial do Windows 98.	54
Figura 27 Área de trabalho do Windows XP.	54
Figura 28 Área de Trabalho do Windows 7.	55
Figura 29 Área de Trabalho do Windows 10.	55
Figura 30 Interface de Linha de Comando (CLI).	56
Figura 31 Interface de Texto para Usuário (TUI).	57
Figura 32 Interface Gráfica para Usuário (GUI).	58
Figura 33 Sistema Cliente-Servidor.	60
Figura 34 Desenvolvedores de Servidor Web (Sites Ativos).	62
Figura 35 Logo da Empresa Wi-Fi Aliance.	64
Figura 36 Logo do MySQL.	66
Figura 37 Cifra de Scytale.	68
Figura 38 Cifra de Cesar.	68
Figura 39 Cifra de Vigenére.	69
Figura 40 Maquina Enigma.	70
Figura 41 HTTPS.	71
Figura 42 Exemplo de Programa em C (fração).	73
Figura 43 Exemplo de HTML	77
Figura 44 Ligações do Modulo.	85
Figura 45 Instalação do Virtual Box (Parte 1).	86
Figura 46 Instalação do Virtual Box (Parte 2).	86

Figura 47 Instalação do Virtual Box (Parte 3).	87
Figura 48 Instalação do Virtual Box (Parte 4).	87
Figura 49 Instalação do Virtual Box (Parte 5).	88
Figura 50 Instalação do Virtual Box (Parte 6).	88
Figura 51 Instalação do Virtual Box (Parte 7).	89
Figura 52 Criação de VM (parte 1).	89
Figura 53 Criação de VM (parte 2).	90
Figura 54 Criação de VM (parte 3).	90
Figura 55 Criação de VM (parte 4).	91
Figura 56 Criação de VM (parte 5).	91
Figura 57 Criação de VM (parte 6).	92
Figura 58 Criação de VM (parte 7).	92
Figura 59 Instalação do Ubuntu (parte 1).	93
Figura 60 Instalação do Ubuntu (parte 2).	93
Figura 61 Instalação do Ubuntu (parte 3).	94
Figura 62 Instalação do Ubuntu (parte 4).	94
Figura 63 Instalação do Ubuntu (parte 5).	95
Figura 64 Instalação do Ubuntu (parte 6).	95
Figura 65 Instalação do Ubuntu (parte 7).	96
Figura 66 Instalação do Ubuntu (parte 8).	96
Figura 67 Instalação do Ubuntu (parte 9).	97
Figura 68 Área de trabalho da máquina virtual.	97
Figura 69 Instalação do LAMP.	98
Figura 70 Finalização da Instalação do LAMP.	98
Figura 71 Resultado da Instalação do LAMP.	99
Figura 72 Criação de um banco de dados.	100
Figura 73 Criação do usuário do banco de dados.	100
Figura 74 Acesso local ao MySQL.	100
Figura 75 Criação de tabela no MySQL.	101
Figura 76 Inserção de dados na tabela.	101
Figura 77 Página de acesso (parte 1).	102
Figura 78 Página de acesso (parte 2).	102
Figura 79 Layout da página de acesso.	103
Figura 80 login.php.	103
Figura 81 verifica_con.php.	104
Figura 82 site.php (parte 1).	104
Figura 83 site.php (parte 2).	105
Figura 84 site.php (parte 3).	105
Figura 85 Layout do site.php.	105
Figura 86 gerenciar.php (parte 1).	106
Figura 87 gerenciar.php (parte 2).	106
Figura 88 gerenciar.php (parte 3).	107
Figura 89 registros.php.	107
Figura 90 Layout do registros.php.	108
Figura 91 Página que envia comando.	108
Figura 92 servidor.c (inclusão das bibliotecas/definição das estruturas).	109
Figura 93 Funções erro, inicializaCliente e abreSocket.	110
Figura 94 Funções abreSocket e clienteSocket.	110
Figura 95 Funções mensagemInicial e timeoutConexao.	111
Figura 96 Funções VerificaConexao e fechaConexao.	111

Figura 97 Funções connection e novaConexao.	112
Figura 98 função envia.	112
Figura 99 <i>main</i> do arquivo servidor.c.	113
Figura 100 braco_wifi.ino.	113
Figura 101 <i>setup</i> do arduino.	115
Figura 102 <i>loop</i> do arduino (parte 1).	115
Figura 103 <i>loop</i> do arduino (parte 2).	116
Figura 104 Primeira Interface para controle do braço robótico.	119
Figura 105 Interface Finalizada.	119
Figura 106 Resultados do projeto.	120

LISTA DE GRÁFICOS

Gráfico 1 Entrevistados que Trabalham.	81
Gráfico 2 Esforço Físico no Trabalho.	81
Gráfico 3 Risco a Saúde no Serviço.	82
Gráfico 4 Trabalham em Linha de Produção.	82
Gráfico 5 Chefes de Empresa ou Microempresa.	83
Gráfico 6 Se Interessam no Desenvolvimento de sua Empresa.	83
Gráfico 7 Pessoas que Compraria o Braço.	84
Gráfico 8 Acreditam no Projeto.	84

LISTA DE TABELAS

Tabela 1 Cronograma do Projeto.	79
Tabela 2 Tabela de Custos.	79
Tabela 3 Funções do programa servidor.c.	110

LISTA DE SIGLAS E ABREVIATURAS

ANSI	<i>American National Standards Institute</i>
ARPA	<i>Advanced Research Project Agency</i>
ASP	<i>Active Server Pages</i>
BCPL	<i>Basic Combined Programming Language</i>
BIOS	<i>Basic Input/Output System</i>
BSD	<i>Berkley Software Distribution</i>
CERT	Centro de Estudos, Resposta e Tratamento
CLI	<i>Command Line Interface</i>
CPU	<i>Central Processing Unit</i>
CRUD	<i>Create, Read, Update e Delete</i>
CTSS	<i>Compatible Time-Sharing System</i>
DNS	<i>Domain Name Server</i>
Eniac	<i>Electronic Numerical Integrator and Computer</i>
GML	<i>Geography Markup Language</i>
GNU	<i>GNU is Not Unix</i>
GUI	<i>Grafic User Interface</i>
Hi-Fi	<i>High Fidelity</i>
HTML	<i>Hypertext Markup Language</i>
HTTP	<i>Hypertext Tranference Proto</i>
HTTPs	<i>Hyperteht Tranference Proto Security</i>
HyTime	<i>Hypermidia/Time-based Document Structuring Language</i>
IBM	<i>International Business Machines</i>
IETF	<i>Internet Engineering Task Force</i>
IP	<i>Internet Protocol</i>
LAMP	Linux, Apache, MySQL e PHP
LGPL	<i>Library General Public License</i>
LST	<i>Long-term Support</i>
MAC	<i>Multiple Access Computers & Man And Computers</i>
MIT	<i>Massachusetts Institute of Technology</i>
Multics	<i>Multiplexed Information and Computing Service</i>
OS	<i>Operating System</i>
PDP	<i>Programmable Data Processor</i>
PHP	<i>Hypertext Preprocessor ou Personal Home Page</i>
ROM	<i>Ready Only Memory</i>
SGBDR	Sistema Gerenciador de Banco de Dados Relacional
SGML	<i>Standard Generalizes Markup Language</i>
SO	Sistema Operacional
SPI	<i>Software in Public Interest</i>
SQL	<i>Structured Query Language</i>
SSEC	<i>Selective Sequence Electronic Calculator</i>
SSL	<i>Secure Socket Layer</i>
TCP	<i>Transmission Control Protocol</i>
TUI	<i>Text User Interface</i>
Unics	<i>Uniplexed Information and Computing Service</i>
VDI	<i>Virtual Disk Image</i>
VM	<i>Virual Machine</i>
W3C	<i>World Wide Web Consortium</i>
Wi-Fi	<i>Wireless Fidelity</i>
WLAN	<i>Wireless Local Area Network</i>
WWW	<i>World Wide Web</i>
XHTML	<i>Extensible Hypertext Markup Language</i>
XML	<i>Extensible Markup Language</i>

SUMÁRIO

1	INTRODUÇÃO	21
1.1	Objetivo	22
1.1.1	Objetivos gerais.....	22
1.1.2	Objetivos específicos.....	22
1.2	Justificativa	23
2	FUNDAMENTAÇÃO TEÓRICA	24
2.1	Motores elétricos	24
2.1.1	História	24
2.2	Robótica	26
2.2.1	Braço Robótico	26
2.2.2	Arduino	28
2.3	Computadores	29
2.3.1	História	29
2.3.1.1	O ábaco.....	29
2.3.1.2	Régua de Calculo	30
2.3.1.3	Máquina de Pascal	31
2.3.1.4	Programação Funcional.....	31
2.3.1.5	Máquina analítica	32
2.3.1.6	Teoria de Boole	32
2.3.1.7	Computador Mecânico.....	32
2.3.1.8	A primeira geração	33
2.3.1.9	A segunda geração	34
2.3.1.10	A terceira geração	35
2.3.1.11	A quarta geração	36
2.4	Sistema operacional	37
2.4.1	Criação do sistema operacional.....	38
2.4.2	Projeto MAC (MIT Project MAC)	39
2.4.3	Unix	40
2.4.3.1	Unics	40
2.4.3.2	Projeto do Unix	41
2.4.3.3	O novo Unix.....	42
2.4.3.4	Unix nas universidades.....	43
2.4.3.5	Comercialização do Unix	43
2.4.4	Projeto GNU	43
2.4.4.1	Free Software Foundation	44
2.4.4.2	Definição de Software Livre	44
2.4.5	GNU/Linux	45
2.4.5.1	Distribuições GNU/Linux	45
2.4.5.1.1	Debian	46
a)	História.....	47
b)	Sistema de empacotamento Debian.....	48
c)	Releases estável, teste e instável	48
2.4.5.1.2	Ubuntu.....	49
a)	Origem da palavra “Ubuntu”	49
b)	Como tudo começou	49
c)	Lançamentos do Ubuntu	50
d)	Ubuntu atualmente	51
2.4.6	Windows	51
2.4.6.1	História	52
2.5	Interface do utilizador	56
2.5.1	Interface de Linha de Comando (CLI)	56
2.5.2	Interface tipo Texto para Usuário (TUI).....	57
2.5.3	Interface Gráfica para Usuário (GUI)	57

2.6	Virtualização	58
2.6.1	Máquina Virtual.....	58
2.6.1.1	Virtual box	58
2.7	Internet	58
2.7.1	Historia	59
2.7.2	Servidores	59
2.7.2.1	Servidores WEB	61
2.7.2.1.1	Apache	62
2.7.2.1.2	World Wide Web (WWW)	62
2.7.2.1.3	Páginas estáticas e dinâmicas	63
2.8	WI-FI.....	64
2.9	Banco de dados.....	65
2.9.1	Bancos de Dados Relacionais	65
2.9.1.1	SQL (Structured Query Language)	65
2.9.1.2	MySQL	66
2.10	Segurança de redes.....	66
2.10.1	Políticas de segurança	66
2.10.2	Criptografia.....	66
2.10.2.1	Criptografia Clássica.....	67
2.10.2.1.1	Cifra de Scytale	67
2.10.2.1.2	Cifra de Cesar	68
2.10.2.1.3	Cifra de Vigenère.....	69
2.10.2.2	Criptografia Moderna	69
2.10.2.3	Chave simétrica.....	70
2.10.2.4	Chave assimétrica	70
2.10.3	Assinatura digital	71
2.10.4	Certificado Digital	71
2.10.4.1	SSL	71
2.10.5	Firewall.....	72
2.11	Programação de computadores	72
2.11.1	Linguagens de Programação.....	72
2.11.1.1	Linguagem C	73
2.11.1.1.1	História	74
2.11.1.2	PHP (Personal Home Page)	74
2.11.1.2.1	PHP 7	75
2.11.1.3	HTML	75
2.11.1.3.1	Historia	75
2.11.1.3.2	Como funciona	76
2.12	Cascading Style Sheets (CSS).....	77
2.12.1	Definição	77
2.12.2	História	77
2.13	Câmera	77
2.13.1	Definição	77
2.13.2	História	78
2.14	Câmera IP	78
2.14.1	Definição	78
2.14.2	Funcionamento.....	78
3	MÉTODOS E PROCEDIMENTOS	79
3.1	Cronograma.....	79
3.2	Custo	79
3.3	Tipo de pesquisa	80
3.4	Caráter da pesquisa	80
3.5	Método de abordagem.....	80
3.6	Dados da pesquisa de campo.....	80
4	DESENVOLVIMENTO	85

4.1	Inserção do Modulo Wi-Fi	85
4.2	Instalação do virtualizador	85
4.3	Criação de uma Maquina Virtual.....	89
4.4	Instalação do Sistema Operacional	93
4.5	Instalação do LAMP-server.....	98
4.6	Criação do banco de dados no MySQL	99
4.7	Criação das páginas do servidor.....	102
4.7.1	Página de acesso (index.html).....	102
4.7.2	Página de controle do braço (site.php)	104
4.7.3	Página de gerencia de usuários (registros.php).....	106
4.7.4	Páginas para envio de comando ao servidor	108
4.8	Comunicação do servidor com o braço robótico	109
5	RESULTADOS E DISCUSSÃO.....	117
6	CONCLUSÃO	120
7	RECOMENDAÇÕES.....	122

1 INTRODUÇÃO

Segundo a revista Consultor Jurídico (2008), o funcionário de uma empresa teve seu braço amputado após um acidente com um moedor no dia 14 de maio de 2002. A 16ª Câmara Cível do Tribunal de Justiça de Minas Gerais condenou a empresa a pagar indenização e pensão mensal para o funcionário. A juíza Vilma Lúcia Gonçalves Carneiro, concedeu à vítima uma indenização de R\$ 28 mil por danos morais. O desembargador Otávio Portes, afirmou que a empresa tem a obrigação de fornecer condições seguras de trabalho para garantir a integridade física e proteção à saúde do empregado.

O Diretor de Novos Negócios da Pollux, Natanael Kaminski, afirmou no jornal Notícias do Dia (2015):

As indústrias vêm sofrendo com o alto custo da mão de obra, absenteísmo [ausência do funcionário no trabalho], rotatividade e buscam mais qualidade e precisão nos processos, além de estarem preocupadas com doenças que podem ser causadas por esforço repetitivo. Todos estes fatores estão fazendo com que elas busquem alternativas para automatização e por consequência, isso acaba impulsionando o nosso negócio.

Essas tecnologias vêm crescendo rapidamente com o processo da globalização, que para SOARES (2008), nem sempre os operadores de algumas dessas máquinas estão capacitados para mexer com estes novos equipamentos, causando ocorrências de acidentes de trabalho.

Os dados estatísticos de Acidentes de Trabalho de 2011 divulgados pelo Ministério da Previdência Social indicam, em comparação com os dos anos anteriores, um pequeno aumento no número de acidentes de trabalho registrados, o número total de acidentes de trabalho registrados no Brasil aumentou de 709.474 casos em 2010 para 711.164 em 2011, conforme pode ser visto na Figura 1.



Figura 1 Acidentes de Trabalhos Registrados de 2007 a 2011.

Fonte: <http://www.tst.jus.br/web/trabalhoseguro/dados-nacionais>

Acesso em: 22 de ago. 2016.

De acordo com SOARES (2008), os acidentes de trabalho acabam causando prejuízos a toda a sociedade, que para seus impostos, deixando de investir em educação, saúde preventiva ou lazer. A empresa tem perdas com o pagamento de indenizações, queda na produtividade durante o tempo de acomodação e assimilação da ocorrência, além de assumir gastos com hospital, medicamentos e apoio psicossocial. O Governo também tem perdas com o pagamento de pensões e aposentadorias precoces por invalidez.

1.1 OBJETIVO

1.1.1 Objetivos gerais

Aprimorar uma interface para controlar remotamente o protótipo de um braço robótico já existente, implementar em um servidor web para o acesso de qualquer lugar, e trabalhar em um sistema de segurança para o equipamento.

1.1.2 Objetivos específicos

Inserir um módulo de rede esp8266 em um arduino Mega 2560 que está ligado a um braço robótico já existente, adaptar uma interface gráfica para controlar o robô, implementar esta interface em um servidor web apache2, hospedado em um Ubuntu 16.04, para o acesso remoto do braço de qualquer lugar, criar um sistema de segurança para o servidor utilizando criptografia SSL, e banco de dados gerenciado pelo MySQL, para o registro de logs e de usuários com

permissão de acesso, e com isso fornecem mais segurança ao servidor e ao equipamento.

1.2 JUSTIFICATIVA

Tendo em vista esses desastres em trabalho e todas as suas consequências ao trabalhador, a empresa, a sociedade e a economia, este projeto visa minimizar os riscos que podem ser oferecidos a saúde física ou mental por alguns serviços, e consequentemente aumentando os lucros da empresa e a economia em geral, considerando que as instituições que se encaixam nessa situação deixariam de pagar indenizações e salários para funcionários que podem estar incapacitados devido a acidentes e não podem mais ser demitidos. Além disso, oferecer a facilidade no manuseio do equipamento, economizando tempo e dinheiro que seriam gastos com treinamento do funcionário.

2 FUNDAMENTAÇÃO TEÓRICA

2.1 MOTORES ELÉTRICOS

São máquinas destinadas a converter a energia elétrica em energia mecânica. Segundo Braga (2014), esses equipamentos possuem conjuntos de bobinas por onde a corrente elétrica percorre, criando assim um campo magnético, movimentando seu eixo de rotação.

2.1.1 História

Em 1886 o cientista alemão Werner von Siemens inventou o que pode ser considerado como a primeira máquina elétrica, pois foi o criador do primeiro gerador de corrente contínua auto induzido, mas essa criação só foi possível graças a pesquisas feitas até então por outros cientistas. Segundo Barreto (2009), em 1600 a obra intitulada de Magnete foi publicada em Londres pelo cientista inglês William Gilbert, descrevendo a força de atração magnética. O fenômeno da eletricidade estática já havia sido observado em 641 a.C., pelo grego Tales, ele verificou que ao atritar uma peça de âmbar com pano, ela adquiria a propriedade de atrair corpos leves, como penas, pelos, cinzas, etc.

No ano de 1663 foi construída pelo Alemão Otto Guericke a primeira máquina eletrostática, e aperfeiçoada pelo suíço Martin Planta em 1775.

Segundo Barreto (2009), ao fazer experiências com correntes elétricas em 1820, o físico dinamarquês Hans Christian Oersted, verificou que a agulha magnética de uma bússola era desviada de sua posição norte-sul quando passava perto de um condutor no qual circulava corrente elétrica. Esta observação feita por Hans permitiu a Oersted reconhecer a relação entre a eletricidade e o magnetismo, dando assim, o primeiro passo para o desenvolvimento do motor elétrico. O inglês William Sturgeon que estudava eletricidade nas horas de folga se baseou na descoberta de Oersted, e constatou, que um núcleo de ferro envolto por um fio condutor elétrico transformava-se em um ímã quando era aplicada uma corrente elétrica, observando também, em 1825, que a força do ímã era interrompida quando a corrente elétrica cessava. Era o começo da invenção do eletroímã, que seria de fundamental importância na construção de máquinas elétricas até hoje.

O cientista italiano S. Dal Negro construiu em 1832, a primeira máquina de corrente alternada com movimento de vaivém. O comutador foi

inventado no ano de 1833 pelo inglês W. Ritchie, construindo um pequeno motor elétrico onde em torno de um imã permanente, possuía um núcleo de ferro enrolado que ficava girando. A polaridade do eletroímã se alterava a cada meia volta através do comutador. O sucesso do motor elétrico só veio após ser desenvolvido em 1838 pelo arquiteto e professor de física Moritz Hermann von Jacobi, esse motor foi aplicado a um bote. Segundo Barreto (2009), alimentados por células de baterias, o bote navegou em uma velocidade de 4,8 quilômetros por hora e transportou 14 passageiros.

Depois de alguns anos, em 1886, a Siemens construiu um gerador sem a utilização de um imã permanente, provando assim que a tensão para o magnetismo poderia ser retirada do próprio enrolamento do rotor. Segundo Barreto (2009), o primeiro dínamo de Werner Siemens tinha uma potência de 30 watts e 1200 rotações por minuto. Essa máquina de Siemens não era somente como um gerador de eletricidade, pois podia operar como um motor, desde que a corrente elétrica fosse de corrente alternada para contínua.

No ano de 1879, na feira industrial de Berlim, foi apresentada a primeira locomotiva elétrica com uma potência de 2kW pela empresa Siemens & Halske.

Essa nova máquina apresentava vantagens em relação à máquina a vapor, a roda d'água e a força animal. Segundo Barreto (2009), entretanto, a fabricação dessa máquina possuía um elevado custo, e foi pensando nisso que muitos cientistas passaram seu tempo e atenção para o desenvolvimento de um motor elétrico mais robusto, barato e de menor custo de manutenção. Os que destacaram dentre esses pesquisadores foram o iugoslavo Nikola Tesla, o italiano Galileu Ferraris e o russo Michael von Dolivo-Dobrovolski. O esforço se cogitou em sistemas de corrente alternada, cujas vantagens já eram conhecidas em 1881. O engenheiro eletricista Galileu Ferraris, em 1885, construiu um motor de corrente alternada de duas fases. Ferraris, foi responsável também por ter inventado o motor de campo girante, e concluiu erroneamente que esses motores construídos com este princípio poderiam, no máximo, obter um rendimento de 50% em relação a potência consumida. Foi então que Tesla apresentou, um pequeno protótipo de motor de indução bifásico, em 1887, que continha um rotor em curto-circuito, mas também apresentou rendimento insatisfatório e suas pesquisas foram abandonadas.

Em 1889 o engenheiro Dobrowolsky, de Berlim, adquiriu a patente de um motor trifásico com rotor de gaiola. Esse motor tinha uma potência de 80 watts, um excelente conjugado de partida, e um rendimento aproximado de 80% em relação a potência consumida. Esse motor prometia uma construção mais simples, era mais silencioso, uma menor manutenção e alta segurança em operação. Em 1891, Dobrowolsky desenvolveu, nas potências de 0,4 a 7,5 kW, a primeira fabricação em série de motores assíncronos, segundo Barreto (2009).

2.2 ROBÓTICA

De acordo com OGATA (1998), o termo robótico se refere ao estudo e à utilização de robôs, e esse termo foi utilizado pela primeira vez pelo cientista americano e escritor, Isaac Asimov.

O ser humano sempre buscou meios para facilitar o seu serviço, pensando assim criaram muitas ferramentas e com a evolução da tecnologia foi desenvolvida a Robótica que é um ramo educacional e tecnológico que engloba computadores, robôs e computação, que trata de sistemas compostos por partes mecânicas automáticas e controladas por circuitos integrados, segundo FERREIRA (2002).

OGATA (1998) ainda afirma que a robótica tem como objetivo a automatização de tarefas que são executadas pelo homem. Ou seja, pode-se resumir que um robô é uma máquina que é capaz de realizar ações independentes, efetua uma tarefa programada, sem ser continuamente supervisionado por um operador humano.

Segundo HARRIS (2010), o termo robô é derivado da palavra checa “*robota*”, geralmente traduzida como “trabalho forçado”. Esta tradução condiz muito com a função da maioria dos robôs, muito deles são criados para trabalhos repetitivos e pesados em fábricas, ele pode fazer tarefas de grande, médio e pequeno porte, para trabalhos que geram perigo para os seres humanos.

2.2.1 Braço Robótico

HARRIS (2010) afirma que o tipo mais comum de robô é o braço robótico, que geralmente é formado por sete segmentos de metal e unido por seis junções. Um computador controla o robô através da rotação de um motor de passo, (alguns robôs maiores utilizam sistemas como hidráulico ou pneumático), diferente

dos motores comuns, os motores de passo se movem em incrementos exatos, isso o permite ser controlado com precisão, podendo repetir o movimento várias vezes seguidas, o robô pode utilizar sensores de movimento para monitorar se o movimento está sendo feito corretamente.

Um robô industrial com seis junções semelhantes a um braço humano. Ele tem o que equivale a um cotovelo, pulso e ombro. Em alguns casos, o ombro é montado sobre uma base estática em vez de um corpo móvel. Este tipo de robô possui seis graus de liberdade, ou seja, ele pode se mover em seis direções diferentes, já um braço humano, possui sete graus de liberdade, conforme HARRIS (2010).

A função de um braço humano é mover a sua mão de um lugar para o outro. Similarmente, a função de um braço robótico é mover um atuador de um lugar para o outro. Pode ser acoplar todo tipo de atuadores a um braço robótico. Cada atuador funciona para um tipo específico de trabalho. O atuador mais comum é uma versão semelhante a uma mão, que pode apanhar e carregar diferentes objetos. A mão robótica possui sensores de pressão acoplados, que informam ao computador a força com que o robô está segurando o objeto. Isso impede que o robô derrube ou quebre o que ele estiver carregando, segundo HARRIS (2010). Existem muitos outros atuadores como soldas, brocas e sprays de pintura.

De acordo com HARRIS (2010), os robôs industriais são feitos para fazer a mesma coisa repetidamente, em um ambiente controlado. Por exemplo, o robô pode operar tampando os potes de geleia em uma linha de montagem. Para ensinar um robô como fazer o seu trabalho, o programador guia o braço dele através de um controle. O robô memoriza a sequência exata de movimentos e os repete toda vez que uma nova unidade chega à linha de montagem.

A grande parte dos robôs industriais, para HARRIS (2010), trabalha em linhas de montagem de automóveis. Os robôs são mais precisos e podem fazer este trabalho de maneira muito mais eficaz que os homens. Eles sempre apertam os parafusos com a mesma força, não importa quantas horas tenham trabalhado. Estes robôs também são extremamente importantes para a indústria da informática. É necessário uma mão extremamente precisa para montar um minúsculo chip de computador.

O processo de fabricação do braço foi feito através de usinagem à laser das peças em acrílico, garantindo um melhor acabamento das peças e

diminuindo ao máximo o peso do conjunto. Foram utilizados parafusos para fixação dos eixos, dos motores e uma base em acrílico para fixar o conjunto do braço. Uma vez montado o braço, foi realizado os primeiros ajustes nas juntas e definido as posições dos servos motores nos eixos do robô, uma vez que o servo motor possui limitação de ângulo conforme pode ser visto na Figura 2.

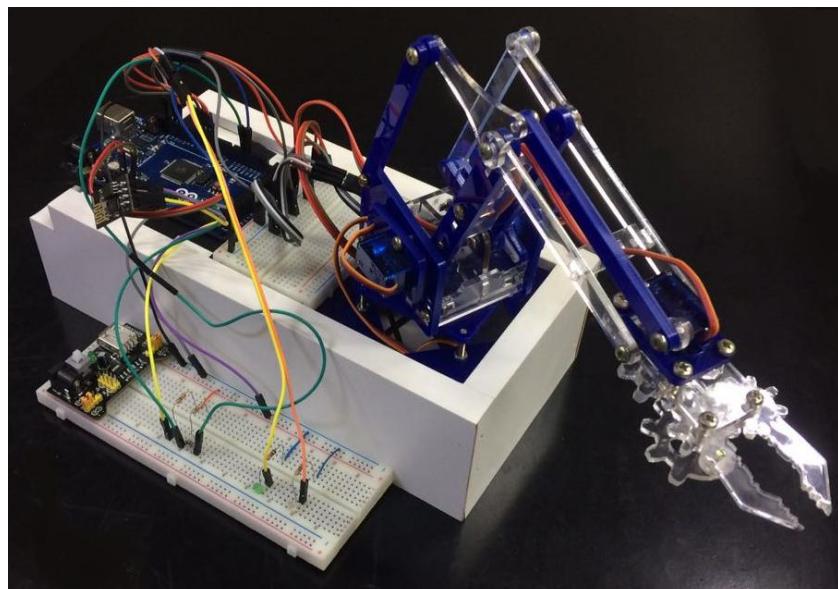


Figura 2 Braço Robótico.

2.2.2 Arduino

Segundo McROBERTS (2011) desde que o Arduino *Project* teve início, em 2005, mais de 150 mil placas foram vendidas ao redor de todo o mundo. O número de placas que não são oficiais supera o das placas oficiais, assim, estima-se que foram vendidas mais de 500 mil placas. Sua popularidade continua crescendo, e a cada dia mais pessoas percebem o grande potencial desse incrível projeto *open-source* para elaborar projetos interessantes de maneira rápida e fácil. A Figura 3 mostra um arduino Mega.

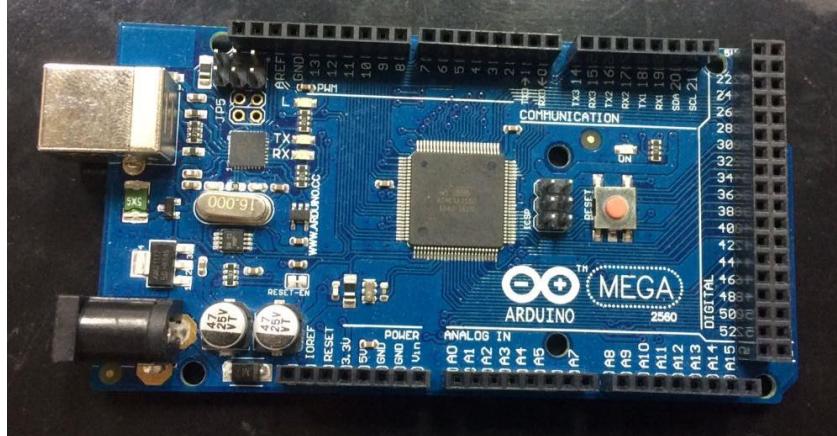


Figura 3 Arduino.

Ainda de acordo com McROBERTS (2011) a grande vantagem do arduino em relação a outras plataformas de desenvolvimento de micro controladores é a facilidade em sua utilização. Mesmo não sendo da área técnica, a pessoa pode aprender rapidamente o básico, criando seus próprios projetos em um tempo relativamente curto. Mais especificamente, artistas, oconsideram a melhor forma de criar suas obras de arte mais interativas, mesmo sem conhecimento específico em eletrônica. Existe uma grande quantidade de pessoas fazendo uso dos arduinos, compartilhando os seus códigos e esquemas de circuito para que outros tenham acesso, podendo modificar e melhorá-lo. Amaior parte dessa comunidade também está disposta a auxiliar outros desenvolvedores.

2.3 COMPUTADORES

Um computador é um dispositivo eletrônico que se destina a receber e processar dados para a realização de diversas operações tais como comunicação, desenvolvimento de conteúdo, busca de informações e centenas de outras possibilidades. Segundo Gomes 2005 tecnicamente, um computador é um conjunto de circuitos e componentes integrados que podem executar operações com rapidez, ordem e sistematização em função de uma série de aplicações práticas para o usuário programadas previamente.

2.3.1 História

2.3.1.1 O ábaco

Antigamente os cálculos eram feitos através de um ábaco, como o da Figura 4, esse equipamento foi a primeira máquina desenvolvida para cálculos e

era muito eficaz na resolução de problemas matemáticos. Ele é formado basicamente por um conjunto de varetas paralelas uma a outra e bolas que realizam a contagem segundo Gugik (2009). A data de seu primeiro registro é datada do ano de cinco mil e quinhentos a.C.

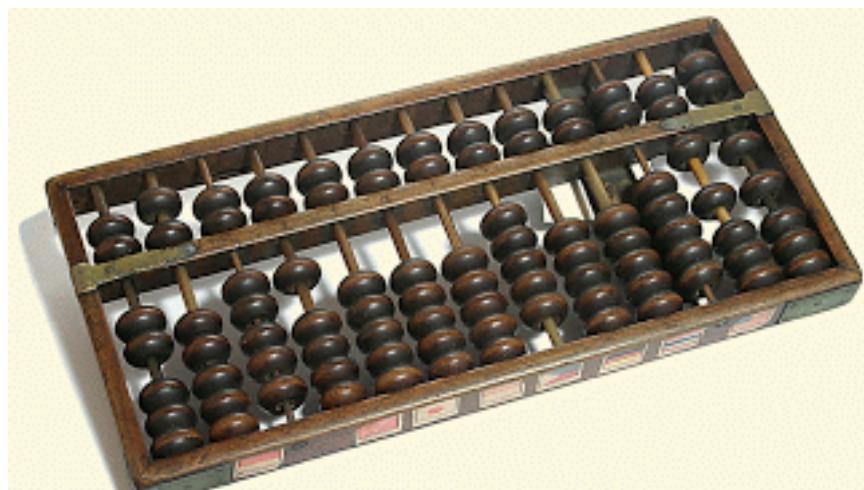


Figura 4 Ábaco.

Fonte: <http://wycellosweb.blogspot.com.br/2013/01/abaco.html>
Acesso em: 22 de ago. 2016.

2.3.1.2 Réguas de Cálculo

Em 1636 d.C., segundo Gugik (2009), um padre inglês que se chamava William Oughtred, viu a necessidade de aperfeiçoar o ábaco e criou um equipamento que foi chamado de régua de cálculo, vista na Figura 5, que tinha o objetivo de realizar multiplicações complexas. Esse equipamento consistia em uma régua que possuía valores pré-calculados e eles eram organizados de modo que os resultados eram acessados automaticamente.

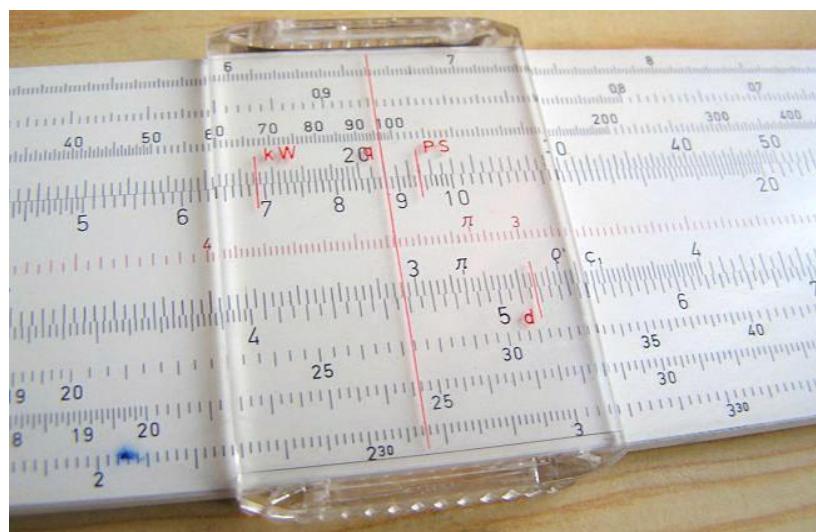


Figura 5Régua de Cálculo.

Fonte: <http://www.tecmundo.com.br/tecnologia-da-informacao/1697-a-historia-dos-computadores-e-da-computacao.htm>
Acesso em: 22 de ago. 2016.

2.3.1.3 Máquina de Pascal

Em 1642, um matemático francês chamado Bleise Pascal desenvolveu a primeira calculadora mecânica da história, mostrada na Figura 6. Esse equipamento continha rodas interligadas que giravam para a realização dos cálculos. Pascal queria desenvolver esse equipamento para realizar as quatro operações básicas, o que não aconteceu, pois era capaz apenas de subtrair e somar. Após alguns anos o alemão Gottfried Leibnitz conseguiu criar uma calculadora que efetuava a soma, divisão e a raiz quadrada segundo Gugik 2009.



Figura 6Máquina de Pascal.

Fonte: <http://www.tecmundo.com.br/tecnologia-da-informacao/1697-a-historia-dos-computadores-e-da-computacao.htm>
Acesso em: 22 de ago. 2016.

2.3.1.4 Programação Funcional

Em todos esses equipamentos citados acima, as suas operações já estavam previamente programadas, impedindo a inserção de novas funções. Segundo Gugik 2009, em 1801, o costureiro Joseph Marie Jacquard construiu uma máquina programável que tinha como objetivo recortar os tecidos de forma automática, esse mecanismo foi chamado de tear programável. A partir desse momento muitos esquemas foram com base nesse projeto.

2.3.1.5 Máquina analítica

Aproveitando os conceitos do tear programável, em 1837, Charles Babbage criou uma máquina capaz de calcular funções diversas como trigonometria e logaritmos, segundo Gugik 2009, a precisão desse equipamento chegava a 50 casas decimais.

2.3.1.6 Teoria de Boole

Segundo Gugik (2009), George Boole desenvolveu, em 1847, um sistema lógico que reduzia a representação de valores em dois algarismos: 0 ou 1. Nessa sua teoria ele utilizou o número 1 como a representação do ativo, verdadeiro e existente, enquanto o número 0 representava o inverso, que era inativo, falso e inexistente. Essa teoria é utilizada até hoje pelos computadores modernos.

2.3.1.7 Computador Mecânico

Segundo Alves (2014), o engenheiro mecânico Charles Babbage foi considerado o primeiro inventor de um computador mecânico, ainda no século XIX, esse computador era usado para criar efeitos sonoros no cinema. Na década de 1940, os computadores mecânicos já não estavam sendo mais utilizados e deram lugar aos computadores de uso geral, esses computadores utilizavam algoritmos simples para perfurar cartões e obter cálculos complexos. Nesse contexto a hp apresentou o Oscilador de Áudio HP200A, mostrado na Figura 7, que foi utilizado pela Disney para a produção de efeitos sonoros.



Figura 7Oscilador HP200A.

Fonte: Reprodução/Creative Commons
[\(<http://www.techtudo.com.br/noticias/noticia/2014/08/dia-da-informatica-confira-historia-do-computador-e-sua-evolucao.html>\)](http://www.techtudo.com.br/noticias/noticia/2014/08/dia-da-informatica-confira-historia-do-computador-e-sua-evolucao.html)

Acesso em: 18 de ago. 2016.

2.3.1.8 A primeira geração

A primeira geração é denominada por computadores a válvula, sendo o Eniac (*Electronic Numerical Integrator and Computer*, Figura 8), um dos primeiros a representar esse tipo de computador. Segundo Alves (2014), esse computador foi criado nos Estados Unidos, durante a segunda guerra mundial, no ano de 1946 por John Eckert e John Mauchley. Esse equipamento foi considerado o primeiro computador digital eletrônico programável de uso geral. Suas válvulas tinham que ser trocadas diariamente devido ao seu tempo de duração que raramente passavam de um dia, sua capacidade de operação era menor a de uma calculadora moderna, esse equipamento pesava cerca de trinta toneladas, o seu hardware contava com setenta mil resistores e dezoito mil válvulas de vácuo que em funcionamento consumia duzentos mil watts de energia. Seu orçamento foi de US\$ quinhentos mil dólares na época, o que representaria US\$ seis milhões de dólares atualmente.



Figura 8Computador Eniac.

Fonte: <http://www.computerhistory.org/revolution/birth-of-the-computer/4/78>
Acesso em: 18 de ago. 2016.

Após o Eniac, a geração de computadores ainda passa por diversos modelos como o Harvard Mark I mostrado na Figura 9, em 1944, que foi utilizado pelos nazistas, o SSEC, que pode ser visto na Figura 10, foi lançado pela IBM em 1948, e era capaz de calcular a posição da lua usando seletiva eletrônica, e foi este

o equipamento a traçar a rota da missão Apollo 11, em 1969, segundo Alves (2014).

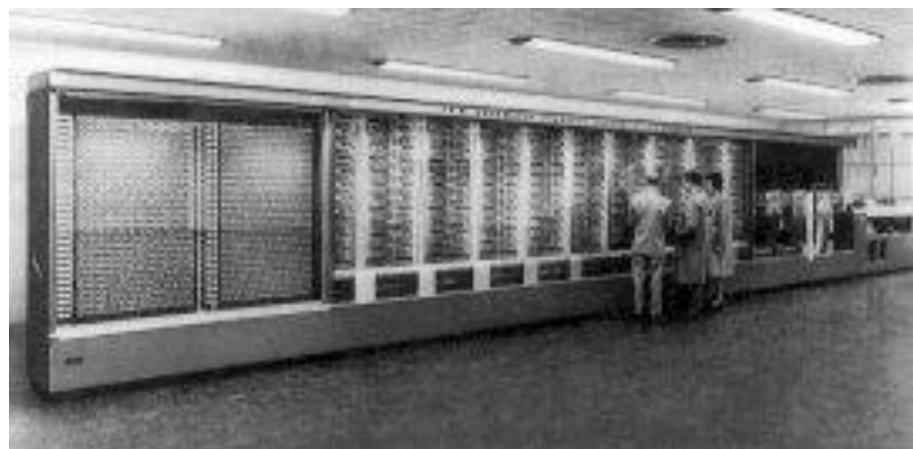


Figura 9Computador Harvard MARK I.

Fonte: <http://piano.dsi.uminho.pt/museuv/1946hmark1.html>

Acesso em: 18 de ago. 2016.



Figura 10Computador IBM SSEC.

Fonte: <http://www.columbia.edu/cu/computinghistory/ssec.html>

Acesso em: 18 de ago. 2016.

2.3.1.9 A segunda geração

O grande salto da primeira geração para a segunda geração foi marcada pela substituição de válvulas eletrônicas por transístores, o que diminuiu drasticamente o tamanho do hardware. Nesse mesmo período foi criada a tecnologia de circuitos impressos, evitando que fios e cabos ficassem espalhados. Segundo Gugik (2009), o IBM 7030, apresentado na Figura 11, na segunda

geração, foi o primeiro supercomputador a ser lançado. Esse computador executava cálculos complexos em microssegundos, o que permitia até um milhão de operações por segundo.



Figura 11 IBM 7030.

Fonte: <http://www.tecmundo.com.br/tecnologia-da-informacao/1697-a-historia-dos-computadores-e-da-computacao.htm>
Acesso em: 22 de ago. 2016.

2.3.1.10 A terceira geração

A terceira geração segundo Gugik (2009), foi marcada pela criação dos circuitos integrados, que era um dispositivo que continha vários circuitos em conjunto no mesmo lugar, o que tornou as máquinas mais velozes e baratas. Um dos principais exemplos dessa geração é o IBM 360/91 que foi lançado em 1967, a Figura 12 mostra como era a máquina.



Figura 12IBM 360/91.

Fonte: <http://www.tecmundo.com.br/tecnologia-da-informacao/1697-a-historia-dos-computadores-e-da-computacao.htm>
Acesso em: 22 de ago. 2016.

2.3.1.11 A quarta geração

A quarta geração é conhecida pela criação dos microprocessadores e computadores pessoais, com essa nova geração os computadores tiveram uma redução drástica do seu tamanho e preço. Agora, atingir o patamar de bilhões de operações por segundo não era mais impossível graças as suas CPUs. Circuitos integrados ficaram ainda menores. Nessa época, os softwares se tornaram tão importantes quanto o hardware. Segundo Gugik (2009), o Altair 8800 que pode ser visto na Figura 13, lançado em 1975, foi o computador que revolucionou tudo o que era conhecido, e com o seu tamanho reduzido, cabia facilmente em uma mesa de escritório. Apesar de sua funcionalidade, o Altair 8800 não era fácil de ser utilizado, foi então que Steve Jobs, em 1976, lançou o primeiro computador pessoal chamado Apple I, visto na Figura 14. Com o sucesso de vendas, os computadores pessoais foram se tornando cada vez mais populares e potentes, permanecendo até os tempos de hoje.

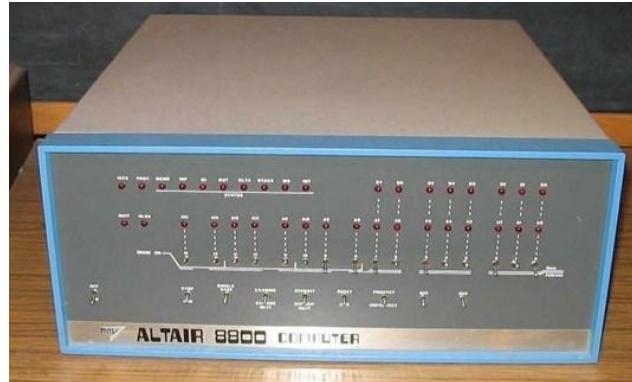


Figura 13Altair 8800IBM 360/91.

Fonte: <http://www.tecmundo.com.br/tecnologia-da-informacao/1697-a-historia-dos-computadores-e-da-computacao.htm>
Acesso em: 22 de ago. 2016.



Figura 14Apple I.

Fonte: <http://apple2history.org/history/ah02/>
Acesso em: 22 de ago. 2016.

2.4 SISTEMA OPERACIONAL

Conforme PEREIRA (2015), Sistema Operacional, do inglês, *Operating System* (OS,) é um programa ou um conjunto de programas que tem a função de gerenciar os recursos do sistema, fornecendo então uma interface que permite a comunicação entre o computador e o usuário, como mostra a Figura 15.

Figura 15 Esquema de Sistema Operacional.

Um sistema operacional pode ser executado ao ligar a máquina, porém, na maioria dos casos, segundo PEREIRA (2015), ele é executado através de um outro programa armazenado em uma memória não-volátil ROM (ou seja, oferece dados apenas para a leitura, e é capaz de armazenar informações mesmo sem energia) nomeada de BIOS, em um processo chamado de "*bootstrapping*" (processos autossustentáveis, que podem prosseguir sem qualquer ajuda externa). Depois que a máquina e seus componentes são ligados, a BIOS procura pelo OS em uma unidade de armazenamento (pode ser definida uma sequência de unidades pelo usuário, assim, caso a BIOS não encontre o OS em uma, ela buscara na próxima unidade) e o inicia, e então o Sistema Operacional passa a controlar a máquina.

2.4.1 Criação do sistema operacional

Um dos primeiros sistemas operacionais a ser desenvolvido foi o CTSS e também um dos primeiros a adotar a técnica de *timesharing*. Essa técnica, permite que vários usuários, ao mesmo tempo, possam utilizar um ambiente para executar programas. Isso ocorre sobre o mesmo sistema operacional, rodando em uma máquina. Segundo Mota Filho (2012), esse tipo de sistema é responsável pelo compartilhamento de processo, processador, memória e disco entre vários

utilizadores. Ele funcionaria de forma que as fatias de tempo do processador, conhecidas como *time slice*, eram destinadas aos programas carregados em memória. Assim, cada programa seria individual na máquina por algumas frações de segundo, dando assim a impressão de que tudo está funcionando ao mesmo tempo.

O CTSS foi desenvolvido por Fernando Jose Corbató no Centro de Computação do MIT. A primeira vez que ele apareceu funcional, foi em 1961, onde foi utilizado um IBM 709, que pode ser visto na Figura 16.



Figura 16 IBM 709.

Fonte: Mark Bartelt (<http://www.cacr.caltech.edu/~mark/IBM709.html>).
Acesso em: 24 de nov. 2016.

2.4.2 Projeto MAC (MIT Project MAC)

Em novembro de 1962, um ex-integrante do MIT, Joseph Carl Robnett Licklider, propôs o projeto MAC e o projeto foi aceito.

Segundo Mota Filho 2012, o projeto MAC foi criado com dois objetivos: um sistema operacional avançado e um laboratório de inteligência artificial. Por esse motivo, a sigla do projeto foi tratada com dois nomes diferentes: *Multiple Access Computers* e *Man And Computers*.

O *Multiple Access Computers* iria desenvolver o sistema operacional Multics (*MULTIplexed Information and Computing Service*) que seria superior ao CTSS segundo Mota Filho 2012.

O objetivo do Multics, era um sistema operacional com suporte para memória virtual, segmentação de memória e recursos de paginação. Segundo Mota

Filho 2012, isso possibilitaria um sofisticado processo de transferência de dados entre discos e memória.

Após muita pesquisa, o Multics foi apresentado em uma sessão especial na *Fall Joint Computer Conference*, em 1965. Depois de algum tempo melhorias, o Multics foi disponibilizado para a comercialização em 1969. Segundo Mota Filho 2012, em julho de 1985 o desenvolvimento do Multics foi cancelado e com isso várias organizações que começaram a suspender o seu uso.

2.4.3 Unix

O Unix é um Sistema Operacional multiusuário e multitarefa, permitindo diversos usuários e várias tarefas sendo executadas simultaneamente. O OS é também um programa de *run level multi-camadas*, ou seja, o usuário pode usar sua interface gráfica, ou por linhas de comando livremente.

Ele pode ser dividido internamente em duas partes, seu núcleo (Kernel), e um interpretador de comandos Shell, que executa os comandos digitados pelo usuário.

2.4.3.1 Unics

Segundo Mota Filho (2012) em 1969, uma empresa que estava auxiliando o desenvolvimento do Multics, a Bell Labs, fornecendo recursos e emprestando alguns de seus desenvolvedores ao MIT, abandonou o projeto. Entretanto isso não impediu que alguns dos integrantes do projeto do Multics mantivessem contato com profissionais da Bell Labs. Motivados pelo Multics dois programadores da empresa, que estavam anteriormente no MIT ajudando no desenvolvimento do Multics, Dennis Ritchie e Kenneth Thompson começaram a desenvolver um projeto pessoal chamado de Unics ou *UNiplexed Information and Computing Service*.

O Unics, de acordo com Mota Filho (2012), tinha por objetivo ser um Multics modesto como o nome sugeria. Ele deveria fazer, um sistema operacional simples, modesto e versátil, deveria funcionar utilizando o sistema de *time sharing* e ter portabilidade entre computadores de todos os tamanhos. Após uma sugestão de Brian Kernighan o nome Unics foi trocado para Unix.

2.4.3.2 Projeto do Unix

Inicialmente o projeto de Ritchie e Thompson foi rejeitado pela Bell Labs, devido ao fato de necessitar a obtenção de um computador de médio porte para o desenvolvimento do sistema operacional. Então eles traçaram novamente um projeto só que mais detalhado que necessitava de um investimento inicial e precisava de uma máquina que poderia ser alugada, apesar disso a ideia outra vez foi rejeitada pela empresa, conforme MOTA FILHO (2012).

Mota Filho 2012 acrescenta que mesmo após as rejeições, Thompson e Ritchie não queriam desistir da ideia, se juntaram a Ruddy Cannaday, outro integrante da Bell Labs e no papel e quadro negro começaram a esboçar o projeto do sistema operacional começando pelo *filesystem* e após isso o *kernel*. Thompson achou um computador velho para o início do projeto era um PDP-7, da DEC, mostrada na Figura 17, que estava não estava sendo usada e Thompson conseguiu a transferência dela para poder utilizá-la.



Figura 17 Unidade PDP-7.

Fonte: *System photographs* (<http://simh.trailing-edge.com/photos.html>).
Acesso em: 28 de ago. 2016.

Em meados de 1969 segundo Mota Filho (2012), Kenneth Thompson iniciou a implementação do *filesystem*, criado por Dennis Ritchie. Thompson dividiu o desenvolvimento do projeto em quatro partes: sistema operacional, ambiente *shell*, editor de texto e compilação do sistema e dos programas. Trabalhando inicialmente com as partes mais relevantes do sistema operacional Thompson

desenvolveu aplicações em nível de usuário para edição de arquivos. Foi para o desenvolvimento do ambiente *shell*, com isso o sistema começava a tomar uma forma.

O PDP-7 em 1970 estava ficando obsoleto, então foi solicitado a compra de uma nova máquina o PDP-11. Em dezembro de 1970 se deu início a migração do Unix para o PDP-11, visto na Figura 18. Em novembro de 1971 a primeira versão do Unix foi lançada ainda que internamente na Bell Labs, escrito em uma linguagem de alto nível, a linguagem B, segundo MOTA FILHO(2012).



Figura 18 Thompson (de pé) e Ritchie programando no PDP-11 em um terminal teletipo.

Fonte: <http://www.amimjf.org/public/pdp11.html>

Acesso em: 28 de ago. 2016.

2.4.3.3 O novo Unix

Mota Filho (2012) expõem que após o surgimento da linguagem C, uma evolução da linguagem B, o Unix precisou ser reescrito e isso significava começar tudo do zero. Refazer tudo foi um processo lento iniciado em 1972, que envolvia compreender como executar rotinas auxiliares e a dificuldade de criar estruturas de dados, uma vez que a linguagem não possuía esse recurso, o que levou Thompson a desistir do Unix. Ritchie por sua vez, continuou o projeto, melhorou a linguagem C, adicionou estruturas e aprimorou o compilador.

Mota Filho (2012) acrescenta que em 1973, Ritchie e Thompson fecharam um acordo de cooperação e refizeram todo o sistema, uma grande inovação do novo Unix foi a implementação dos *pipes*, que permitem os programadores amarrar vários processos e gerar uma saída, redirecionando a saída de um programa para outro.

2.4.3.4 Unix nas universidades

Em 1976 Kenneth Thompson tirou uma licença de seis meses da Bell Labs para dar aulas na Universidade de Berkeley, no Estado da Califórnia. Lá ele desenvolveu o que viria a ser a versão 6 do Unix, voltado para universidades. O Unix se disseminou rapidamente pelas universidades norte americanas. Mesmo após a volta de Thompson para a Bell Labs, professores e alunos continuaram a desenvolver o Unix. Na época o código do Unix era aberto e permitia mudanças de acordo com uma licença criada para este fim. Com isso surgiu o *Berkley Software Distribution* (BSD), um Unix adaptado para o ambiente acadêmico, de acordo com MOTA FILHO(2012).

2.4.3.5 Comercialização do Unix

Com a velocidade que o Unix se espalhou pelas universidades estava mais que claro que ele seria uma ótima fonte de renda conforme Mota Filho (2012). Inicialmente a ideia foi criar programas para o Unix para uso comercial. Em 1984, a AT&T, empresa controlada pelo Bell Labs, criou uma subsidiária independente a AT&T Computer Systems. Uma empresa totalmente nova que possuía recursos próprios que era controlada pela AT&T, que podia gerir novos negócios assim como a comercialização do Unix.

2.4.4 Projeto GNU

O projeto GNU foi criado por Richard Stallman, que em busca de um sistema operacional livre, se demitiu do MIT e iniciou o projeto com o objetivo de criar um sistema operacional igual ao Unix, porém livre, isto é, seu código-fonte sempre estará disponível para a comunidade de forma gratuita. O nome gnu dado ao projeto se refere ao animal de mesmo nome, assim como o trocadilho “*Gnu’s Not Unix*” o que significa que o projeto GNU sempre será livre ao contrário de alguns softwares que eram livres e não o são mais. O projeto GNU refere-se a diversos aplicativos livres, para compor um sistema operacional livre, segundo MOTA FILHO(2012). O símbolo do projeto é a desenho da cabeça de um gnu como na Figura 19.

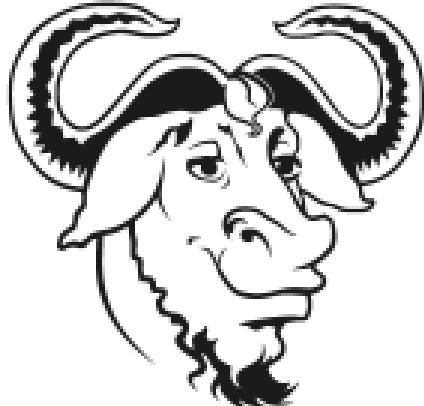


Figura 19 Caricatura de um Gnu, símbolo do Projeto GNU.

Fonte: *The GNU Operating System* (<http://www.gnu.org/>).

Acesso em: 28 de ago. 2016.

Mota Filho (2012) denota, que no início do projeto Stallman ouviu falar de um compilador multi linguagens, o *Free University Compiler Kit*, e pediu para o autor se ele podia utilizar o compilador em seu projeto. Richard recebeu uma resposta zombada do autor dizendo que a universidade era “free”, mas o compilador não. Isso fez com que Stallman começa seu projeto pelo compilador desenvolvendo o GCC (GNU C Compiler). Nesse meio tempo também criou também o editor de texto GNU Emacs, muito utilizado até hoje. O GNU Emacs rodava com perfeição no Unix o que fez com que algumas pessoas pedissem a permissão para usá-lo, então Stallman disponibilizou-o em um servidor ftp público do MIT.

2.4.4.1 Free Software Foundation

Em 1985 uma organização chamada *Free Software Foundation* ou FSF, criada por Richard Stallman, segundo Mota Filho (2012), tinha como objetivo arrecadar fundos para a manutenção do projeto GNU. Essa organização é mantida até os dias de hoje, possuindo empregados que desenvolvem e mantêm vários pacotes e programas do sistema GNU.

2.4.4.2 Definição de Software Livre

A expressão “Software Livre” significa que o código-fonte do software é distribuído juntamente com o software. A definição de Software Livre ou “*free software*” não está ligada a software gratuito, mas sim a software livre. Um software livre obrigatoriamente deve ter seu código fonte disponível, podendo ser alterado e redistribuído. Entretanto ele pode ser distribuído de diversas formas,

necessariamente um software livre não precisa ser gratuito. O seu custo pode estar contido na embalagem, na mídia onde ela será gravada. É possível também cobrar por suporte técnico ou manutenção do software, conforme a Free Software Foundation (2016).

O Free Software Foundation (2016) salienta que uma das premissas mais importantes de um software livre é que se você comprar um software que seja livre e disponibilizar uma cópia dela na internet isso não será um caso de pirataria. De acordo com Richard Stallman, o criador da definição Software livre, afirma que o software livre nos proporciona:

- A liberdade de executar o programa como você desejar, para qualquer propósito;
- A liberdade de estudar como o programa funciona, e adaptá-lo às suas necessidades. Para tanto, acesso ao código-fonte é um pré-requisito.
- A liberdade de redistribuir cópias de modo que você possa ajudar ao próximo.
- A liberdade de distribuir cópias de suas versões modificadas a outros. Desta forma, você pode dar a toda comunidade a chance de beneficiar de suas mudanças. Para tanto, acesso ao código-fonte é um pré-requisito.

2.4.5 GNU/Linux

De acordo com Ferreira (2010), o GNU/Linux é um sistema operacional criado por Linus Benedict Tovards para ser igual ao Unix, porém distribuído de forma gratuita. A ideia para o desenvolvimento do *kernel* Linux foi inspirada no Minix, um S.O. *unix-like* disponível muito usado na época.

Linus Tinha como desafio desenvolver um *kernel* clone do Unix com memória virtual, capacidade de multiusuários e multitarefa. Após algum tempo de trabalho, conseguiu criar um *kernel* capaz de executar os comandos padrões clonados do Unix pelo projeto GNU, de acordo com FERREIRA(2010).

2.4.5.1 Distribuições GNU/Linux

Segundo Mota Filho (2012), uma distribuição Linux é junção de um *kernel* Linux, diversos aplicativos do Free Software Foundation projeto GNU ou de

terceiros e de um conjunto compiladores. Em suma, uma distribuição é um pacote com o *kernel* e vários programas que são compatíveis com ele.

Existem distribuições ditas puras, são aquelas que não foram baseadas em nenhuma distribuição anterior. Uma distribuição pode também ser derivada de outra, isto é, a ela foi criada a partir de uma distribuição já existente e herdou suas características, segundo Mota Filho (2012).

2.4.5.1.1 Debian

Segundo Software in Public Interest (2016) o Projeto Debian é uma distribuição GNU/Linux pura, criada com o objetivo de produzir um sistema operacional compostos totalmente de softwares livres. O produto desse projeto atualmente inclui a distribuição do *kernel* Linux com softwares do Projeto GNU e muitos outros pacotes e aplicações livres que juntos formam a distribuição Debian GNU/Linux.

O Debian é uma distribuição que desde o início estava aberta à contribuição do trabalho de desenvolvedores e utilizadores. Ela é uma das distribuições Linux mais significantes disponíveis. Além disso ela é uma distribuição “micro empacotada”, isto é, ela utiliza informações de dependências detalhada relativamente a relações inter-pacotes para assegurar a consistência do sistema quando há alguma atualização, segundo Software in Public Interest (2016).

O projeto adotou um grande conjunto de políticas e procedimentos para empacotar e entregar o software com o objetivo de manter a qualidade do produto final de acordo com Software in Public Interest (2016).

O logo de uso aberto do Debian mostrado na Figura 20, sob a licença *Copyright (c) 1999 Software in Public in the Interest, Inc.* e são distribuídas sob a licença do *GNU Lesser General Public Licence*, de acordo com a Software in Public Interest (2016).



debian

Figura 20 Logo do Projeto Debian.

Fonte: Wikimedia Commons (<https://commons.wikimedia.org/wiki/File:Debian-OpenLogo.svg>).

Acesso em: 07 de set. 2016.

a) História

Segundo DebianDocumentationTeam (2015) o projeto Debian foi criado em 1993 por Ian Murdock, quando estudava na Universidade de Pardue. De novembro de 1994 à novembro de 1995 o projeto foi patrocinado pelo Projeto GNU da *Free software Foundation*.

Em 1994 foi gasto uma grande parcela do tempo para organizar o Projeto Debian para que a comunidade pudesse contribuir mais ativamente, assim como para trabalhar no desenvolvimento do dpkg, o comando básico utilizado para lidar com os pacotes do Debian no sistema de acordo com DebianDocumentationTeam (2015).

Iam Murdock juntamente com Bruce Perens, iniciaram várias facetas importantes como a Orientação de Software Livre do Debian e o Contrato social do Debian, e a iniciação do Projeto de Hardware aberto, durante esse tempo o Debian vinha ganhando espaço no mercado e uma reputação como plataforma de utilizadores sérios do Linux, conforme DebianDocumentationTeam(2015).

Segundo DebianDocumentationTeam (2015) em 16 de julho de 1997, foi criada a Software in Public Interest, Inc. (SPI), destinada originalmente para fornecer uma entidade legal que pudesse receber doações para o Projeto Debian, com o tempo essa organização cresceu e começou a suportar outros projetos de softwares livre que não estavam ligados ao projeto Debian.

Atualmente o Debian está na sua versão 8.5 como pode ser visto na figura 21.

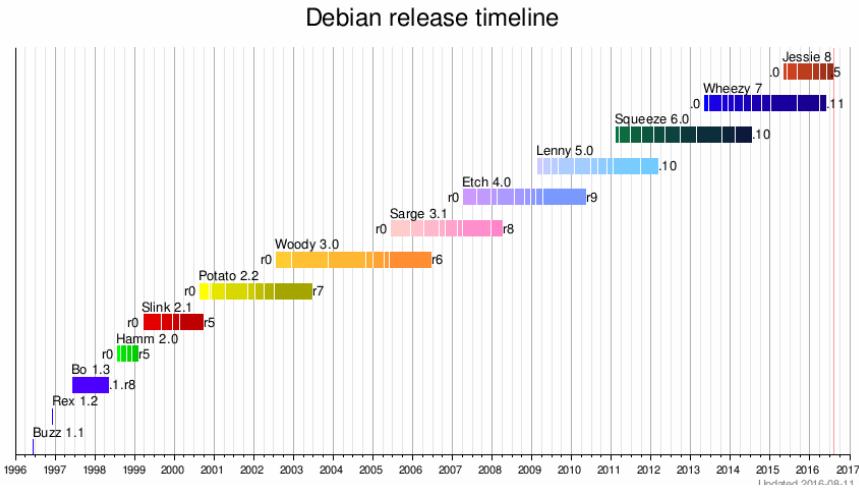


Figura 21 Lançamentos do Debian.

Fonte: Wikipédia (https://en.wikipedia.org/wiki/List_of_Debian_releases).

Acesso em: 07 de set. 2016.

b) Sistema de empacotamento Debian

O sistema de empacotamento do Debian é capaz de resolver as dependências e conflitos com o sistema dos pacotes a serem instalados, além de facilitar a instalação e atualização dos pacotes com o “*apt-get*”, afirma o DebianDocumentationTeam(2015).

c) Releases estável, teste e instável

Segundo Mota Filho (2012). O Debian possui três modalidades básicas para os separar os pacotes, chamado formalmente de releases: a estável (*stable*), teste (*testing*) e a instável (*unstable*). Existem outras, porém menos utilizadas por usuários comuns como a experimental.

O Release *unstable*, recebe os pacotes mais recentes dos programas não testados pela equipe do Debian, é onde os desenvolvedores mais trabalham. O *testing* o release onde os pacotes vão quando já passaram por um teste e revisão na *unstable*, nessa modalidade os pacotes são atualizados constantemente pelos desenvolvedores, mediante a um feedback fornecido pelos usuários. Após diversas atualizações correção de bugs e de falhas de segurança os pacotes são enviados para a *stable*, nessa modalidade os pacotes somente sofrem alterações caso seja descoberta uma falha na segurança ou *bug* de acordo com Mota Filho (2012).

2.4.5.1.2 Ubuntu

O Ubuntu é um sistema operacional de código aberto baseado no sistema GNU/Linux Debian, ele é distribuído para computadores pessoais, smartphones e para servidores. O desenvolvimento do Ubuntu é conduzido pela multinacional sediada na Inglaterra a Canonical Ltd, uma companhia do Sul Africano Mark Shuttleworth. A Canonical consegue gerar lucro a partir do suporte técnico e serviços relacionados com o Ubuntu. O projeto Ubuntu é desenvolvido abertamente com a premissa de open-source software development, ou seja, as pessoas são encorajadas a usar o software livre, estuda-lo, melhora-lo e redistribui-lo conforme Canonical (2016).

a) Origem da palavra “Ubuntu”

Segundo Canonical (2016) Ubuntu é uma palavra antiga da África que quer dizer ‘humanidade para os outros’. Ela também significa ‘Eu sou o que sou por causa de quem todos nós somos’. O objetivo do Sistema Operacional Ubuntu é levar esse espirito para o mundo dos computadores.

O logotipo do Ubuntu, Figura 22, reforça a ideia de união ele é chamado de “*circle of friends*” ou círculo de amigos.



Figura 22 Logo do Ubuntu.
Fonte: Ubuntu (<http://design.ubuntu.com/brand/ubuntu-logo>).
Acesso em: 07 de set. 2016.

b) Como tudo começou

No ano de 2004 o GNU/Linux já estava estabelecido como uma plataforma de servidor corporativo, entretanto o *free software* não era uma

realidade para o dia a dia da maior parte dos computadores desktop. Por isso Mark Shuttleworth juntou um grupo de desenvolvedores de um dos maiores projetos de desenvolvimento, o projeto Debian, e começou a criar um Linux desktop *easy-to-use* (fácil de usar) o Ubuntu (Canonical, 2016).

c) Lançamentos do Ubuntu

Os lançamentos do Ubuntu são feitos de seis em seis meses e a cada dois anos é publicado uma versão LST (*long-term support* ou suporte de longo prazo), geralmente essa versão é usada para adicionar uma mudança de grande escala no Sistema Operacional e tem suporte por 5 anos a partir da data do seu lançamento. O lançamento de novas versões agendadas foi uma grande inovação criada pelo grupo de desenvolvimento do Ubuntu. Ao contrário de outras distribuições comerciais o Ubuntu possui apenas uma versão, ele se foca em criar uma versão tanto comercial quanto para comunidade de alta qualidade para todos os seus usuários, além de oferecer suporte e atualizações gratuitas, salienta Canonical (2016). A linha do tempo das distribuições pode ser vista na Figura 23.

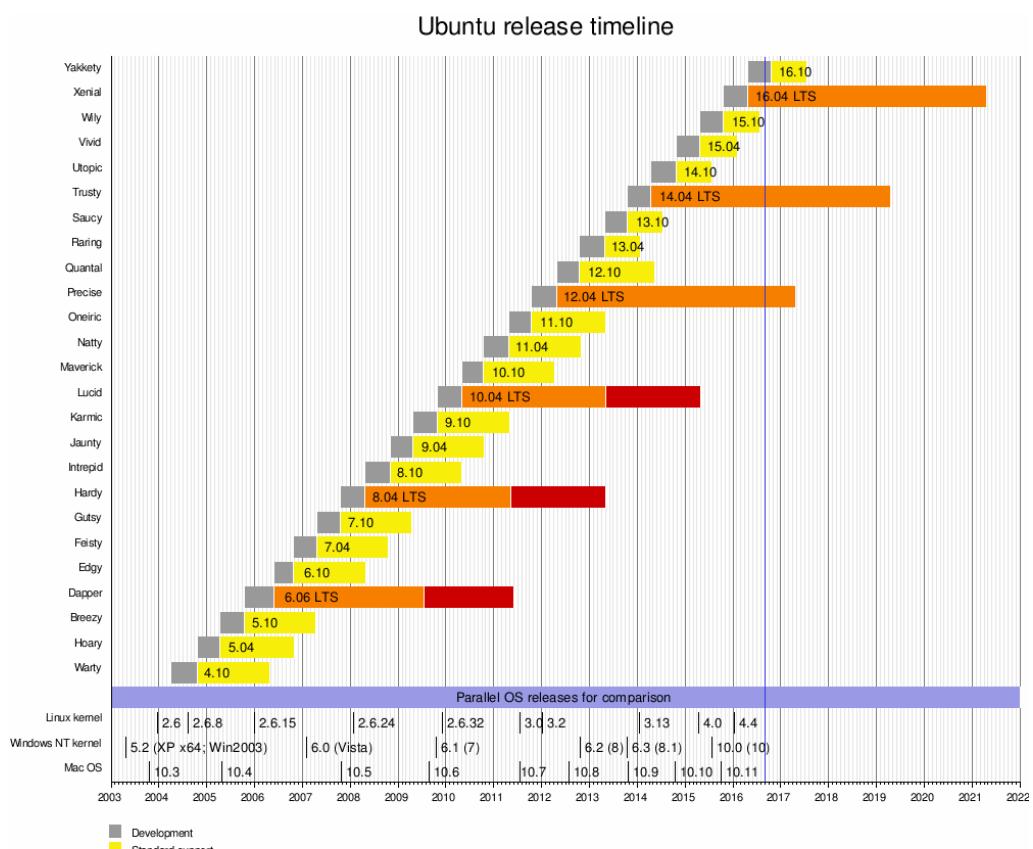


Figura 23 Lançamentos do Ubuntu.
Fonte: Wikipédia (https://en.wikipedia.org/wiki/List_of_Ubuntu_releases).
Acesso em: 07 de set. 2016.

O primeiro lançamento oficial do Ubuntu foi a versão 4.10, com codinome '*Warty Warthog*', em português Facóquero Enverrugado, foi lançado em outubro de 2004, e despertando o interesse de muitos, fazendo com que muitos entusiastas do software livre e pessoas experientes se juntassem a comunidade do Ubuntu (Canonical, 2016).

De acordo com Canonical (2016), apesar do desenvolvimento do Ubuntu ser coordenado pela Canonical, a governança do Ubuntu é de certo modo independente uma vez que ele possui líderes voluntários ao redor do mundo que tomam responsabilidades de muitos elementos críticos do projeto. Graças à colaboração entre os usuários e voluntários juntamente com a Canonical e várias outras empresas o Ubuntu se tornou uma das maiores distribuições Linux se tornando uma das melhores plataformas disponível.

d) Ubuntu atualmente

Atualmente o Ubuntu possui nove '*Flavors*', que Ubuntus com diferentes pacotes conjuntos de pacotes instalados. Entretanto todos eles usam os mesmos repositórios de pacotes, então é possível instalar qualquer pacote disponível no repositório independente do '*flavor*', conforme Canonical (2016).

Além disso o Ubuntu tem versões especiais para servidores, OpenStack Clouds, e para dispositivos móveis. Todas as edições possuem as mesmas infraestruturas de software. Nos dias atuais o Ubuntu é uma das distribuições com mais usuários devido ao seu suporte tanto da comunidade quanto da equipe de desenvolvedores, acrescenta Canonical (2016).

2.4.6 Windows

Windows do inglês, significa "Janela". Ele é um sistema multitarefa, multiusuário e multiplataforma, ou seja, permite execução de mais de uma tarefa ao simultaneamente, permite vários usuários e é capaz de rodar em diversas plataformas como INTEL, DEC Alpha, MIPS e entre outros.

O Windows é também um programa de código fechado, ou seja, os usuários não têm acesso ao seu código fonte, além de ser pago. Além disso, o OS prende o usuário ao *run level* gráfico, sendo extremamente raro a necessidade de usar as linhas de comando, além de funcionalidades mais vastas.

As linguagens usadas no desenvolvimento do Windows foram C, C++ e Assembly.

2.4.6.1 História

Em 1981 segundo Harada 2014, a Microsoft (empresa responsável pelo desenvolvimento do software), iniciou o desenvolvimento de um Gerenciador de Interface, que foi chamado de Windows 95, Figura 24, essa interface foi a primeira que possibilitava o uso do mouse em um ambiente gráfico, porém, antes desta versão a Microsoft já havia desenvolvido alguns sistemas operacionais, mas eles não continham uma interface gráfica.

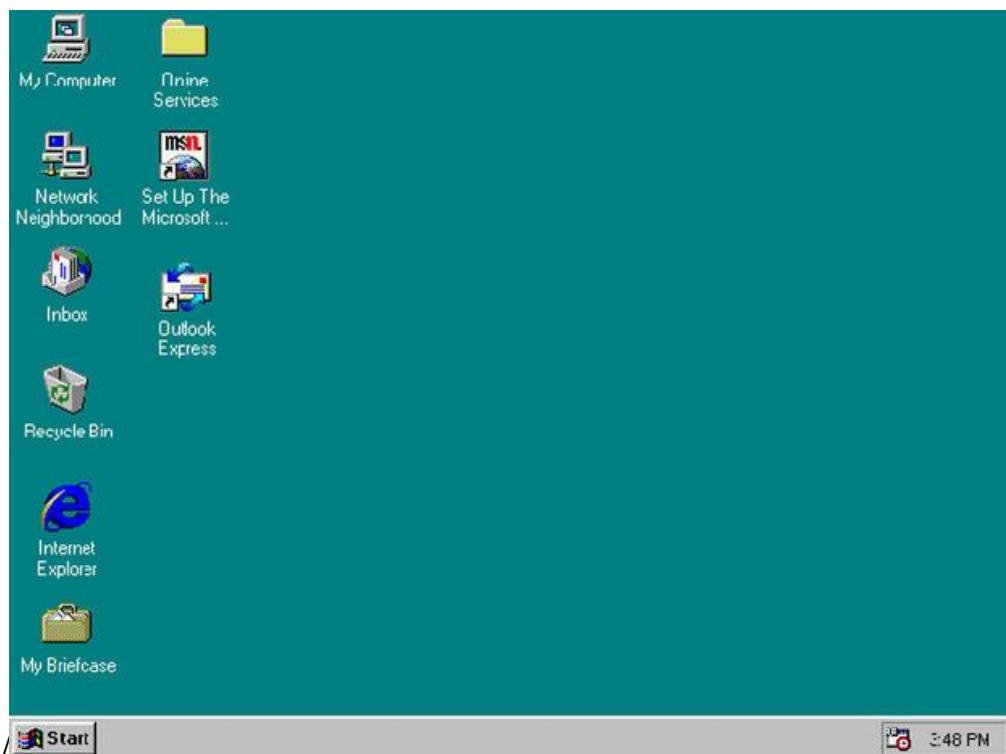


Figura 24 Tela Inicial do Windows 95.

Fonte: (<http://www.techrepublic.com/article/windows-95-is-reborn-in-a-browser/>)
Acesso em: 13 de set. 2016.

As primeiras versões do Windows começaram a ser vendidas em 1983, e eram acessadas através de quatro disquetes. Esse sistema utilizava apenas 1 Mb do HD (hard disk ou disco rígido). Com o sucesso de vendas desse sistema a Microsoft continuou a desenvolver sistemas operacionais e se tornou uma das maiores empresas nesse ramo. Após o lançamento do Windows 95 foi lançado em 1993 o Windows NT, Figura 25, que trazia a funcionalidade de trabalhar com um servidor de arquivos.



Figura 25 Tela inicial do Windows NT.
Fonte: <http://www.guidebookgallery.org/screenshots/winnt40serv>
Acesso em: 13 de set. 2016.

Em 1998, o Windows 98, Figura 26, veio para substituir o Windows 95, e foram vários lançamentos até o maior sucesso da empresa em 2001, que foi o Windows mostrado na figura XP Figura 27, esse sistema trazia diversas melhorias e foram vendidas mais de quatrocentos milhões de cópias até janeiro de 2006.

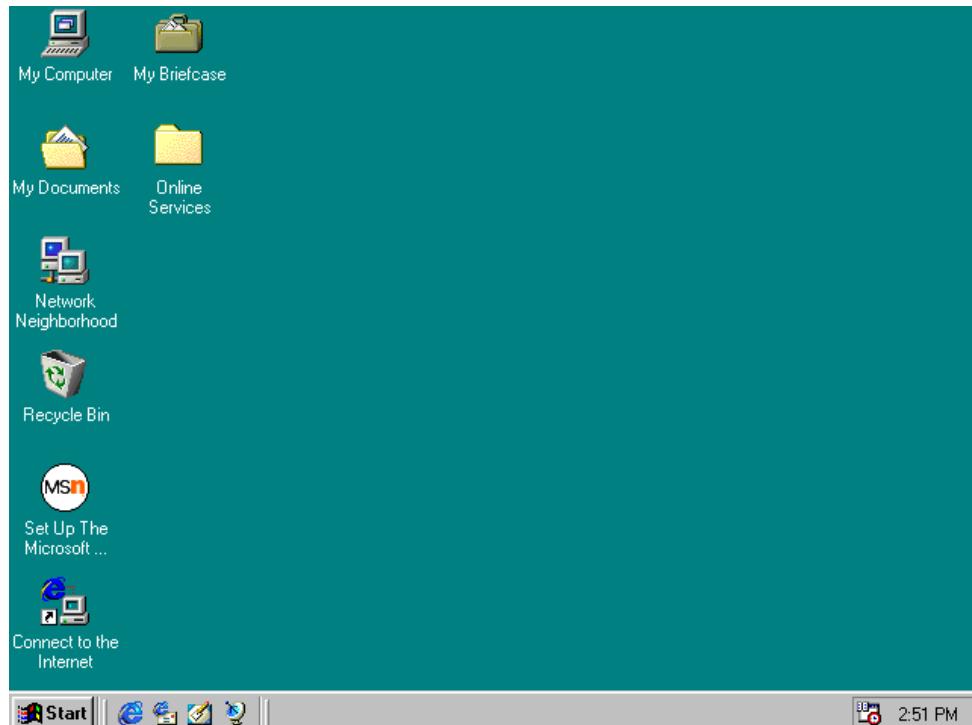


Figura 26 Tela Inicial do Windows 98.
Fonte: <http://www.guidebookgallery.org/screenshots/win98>
Acesso em: 13 de set. 2016.



Figura 27 Área de trabalho do Windows XP.
Fonte: <http://www.techtudo.com.br/noticias/noticia/2014/01/suporte-a-windows-xp-e-office-2003-termina-em-abril-saiba-o-que-muda.html>
Acesso em: 13 de set. 2016.

O Windows 7, mostrado na Figura 28, foi anunciado em 2009, se tornando um dos melhores e mais completos da categoria. O próximo sistema de sucesso e com muitas melhorias foi o Windows 10, sendo o último lançamento até os dias de hoje que pode ser visto na Figura 29.



Figura 28Área de Trabalho do Windows 7.

Fonte: <http://www.vftutoriais.net/2015/01/01/windows-7-aio-sp1-novembro2014/>
Acesso em: 13 de set. 2016.



Figura 29Área de Trabalho do Windows 10.

Fonte: https://www.microsoftstore.com/store/msbr/pt_BR/pdp/Windows-10-Pro/productID.320406200
Acesso em: 13 de set. 2016.

2.5 INTERFACE DO UTILIZADOR

De acordo com ARRUDA (2011) a Interface do Utilizador surgiu na Quarta Geração de computares e é o que permite a interação do usuário com a máquina ou programa, possibilitando a entrada de comandos e a visualização da resposta retornada, as interfaces do utilizador estão evoluindo, dês da primeira interface até a última. Elas vieram para facilitar determinadas tarefas ou mesmo tornar a interface mais agradável. Antes disso, a resposta ao usuário era dada através de roldanas, ou agulhas que apontavam para o resultado, impressão ou até mesmo sons.

2.5.1 Interface de Linha de Comando (CLI)

Segundo ARRUDA (2011) As Primeiras Interfaces do Utilizador foram as interfaces de linha de comando, chamadas de "CLI" (*Command Line Interface* ou Interface de Linha de comando), como mostra a Figura 30.

```

Debian GNU/Linux 7 debsvr-matriz tty1
debsvr-matriz login: root
Password:
Last login: Mon Aug 22 15:31:21 BRT 2016 on tty1
[REDACTED]
Linux debsvr-matriz 3.2.0-4-amd64 #1 SMP Debian 3.2.46-1 x86_64
-----
Redes de Computadores
Prof. Kleber (KGe)
SOR-II | ADMSSI
-----
root@debsvr-matriz:~# 
root@debsvr-matriz:~# _
```

Figura 30 Interface de Linha de Comando (CLI).

A Interface de Linha de comando é baseada em Texto, e possuem uma série de comandos e parâmetros que devem ser inseridos pelo usuário para instruir a máquina a realizar determinada ação, afirma ARRUDA (2011).

Esta Interface embora seja a mais antiga, ainda é muito usada em servidores, por ser extremamente leve e responsiva.

2.5.2 Interface tipo Texto para Usuário (TUI)

Para ARRUDA (2011) a Interface conhecida como "TUI" (*Text User Interface* do inglês, Interface de Texto para Usuário), diferente da Interface de linha de comando, já exibe os comandos que podem ser executados, e o usuário precisa apenas pressionar a tecla referente a opção desejada. Como apresenta a Figura 31.

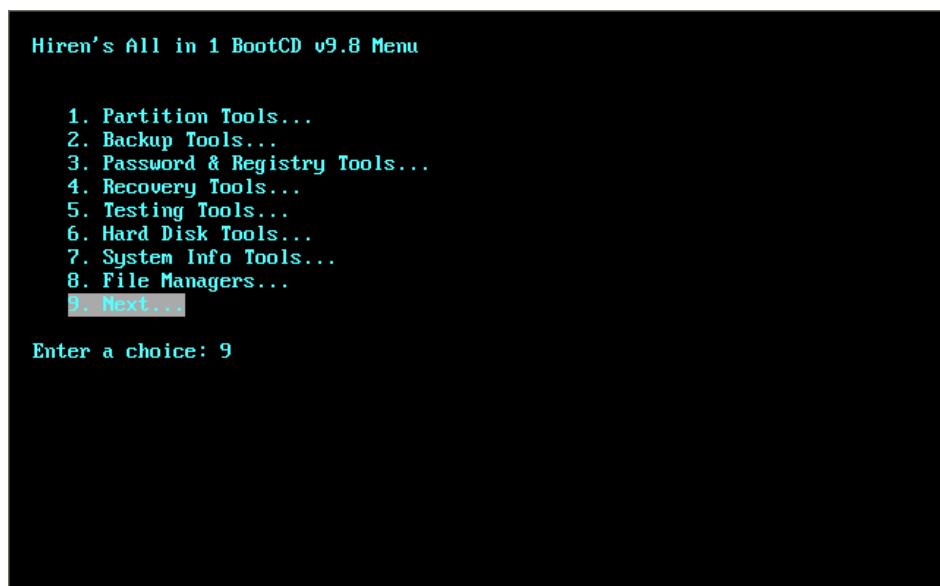


Figura 31 Interface de Texto para Usuário (TUI).

2.5.3 Interface Gráfica para Usuário (GUI)

Na década de 1960, surgiu a *Grafic User Interface* (GUI) ou Interface Gráfica para Usuário exemplificada na Figura 32, já possui um design mais trabalhado, designando ícones, animações e outros elementos que visam a facilidade aos usuários além de um ambiente mais agradável.

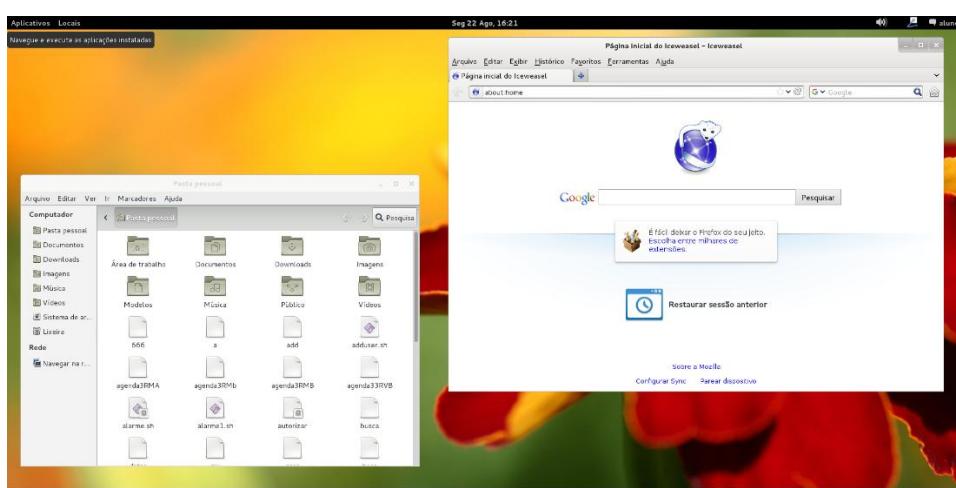


Figura 32 Interface Gráfica para Usuário (GUI).

A Interface já permite o uso do mouse que segundo ARRUDA (2011) foi desenvolvido pelo engenheiro elétrico Douglas Engelbart e sua equipe, inspirados pelo trabalho de Vannevar Bush.

2.6 VIRTUALIZAÇÃO

Segundo Vmware 2016, o processo de criar uma representação baseada em software e virtual de alguma coisa, ao invés de um processo físico, é chamado de virtualização. A virtualização pode ser aplicada em diversas áreas como aplicativos, servidores, armazenamento e redes, esse processo possibilita reduzir as despesas em um projeto e ao mesmo tempo aumentar a eficiência e agilidade em todos os pontos.

A virtualização utiliza um programa (software) para simular a existência do hardware e criar assim um sistema de computadores virtual.

2.6.1 Máquina Virtual

Um sistema de computadores virtual é chamado de máquina virtual (VM): um software isolado que contém um sistema operacional e aplicativos, e cada VM é completamente independente segundo Vmware 2016.

2.6.1.1 Virtual box

Virtual Box é um produto de virtualização gratuito e de código aberto voltado tanto para a empresa, quanto uso doméstico. Esse programa é responsável por criar as VMs (Máquinas Virtuais) em sua máquina real. Ele está disponível nas plataformas Windows, Linux, Macintosh e os hosts Solaris e será usado durante o desenvolvimento deste projeto segundo Virtual box 2016.

2.7 INTERNET

A Definição para internet encontrada no dicionário Houaiss é: "A *internet* é uma rede de computadores dispersos por todo o planeta que trocam dados e mensagens utilizando um protocolo comum."

2.7.1 Historia

Segundo DUMAS (2011), O Início da Internet veio na década de 1960, durante a Guerra Fria, com o objetivo de criar elos de comunicação entre diferentes centros militares, e que pudesse aguentar a uma destruição parcial, graças ao Departamento de Defesa americano.

Nesta época, o cientista e pesquisador Paul Barn, considerado um dos pioneiros da internet, desenvolveu um conjunto baseado em um sistema descentralizado, uma rede projetada como uma teia de aranha, onde os dados buscam o melhor caminho para seu destino. De acordo com DUMAS, essa tecnologia recebeu o nome de “Troca de Pacotes”, *packet switchin*, em inglês.

Em 1957, a Advanced Research Project Agency (ARPA) foi fundada pelo presidente Eisenhower. Em 1969 a ARPAnet já funcionava, e a princípio, conectaria as universidades de Santa Barbara, Los Angeles e Stanford. Nos anos seguintes, surgiram o correio eletrônico, criado pelo engenheiro americano Ray Tomlinson, e um aplicativo que possibilitava a troca de e-mails de forma ordenada, desenvolvido por Lawrence G. Roberts, tornando as mensagens eletrônicas o elemento mais utilizado na rede, segundo DUMAS (2011).

A ARPAnet foi a primeira rede com mais de 15 nós. A parte da comunicação militar foi separada e isolada desta rede, e passou a ser chamada de MILnet.

Outras redes que conectavam diversos institutos de pesquisa foram criadas nos Estados Unidos, França e Grã-Bretanha, a única coisa que ainda faltava era uma linguagem comum para todas, problema que foi resolvido em 1974 por Robert Kahn e Vint Cerf, criadores do protocolo TCP/IP. Esta Tecnologia foi adotada pela ARPAnet em 1976. Pouco tempo depois as primeiras sobrecargas na rede aconteceram devido ao grande número de usuários. Em 1986 foi lançada uma nova rede pela National Science Foundation, e 4 anos depois a ARPAnet se juntou a ela. E assim a internet atinge seu primeiro milhar de computadores conectados, de acordo com DUMAS (2011).

2.7.2 Servidores

De acordo com Eriberto (2012) Servidor é uma máquina ou programa que oferece serviços aos clientes dentro de uma rede. Os servidores surgiram com

desenvolvimento das redes de computadores, que devido ao crescimento, veio a necessidade de dedicar algumas máquinas para fornecer funcionalidades a outras, a essas funcionalidades dá-se o nome de "serviços", eles podem ser desde um compartilhamento de dados e softwares até o compartilhamento de recursos físicos como a memória e o processamento. A máquina ou programa que solicita um serviço dá-se o nome de "cliente", o cliente pode estar tanto no próprio equipamento onde o servidor está operando como pode estar em um dispositivo com acesso à rede do servidor.

Segundo McKie (1995) o sistema cliente-servidor (Figura 33), é o mais utilizado pelo modelo de solicitação-resposta, o servidor é um dos componentes desse sistema. O modelo descreve a relação de programas numa aplicação, o seu funcionamento começa com a solicitação de um cliente para um servidor, o qual efetua algumas ações e envia uma resposta ao cliente em questão, essa resposta geralmente é um resultado ou reconhecimento.

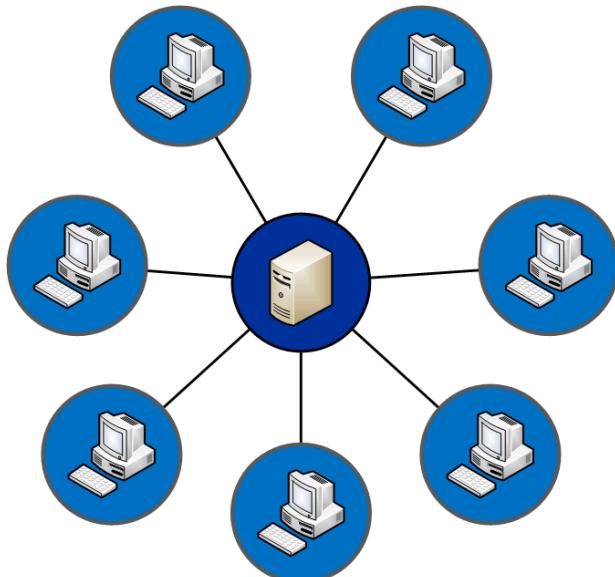


Figura 33 Sistema Cliente-Servidor.

A maior parte da internet é baseada no modelo cliente-servidor. Roteadores, servidores-raiz, DNSs (Domain Name Server) direcionam o tráfego pela internet. Existem milhares de servidores conectados à internet, funcionando ao redor do mundo, praticamente todo o acesso de um usuário comum na internet passa por um ou mais servidores segundo Albitz e Liu (1998).

2.7.2.1 Servidores WEB

Segundo Ljubuncic (2011) servidores webs são servidores que tem a função de armazenar e dar acesso a páginas escritas em HTML, PHP, Java e entre outras linguagens voltadas para a WWW, através do protocolo HTTP ou HTTPS.

Servidor web é um sistema de computador que processa solicitações HTTP, o protocolo básico de internet usado para distribuir informação no *World Wide Web*. O termo "servidor web" pode tanto se referir ao hardware quanto os softwares usados para gerenciar e aceitar as requisições HTTP, esse termo pode também apontar para o sistema inteiro (LJUBUNCIC, 2011).

Segundo Aulds (2001) a principal função de um servidor web é guardar, processar e fornecer as páginas web para clientes. As páginas entregadas ao cliente geralmente são documentos HTML, que podem conter imagens, *scripts* e *styles sheets*. Outra função desses servidores é a de receber algum tipo de conteúdo dos clientes, que podem ser dados em forma de texto ou arquivos. Essa propriedade pode ser implementada por meio da submissão de *web forms*.

Conforme Aulds (2001) um *web browser* ou *web crawler*, inicia a comunicação com o servidor solicitando um recurso usando o HTTP, então o servidor responde com a página requisitada ou uma página de erro caso não seja possível acessar o endereço em questão. Esse recurso pode ser um arquivo armazenado na memória secundária do servidor ou uma página gerada pelo servidor de acordo com a requisição feita ao servidor.

Alguns dos servidores web suportam o *server-side scripting*, usando *scripting languages* (linguagens de extensão) como o ASP (*Active Server Pages*), PHP (*Hypertext Preprocessor*), Perl ou outras linguagens. Isso significa que é possível adicionar algumas funcionalidades ao servidor sem alterar o *software* do servidor por meio de scripts. Essa função na geralmente é usada para gerar páginas HTML dinâmicas, ao contrário de retornar uma página estática ao cliente, segundo Aulds (2001).

Um servidor web não precisa necessariamente estar conectado à internet, ele pode estar trabalhando em uma rede local. Esse servidor pode estar embutido em dispositivos como impressoras, roteadores, câmeras IP ou webcams.

O que significa que nenhum software além de um navegador web é necessário no lado do cliente de acordo com Aulds(2001).

2.7.2.1.1 Apache

Criado por Rob McCool em 1995, o Apache é uma ferramenta de servidores web, que segundo a Netcraft é um programa usado por cerca de 50% dos sites ativos em 2015 conforme a Figura 34.

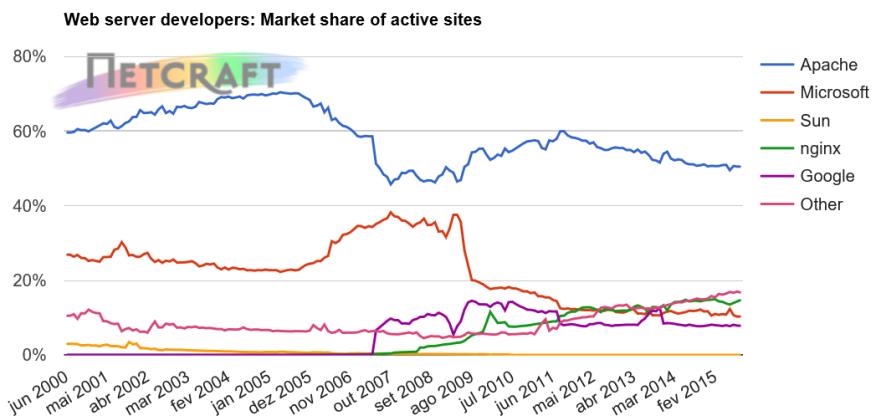


Figura 34 Desenvolvedores de Servidor Web (Sites Ativos).

Fonte: <https://news.netcraft.com/archives/2015/09/16/september-2015-web-server-survey.html>
Acesso em: 24 de ago. 2016.

Toda vez que você acessa um site, uma requisição é enviada ao servidor em que o site está hospedado, o servidor é responsável por responder essas requisições com algum serviço. Um servidor é um computador disponibilizado em uma rede com o objetivo de prover serviços para outros hosts segundo Moura (2011).

O servidor web apache funciona na estrutura cliente-servidor, o cliente envia uma requisição ao servidor, e o servidor responde ao cliente em codificação HTML. Segundo Moura (2011), ao receber uma solicitação php o servidor aciona o interpretador PHP que processa essas solicitações, como acessar o banco de dados e sistema de arquivos.

2.7.2.1.2 World Wide Web (WWW)

A *World Wide Web*, popularmente chamada de "WWW", surgiu na década de 1990, quando a internet começou a se popularizar e alcançar a população em geral.

Ela possibilita o uso de interfaces gráficas na internet, permitindo a criações de sites dinâmicos e acessíveis para qual quer um.

2.7.2.1.3 Páginas estáticas e dinâmicas

As páginas web estáticas mostram apenas a informação exata contida no documento quando alguém as visitam. As páginas estáticas não precisam ser compostas por apenas textos, elas podem conter também vários tipos de arquivos, estilos e até vídeos. Entretanto, os visitantes da página visualizaram sempre o mesmo texto, imagens ou vídeos até que o código da página seja alterado.

As páginas web dinâmicas ao contrário das páginas estáticas são capazes de mostrar conteúdos diferentes de acordo com o usuário que acessa partindo de um mesmo código fonte. O site pode exibir diferentes páginas de acordo com o sistema operacional, o navegador web ou o equipamento utilizado pelo visitante.

A criação de páginas dinâmicas tem um nível de dificuldade maior se comparado as páginas estáticas, além de seu carregamento ser mais lento. Porém as páginas dinâmicas podem facilitar a atualização de dados em sites que possuem muitas páginas e uma grande quantidade de dados, elas podem ser usadas também, para melhorar a aparência das páginas, além de serem utilizadas na autenticação de usuários em sites. Existem dois tipos de páginas dinâmicas, aquelas que são processadas pelo cliente e o que são processadas pelo servidor.

As páginas dinâmicas processadas pelo cliente são utilizadas para criar efeitos especiais para web na apresentação da página como *rollovers*, controle de janelas, mover objetos pela página, controle de formulário ou cálculos. O código que cria esses efeitos se encontram no próprio corpo da página HTML, o navegador se encarrega de interpretar e executar os efeitos e funcionalidades. A sua principal desvantagem é a dependência do sistema onde estão executando uma vez que o processo para geração dos efeitos se dá na máquina do cliente, a vantagem é que ele diminui a carga sobre o servidor e oferece respostas imediatas a ações do usuário.

As páginas processadas pelo servidor podem fazer todo o tipo de aplicação web, como agendas, fóruns, sistemas de documentação, estatísticas, jogos, chats, entre outros. Ela é muito útil quando é necessário acessar informações centralizadas, em um banco de dados dentro do servidor. Quando um

cliente faz uma requisição de página para o servidor, este executa alguns scripts que geram a página resultado, que possui apenas o código HTML. Nesse tipo de página dinâmica apenas o servidor é quem maneja os bancos de dados e as informações enviadas pelo cliente como imagens, texto ou outros recursos, processa e então envia uma resposta com todas as operações para o cliente. O carregamento dessas páginas é mais lento uma vez que o servidor precisa processar a requisição da página e gerar uma resposta, o que pode também custar recursos de hardware do servidor web.

2.8 WI-FI

Segundo PIXININE (2015), a internet sem fio surgiu, no final da década de 90, quando os computadores começaram a conquistar o público. Hoje, o Wi-Fi é extremamente popular e está em diversos aparelhos eletrônicos como computadores, smartphones e *video games*.

Para PIXININE (2015), Wi-Fi significa “*Wireless Fidelity*”, nome que foi dado à expressão “Hi-Fi” (*High Fidelity*), usada pela indústria fonográfica nos anos 50. O termo Wi-Fi foi registrado pela Wi-Fi Alliance fundada pelas empresas 3com, Nokia, Lucent Technologies e Symbol Technologies, mas com sua popularidade qualquer tecnologia que utiliza Wi-Fi foi nomeada como WLAN (*Wireless Local Area Network*). Logo da empresa Wi-Fi Alliance na figura 35.



Figura 35 Logo da Empresa Wi-Fi Alliance.

Fonte: Wi-Fi Alliance (http://www.wi-fi.org/sites/default/files/public/default_images/WFA_Alliance_Flat_Web_LR.png)
Acesso em: 15 de set. 2016.

A rede Wi-Fi, de acordo com PIXININE (2015), usa ondas de rádio para se comunicar pela internet, essa onda é transmitida por meio de um adaptador, o aparelho Wi-Fi que recebe a onda, decodifica e emite a partir de uma antena, os sinais de internet emitidos por uma rede podem chegar via cabo, linha telefônica ou onda de rádio como por exemplo o 3G, para que um dispositivo tenha

acesso a esses sinais é preciso que o aparelho esteja em um determinado raio de ação.

2.9 BANCO DE DADOS

Um banco de dados, para DATE (1999) é simplesmente um sistema de armazenamento computadorizado, como um armário de arquivos eletrônico, que permite o usuário a busca e atualização dessas informações sempre que solicitado.

2.9.1 Bancos de Dados Relacionais

Segundo DIAS (2007) o Sistema Gerenciador de Banco de Dados Relacional (SGBDR) é um software que permite controlar armazenamento, exclusão, recuperação, integridade e segurança das informações em um banco de dados. O que o torna Relacional é o armazenamento dos dados feitos em forma de tabelas que são organizadas em colunas, e cada coluna armazena um tipo de dado, consequentemente, cada linha registra uma "categoria". Exemplificando, uma tabela de *Clientes* possui as colunas com *Numero*, *Nome* e *Sobrenome*, uma linha na tabela ficaria algo parecido com [001, "Joao", "Santos"].

Naturalmente as tabelas possuem chaves, ou seja, uma (ou mais) colunas que identificam uma linha na tabela. Para facilitar a navegação, são definidos índices, quem permitem a busca de dados em uma ou mais colunas na tabela.

2.9.1.1 SQL (Structured Query Language)

De acordo com DATE (1999) SQL é a linguagem padrão usada por maioria dos bancos de dados relacionais. A tradução para a sigla SQL é "Linguagem de consulta estruturada".

A Linguagem foi desenvolvida na IBM *Research*, no começo da década de 1970. O protótipo da IBM, System R, foi o primeiro a ter a linguagem implementada em grande escala, e depois foi adotada por diversos outros produtos comerciais da IBM e muitos outros fornecedores, segundo DATE (1999).

Para DATE (1999), o CRUD são as funcionalidades básicas de um SGBDR. CRUD é a sigla para *Create*, *Read*, *Update* e *Delete*, que se referem as opções Inserção, Leitura, Atualização e Exclusão.

2.9.1.2 MySQL

O MySQL é um Sistema de Gerenciamento de Banco de Dados Relacional (SGBDR), que tem como interface a linguagem SQL.

O fator que popularizou tanto o MySQL (Figura 36) é sua integração com o PHP, e hoje ele é um dos SGBDRs mais usados atualmente, sendo usado por grandes empresas como: Google, NASA, Sony, Nokia, Cisco Systems, HP e entre outros.



Figura 36 Logo do MySQL.

Fonte: <https://en.wikipedia.org/wiki/File:MySQL.svg>

Acesso em: 24 de nov. 2016.

2.10 SEGURANÇA DE REDES

De acordo com o CERT (Centro de Estudos, Respostas e Tratamento) a segurança de redes é umas das partes mais importantes para uma rede de internet, que hoje a internet está presente nos computadores, celulares, televisões e sistema de áudio.

2.10.1 Políticas de segurança

Para o CERT (Centro de Estudos, Respostas e Tratamento), política de segurança é o que define os direitos e as responsabilidades de cada um em relação a segurança dos recursos computacionais que utiliza, e as penalidades às quais está sujeito, caso não a cumpra.

2.10.2 Criptografia

Segundo MEDEIROS (2015) criptografia é a junção de duas palavras gregas *κρυπτός* (*kríptós* – secreto, escondido) e *γράφειν* (*gráfein* – escrita).

Criptografia é um método de segurança que transforma um arquivo legível em algo ilegível, assim só o destinatário que irá receber o arquivo poderá entendê-lo.

Atualmente grande parte dos dispositivos e aplicativos que usem a internet utilizam criptografia como meio de segurança, algumas vantagens de usar a criptografia são:

- Proteger dados pessoais armazenados em seu computador;
- Criar uma área no seu computador que apenas pode ter arquivos criptografados;
- Proteger e-mails, transações bancárias e comerciais realizadas.
- Existem basicamente dois tipos de criptografia, a de chave simétrica e a assimétrica.
- A criptografia também foi dividida em criptografia clássica e moderna.

2.10.2.1 Criptografia Clássica

Segundo MEDEIROS (2015), a criptografia clássica é a criptografia mais antiga que vai desde os povos antigos, idade média até a segunda guerra mundial.

As cifras mais conhecidas e utilizadas na época foram a de Scytale, a de César e a de Vigenère.

2.10.2.1.1 Cifra de Scytale

A cifra de Scytale, Segundo MEDEIROS (2015), consiste em enrolar uma fita de tecido em um bastão de madeira, a frase a ser cifrada era escrita no tecido de um determinado tamanho de bastão desenrolada e enviada, ao receber o tecido deve ser enrolado em um bastão do mesmo tamanho que foi enviado, como pode ser visto na Figura 37.



Figura 37 Cifra de Scytale.

Fonte: Oliver Kuhn (<http://csegrecorder.com/assets/images/features/2013-03-science-break-01.jpg>)

Acesso em: 29 de ago. 2016.

2.10.2.1.2 Cifra de Cesar

Esta cifra é umas das cifras mais conhecidas, a cifra foi utilizada por Júlio César para comunicar suas tropas durante a guerra.

A cifra é bastante simples de ser usada já que consiste de uma letra do alfabeto pela chave definida, de acordo com MEDEIROS (2015) a chave definida pode ser três casas adiante da letra utilizada, por exemplo a letra A é substituída pela letra D, a letra B pela letra E, e assim por diante, como na Figura 38.

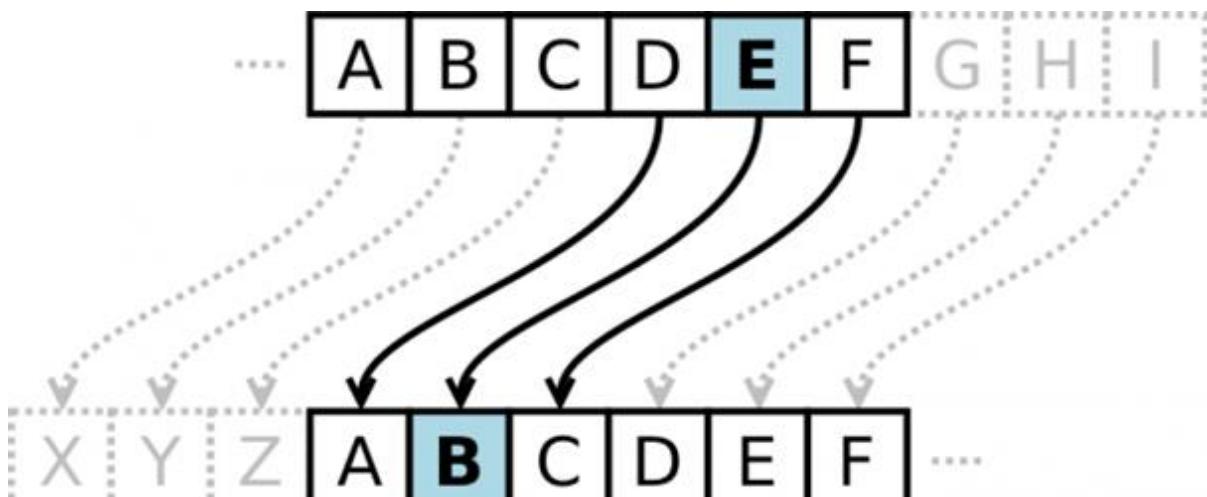


Figura 38 Cifra de Cesar.

Fonte: Douglas Ciriaco (<http://imagens.canaltech.com.br/115379.193513-Cifra-de-Cesar.png>)

Acesso em: 29 de ago. 2016.

2.10.2.1.3 Cifra de Vigenère

A cifra de Vigenère Figura 39, descrita primeiramente pelo italiano Giovan Battista Bellaso, em 1553, em sua obra La cifra del. Sig. Giovan Batista Bellaso e por muito tempo foi considerada como a cifra indecifrável, até que em meados do século XX Charles Babbage e Friedrich Kasiki encontraram um método de resolvê-la, para MEDEIROS (2015).

	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	w	v	x	y	z
1	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	W	V	X	Y	Z	A
2	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	W	V	X	Y	Z	A	B
3	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	W	V	X	Y	Z	A	B	C
4	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	W	V	X	Y	Z	A	B	C	D
5	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	W	V	X	Y	Z	A	B	C	D	E
6	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	W	V	X	Y	Z	A	B	C	D	E	F
7	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	W	V	X	Y	Z	A	B	C	D	E	F	G
8	I	J	K	L	M	N	O	P	Q	R	S	T	U	W	V	X	Y	Z	A	B	C	D	E	F	G	H
9	J	K	L	M	N	O	P	Q	R	S	T	U	W	V	X	Y	Z	A	B	C	D	E	F	G	H	I
10	K	L	M	N	O	P	Q	R	S	T	U	W	V	X	Y	Z	A	B	C	D	E	F	G	H	I	J
11	L	M	N	O	P	Q	R	S	T	U	W	V	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
12	M	N	O	P	Q	R	S	T	U	W	V	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
13	N	O	P	Q	R	S	T	U	W	V	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
14	O	P	Q	R	S	T	U	W	V	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
15	P	Q	R	S	T	U	W	V	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
16	Q	R	S	T	U	W	V	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
17	R	S	T	U	W	V	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
18	S	T	U	W	V	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
19	T	U	W	V	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
20	U	W	V	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
21	W	V	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
22	V	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	W
23	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	W	V
24	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	W	V	X
25	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	W	V	X	Y

Figura 39 Cifra de Vigenère.

Fonte: Marcelo Ferroni (http://galileu.globo.com/edic/118/imagens/eureca_01.gif)
Acesso em: 29 de ago. 2016.

MEDEIROS (2015) também explica que o processo de cifragem funciona assim: o usuário cifrará a mensagem com uma chave alfabética, caso a quantidade de caracteres da chave for menor que o tamanho de caracteres da mensagem, a chave será repetida até ambas terem a mesma quantidade de caracteres. Fazendo uma relação entre as duas, (cada letra da mensagem será cifrada com um alfabeto definido pelo caractere da chave ao qual estará relacionada.

2.10.2.2 Criptografia Moderna

Com o avanço da tecnologia e da criptografia, foram desenvolvidos alguns métodos para ajuda a cifragem e decifragem, uns dos equipamentos mais

conhecidos é a máquina Enigma, utilizada pelos alemães durante a segunda guerra mundial, afirma MEDEIROS (2015)

A Enigma como se pode ver na Figura 40, parece com uma máquina de escrever, onde ao invés de colocar o resultado no papel, o mesmo era mostrado em um painel luminoso com os caracteres do alfabeto. A chave usada para cifrar e decifrar uma mensagem era configurada por meio de rotores eletromecânicos geralmente três ou mais que podiam ser alterados conforme a necessidade para formar a chave.



Figura 40Maquina Enigma.

Fonte: Carlos Cardoso (<http://uploads.meiobit.com/2014/03/20140313enigma.jpg>)
Acesso em: 29 de ago. 2016.

2.10.2.3 Chave simétrica

Segundo o CERT (Centro de Estudos, Respostas e Tratamento), conhecida também como criptografia secreta ou única, ela utiliza uma mesma chave tanto para codificar como para decodificar o arquivo, alguns exemplos de métodos criptográficos que usam chave simétrica são: AES, *Blowfish*, RC4, 3DES e IDEA.

2.10.2.4 Chave assimétrica

De acordo com o CERT (Centro de Estudos, Respostas e Tratamento), ao contrário da criptografia simétrica ela utiliza duas chaves assim uma chave pode ser divulgada e a outra fica privada, quando um arquivo é criptografado com uma das chaves somente o par dela conseguira decodificar,

alguns métodos criptográficos que usam chaves assimétricas são: RSA, DSA, ECC e Diffie-Hellman.

2.10.3 Assinatura digital

Assinatura digital, segundo o CERT (Centro de Estudos, Respostas e Tratamento), permite comprovar a autenticidade e a integridade de uma informação, ou seja, que ela foi realmente gerada por quem diz ter feito e que não foi alterada.

Ela se baseia no fato que apenas o dono conhece a chave privada e se ela foi usada para codificar um arquivo, então apenas o dono que poderia ter criado o arquivo criptografado.

2.10.4 Certificado Digital

Certificado digital, para o CERT (Centro de Estudos, Respostas e Tratamento) é um registro eletrônico composto por um conjunto de dados que distingue uma entidade e associa a ela uma chave assimétrica, ela é emitida para pessoas, empresas, equipamentos e serviços na rede (como por exemplo, um site).

O certificado é similar a um documento de identidade, ele contém os dados pessoais e quem o emitiu, a entidade que emite o certificado é a Autoridade Certificadora (AC).

2.10.4.1 SSL

Segundo a Central Server, SSL (*Secure Socket Layer*) é um protocolo de segurança que fornece mais segurança aos arquivos transmitidos pela internet, o SSL pode ser usado em diversos serviços mas geralmente é utilizado em páginas web, o SSL usa criptografia no envio e recebimento de arquivos, usaremos a versão de 256 bit.

O site está usando SSL quando aparece “*HTTPS*” em verde ao lado de um cadeado, como a Figura 41.



Figura 41 HTTPS.

2.10.5 Firewall

A Cisco define Firewall como um dispositivo de segurança que monitora o tráfego de entrada e saída da rede, um Firewall pode ser um hardware, software ou os dois. Existem alguns tipos de firewalls.

- Firewall de proxy funciona como uma passagem de rede para outra aplicação.
- Firewall com inspeção de estado, ou conhecido como firewall tradicional ele permite ou bloqueia o tráfego da rede, o que deve ser bloqueado ou que deve ser permitido pelas regras criadas pelo administrador.

2.11 PROGRAMAÇÃO DE COMPUTADORES

De acordo com Ascencio e Campos (2007) programação de computadores é um processo conduzido a partir de um problema de computação para gerar um programa de computador executável. A programação envolve atividades como análise, desenvolvimento, estudo de caso, geração de algoritmos, escrita, manutenção verificação de requisitos básicos e até a correção e consumo de recursos como hardware ou software e até a implementação em si.

A programação geralmente é feita através de uma ou mais linguagens de programação. O objetivo da programação é automatizar um processo, procedimento ou resolver um problema. O processo da programação muitas vezes exige um grau de conhecimento do problema a ser resolvido, pensamento lógico e também aplicação de técnicas e algoritmos especializados (ASCENCIO, CAMPOS, 2007).

2.11.1 Linguagens de Programação

A linguagem de Programação é uma linguagem de computação formal ou construída projetada com o intuito de se comunicar com uma máquina, particularmente um computador. Elas são usadas para controlar uma máquina resolver um problema ou implementar um algoritmo de acordo com Ascencio e Campos (2007).

2.11.1.1 Linguagem C

De acordo com Kernigan e Ritchie (1988) a linguagem C é uma linguagem de programação de aplicação geral que tem como características básicas a economia de expressão, controle de fluxo e estrutura de dados, um exemplo de uma fração de um programa em C pode ser visto na Figura 42. Ela é uma linguagem de baixo nível, isto é, a linguagem apresenta pouca ou nenhuma abstração em relação ao conjunto de instruções do computador a linguagem não possui nenhuma área particular de aplicação. Entretanto a sua falta de restrições e sua generalidade fazem com que ele seja mais conveniente e efetivo para várias aplicações comparado a algumas linguagens mais poderosas. Além disso, sua alta portabilidade, isto é, a facilidade de implementar o mesmo software em dois computadores diferentes, foi uma das características que ajudaram o C a ser uma das linguagens mais utilizadas.

```

main.c *main.c
78     for(i=N;i > 0;i--)pull(&pilha1,i);
79
80     for(i=0;i < M;i++){
81         scanf("%d %d",&po,&pd);
82         if(po == 1)
83             item = push(&pilha1);
84         if(po == 2)
85             item = push(&pilha2);
86         if(po == 3)
87             item = push(&pilha3);
88         printf("item %d\n",item);
89         if(item == -1)erro = 1;
90         else{
91             if(pd == 1)
92                 erro = transfere(&pilha1,item);
93             if(pd == 2)
94                 erro = transfere(&pilha2,item);
95             if(pd == 3)
96                 erro = transfere(&pilha3,item);
97         }
98         printf("p1 item %d -> topo %d\n",pilha1.item[pilha1.topo],pilha1.topo);
99         printf("p2 item %d -> topo %d\n",pilha2.item[pilha2.topo],pilha2.topo);

```

Figura 42 Exemplo de Programa em C(fração).

Segundo Kernigan e Retchie (1988), o C não possui operações para trabalhar diretamente com objetos compostos como *strings* de caracteres, sets, listas ou arranjos. Também não possui em sua biblioteca padrão operações para manipular um arranjo ou *string*, porém essas estruturas podem ser copiadas como uma unidade. A linguagem não define nenhuma função para alocação e as pilhas de instruções fornecidas pelas variáveis locais das funções. Finalmente o C sozinho não possui funções para *input/output* (entrada/saída de dados); não tem possuir declarações de *READ* ou *WRITE*, não possui métodos embutidos para acesso de arquivos. Todos esses mecanismos de alto nível devem ser declarados

explicitamente e chamados através de funções. A maior parte das implementações em C tem incluído uma enorme coleção dessas funções.

O C por si só possui somente controles de fluxo diretos de único segmento como testes, *loops* agrupamentos e subprogramas, entretanto não possui a multiprogramação, operações paralelas, sincronização nem co-rotinas acrescenta Kernigan e Ritchie (1988).

2.11.1.1 História

De acordo com Schildt (1996) a linguagem C foi inventada por Dennis Ritchie em 1972 por Dennis Ritchie em um PDP-11. Ela foi baseada na Linguagem B criada em 1967 por Ken Thompson em um PDP-7, para transferir a primeira versão do Linux para uma linguagem de alto nível. A linguagem B por sua vez foi inspirada na linguagem de programação BCPL (*Basic Combined Programming Language*) uma linguagem de alto nível desenvolvida por Martin Richards.

Por vários anos o padrão da linguagem C, era aquela descrita no livro *The C Programming Language*, de Brian Kernighan e Dennis Ritchie. Entretanto com o passar do tempo a popularidade da linguagem vinha crescendo e mais pessoas estavam a utilizá-la. Mas como não havia um padrão havia algumas discrepâncias nas implementações ao passo que o hardware se modificava. Por esse e outros motivos, o ANSI (*American National Standards Institute*) estabeleceu em 1983, um comitê para criar um padrão para a definir a linguagem C, conforme Schildt (1996).

2.11.1.2 PHP (Personal Home Page)

Segundo PHP Group 2016, o PHP é uma linguagem interpretada livre, usada para desenvolver aplicações que funcionam com um servidor, capazes criar conteúdo dinâmico para páginas na WEB.

A linguagem foi criada em 1994 por Rasmus Lerdof, e chamada de *Personal Home Page Tools*. Segundo PHP Group 2016, ela substituiu alguns scripts Perl que eram usadas em suas páginas pessoais. O código fonte foi liberado ao público apenas em 1995.

Em 1997 um novo pacote desta linguagem foi lançado, chamando-se PHP/FI, trazendo ferramentas *Forms interpreter*, um interpretador de comandos SQL.

Pouco tempo depois surge o PHP 3, desenvolvido por Zeev Suraski, contendo seus primeiros recursos de orientação a objetos, e mais tarde, juntamente com Andi Gutmans, Zeev escreveu o PHP 4, oferecendo mais recursos de orientação a objetos e alcance a mais pacotes, sendo lançado em 2000. O Problema do PHP 4 era a falta dos Apontadores, ou *Handlers*, recurso que permite a cópia de objetos de forma dinâmica, ou seja, alterando um objeto, dasas as cópias serão alteradas, problema que foi corrigido no PHP 5, que foi lançado em 2004 conforme PHP Group (2016).

2.11.1.2.1 PHP 7

A versão que será utilizada em nosso projeto é a PHP 7, sendo a versão mais recente desta linguagem, com novos recursos e mais facilidades ao usuário, segundo PHP Group 2016.

2.11.1.3 HTML

De acordo com CARVALHO (2007), o HTML (Abreviação para *HiperText Markup Language*, em português, Linguagem de Marcação de Hipertexto) é uma linguagem de marcação que vem da junção do SGML e do HyTime, usada pra o desenvolvimento de sites ou páginas web.

O SGML (*Standard Generalizes Markup Language*), para CARVALHO (2007) é uma metalinguagem, que pode definir linguagens de marcação para documentos. Ela é a sucessora da GML, desenvolvida na década de 1960 pela IBM, por Charles Goldfarg, Edward Mosher e Raymond Lorie. Esta linguagem não deve ser confundida com a *Geography Markup Language* (GML), criada pela Open GIS.

O HyTime (redução para *Hypermedia/Time-based Document Structuring Language*) é também uma linguagem de margeação, que segundo CARVALHO (2007), é uma aplicação do SGML.

2.11.1.3.1 Historia

Para RAGETT (1998), a linguagem foi desenvolvida pelo físico britânico Tim Berners-Lee, na mesma época em que surgiu o HTTP. Segundo a *World Wide Web Consortium*, Inicialmente, O HTML era apenas um conjunto de ferramentas criados para resolver alguns problemas de Tim, como a disseminação

de pesquisas e a comunicação entre ele e alguns de seus colegas, porém, na década de 1990 a Linguagem foi definida em especificações formais, e publicada em 1993 por Berners-Lee e Dan Connolly na IETF (*Internet Engineering Task Force*), apenas como uma aplicação formal para o SGML. Um ano depois a IETF criou um grupo de trabalho, que em 1995 publicou o HTML 2.0, e dês de então as especificações foram mantidas com ajuda das desenvolvedoras de softwares e da *World Wide Web Consortium* (W3C).

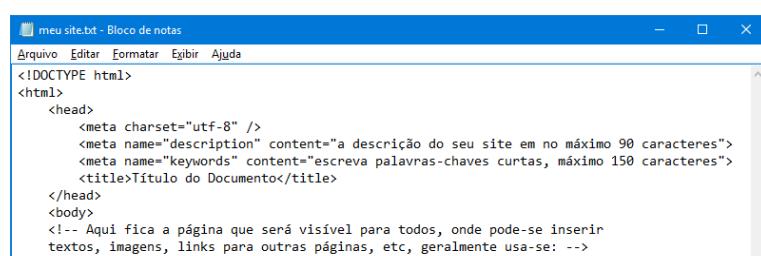
A W3C ainda lançou o HTML 3.5 no final de 1997, e o HTML 4.1 em 1999. No ano de 2000, a linguagem se tornou uma norma internacional, porém desde a publicação do HTML 3.5 a W3C já vinha trabalhando no desenvolvimento do XHTML, que é uma especificação do HTML, com base no XML que é considerado um sucessor do HTML para a W3C. O XHTML utiliza sintaxes mais rigorosas, fazendo com que o código seja mais simples de ser processado, de acordo com a PEMBERTON (2008)

Em 2008, foi publicado pela W3C o HTML5, que apesar de mantes suas sintaxes semelhantes à do SGML, ele deixou de ser uma aplicação SGML, e além disso, veio uma alternativa baseada no XML, o XHTML5, segundo DUBOST (2008)

2.11.1.3.2 Como funciona

O HTML, segundo NEVES (2004) é formado uma grande série de marcadores, do inglês, *tags* que nesta linguagem são os parênteses angulares ("<" e ">"). Esses marcadores são os comandos que formatam a linguagem. Um Elemento é formado por uma *tag* (um marcador), atributos, valores e filhos. Os atributos, mortificam os resultados padrões do elemento, os valores, caracterizam essa mudança, e os filhos, podem ser outros elementos dentro da marca ou um texto.

De acordo com NEVES (2004) um documento em HTML, é um arquivo de texto bem simples que pode ser criado em qualquer editor de texto como o bloco se notas do Windows, conforme mostra a Figura 43, em editores mais avançados que facilitam a visualização do documento, ou em editores específicos para HTML que oferecem ferramentas para facilitar o desenvolvedor.



```

meu site.txt - Bloco de notas
Arquivo  Editar  Formatar  Exibir  Ajuda
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8" />
    <meta name="description" content="a descrição do seu site em no máximo 90 caracteres">
    <meta name="keywords" content="escreva palavras-chaves curtas, máximo 150 caracteres">
    <title>Título do Documento</title>
  </head>
  <body>
    <!-- Aqui fica a página que será visível para todos, onde pode-se inserir
textos, imagens, links para outras páginas, etc, geralmente usa-se: -->
  </body>
</html>

```

Figura 43 Exemplo de HTML

2.12 CASCADING STYLE SHEETS (CSS)

2.12.1 Definição

CSS é uma linguagem que especifica como os documentos são apresentados aos usuários e define como serão exibidos os elementos contidos no código de uma página da internet, a sua principal finalidade é efetuar a separação entre o formato e o conteúdo de um documento segundo PEREIRA (2009).

2.12.2 História

Com a evolução dos sites a cada dia, logo foi necessário aprimorar o desenvolvimento de linguagens de marcação como XML,HTML e XHTML. Essas linguagens precisavam apresentar os conteúdos das páginas mais elegantes e atrativas para o usuário. Novas *tags* e atributos foram criadas para o HTML, mas ocorreram complicações, a partir destas nasceu o CSS, que segundo PEREIRA (2009), foi desenvolvido para habilitar a separação do conteúdo e formato de um documento.

2.13 CÂMERA

2.13.1 Definição

A máquina fotográfica ou câmera é um instrumento óptico que possui por finalidade conjugar imagens finais reais. Segundo Correa Marques (2016), o instrumento óptico tem toda combinação conveniente de dispositivos ópticos, como espelhos, prismas e lentes segundo Patrício 2011.

2.13.2 História

Por volta de 350 a.C, o conceito de fotografia foi criado quando o filósofo grego Aristóteles inventou um método de observar os eclipses solares que foi chamada de câmara escura. Esse método foi um dos mais importantes devido ao conhecimento dos princípios óticos. A câmara escura foi a primeira máquina fotográfica da história. Essa máquina só exibia as imagens, surgindo a necessidade de fixa-las. No início do século XIX, Thomas Wedgwood usou a substância química nitrato de prata para fixar as imagens da câmera escura, esse processo durava várias horas. Segundo Patrício 2011, no final do século XIX, George Eastman, fundador da Kodak Company, teve a ideia de criar uma longa camada de nitrato de celulose que, a cada foto, era enrolada em uma espécie de carretel. Suas câmeras ficaram conhecidas pela simplicidade e pelo fato dos usuários precisarem apenas apertar um botão.

2.14 CÂMERA IP

2.14.1 Definição

É uma câmera que pode ser acessada e controlada via qualquer rede IP, como a Intranet, Internet ou LAN. Usando simplesmente uma conexão com a internet caso seja utilizada de forma externa e um navegador web segundo Corrêa 2013.

2.14.2 Funcionamento

Essas câmeras são equipadas com um servidor interno e uma placa de processamento. Essas câmeras não necessitam de softwares ou placas adicionais, tornando fácil suas instalação e manuseio dentro de uma rede segundo Corrêa 2013.

3 MÉTODOS E PROCEDIMENTOS

3.1 CRONOGRAMA

As atividades à serem desenvolvidas seguirão as seguintes etapas, conforme o cronograma do projeto, visto na Tabela 1.

Tarefa	Dias	Início	Fim
Início das Aulas	1	25/07/2016	25/07/2016
Verificação dos requisitos do projeto	1	26/07/2016	26/07/2016
Compartilhamento no ONEDRIVE	1	27/07/2016	27/07/2016
Definição dos requisitos	4	28/07/2016	01/08/2016
Elaboração das respostas acerca do tema do projeto	2	02/08/2016	04/08/2016
Início da Documentação	3	05/08/2016	08/08/2016
Introdução	3	09/08/2016	11/08/2016
Fundamentação Teórica	25	12/08/2016	06/09/2016
Mapa Mental	1	07/09/2016	07/09/2016
Métodos e Procedimentos	20	08/09/2016	27/09/2016
Pré-Banca	2	19/09/2016	20/09/2016
Desenvolvimento	44	21/09/2016	31/10/2016
Resultados e discussão	7	01/11/2016	08/10/2016
Recomendações	5	09/10/2016	16/10/2016
Finalização	3	17/10/2016	21/10/2016
Ensaio TCC	3	22/10/2016	25/10/2016
Apresentação TCC	2	28/10/2016	29/10/2016

Tabela 1 Cronograma do Projeto.

3.2 CUSTO

Foi estipulado um valor R\$ 9.711,24 levando em consideração a mão de obra, e todo o equipamento necessário, como pode ser visto na Tabela 2. Estes custos podem diminuir caso o contratante já ofereça parte do equipamento.

Tabela de Preços					
Produtos/Programas	Marcas/Versões	Preços	Quantia	Total	
Computador	Dell	R\$ 2.650,00	1	R\$ 2.650,00	
Windows 10	Pro	R\$ 799,00	1	R\$ 799,00	
Braço de acrilico	Fabricação caseira	R\$ 39,99	1	R\$ 39,99	
Arduino	Mega 2560	R\$ 80,00	1	R\$ 80,00	
Modulo Wi-fi	Ai-cloud inside	R\$ 29,90	1	R\$ 29,90	
Protoboard	Minipa	R\$ 15,00	2	R\$ 30,00	
Jumper	-	R\$ 0,85	80	R\$ 68,00	
Mão de obra	Humana	R\$ 1.200,00	5	R\$ 6.000,00	
Programa Arduino	Arduino	-	1	-	
Virtualizador	VirtualBox	-	1	-	
Apache	Apache 2	-	1	-	
Linux	Ubuntu	-	1	-	
Php 7	Php	-	1	-	
Mysql	Mysql	-	1	-	
SSL	SSL	-	1	-	
Fonte ajustavel	Elego	R\$ 12,00	1	R\$ 12,00	
Resistor	-	R\$ 0,05	5	R\$ 0,25	
Led	-	R\$ 0,05	2	R\$ 0,10	
Cabo serial	-	R\$ 2,00	1	R\$ 2,00	
TOTAL		R\$		9.711,24	

Tabela 2 Tabela de Custos.

3.3 TIPO DE PESQUISA

Cartesiano, pois o projeto terá como objetivo, otimizar a utilização de um braço robótico através do desenvolvimento e análise de uma interface simples e rápida.

3.4 CARÁTER DA PESQUISA

A abordagem será quantitativa, pois empregará dados estatísticos e se apoiará em medidas e cálculos mensuráveis.

3.5 MÉTODO DE ABORDAGEM

A pesquisa que será feita visa a verificação das formas de contribuição para o tema discutido, utilizando teorias baseadas em livros e documentos científicos. Será feita uma pesquisa de campo para obter uma estimativa da aceitação do projeto por meio pesquisas online com um formulário com 8 perguntas fechadas de respostas únicas.

3.6 DADOS DA PESQUISA DE CAMPO

A Pesquisa de Campo foi composta por 8 questões, sendo todas elas de múltipla escolha, com as seguintes alternativas: “sim”, “não” e “não sei responder”.

Um total de 42 pessoas foram entrevistadas para analisar o cenário dos trabalhadores, e os resultados foram os seguintes:

52% dos entrevistados trabalham, e 48% não trabalham, como mostra o Gráfico 1.

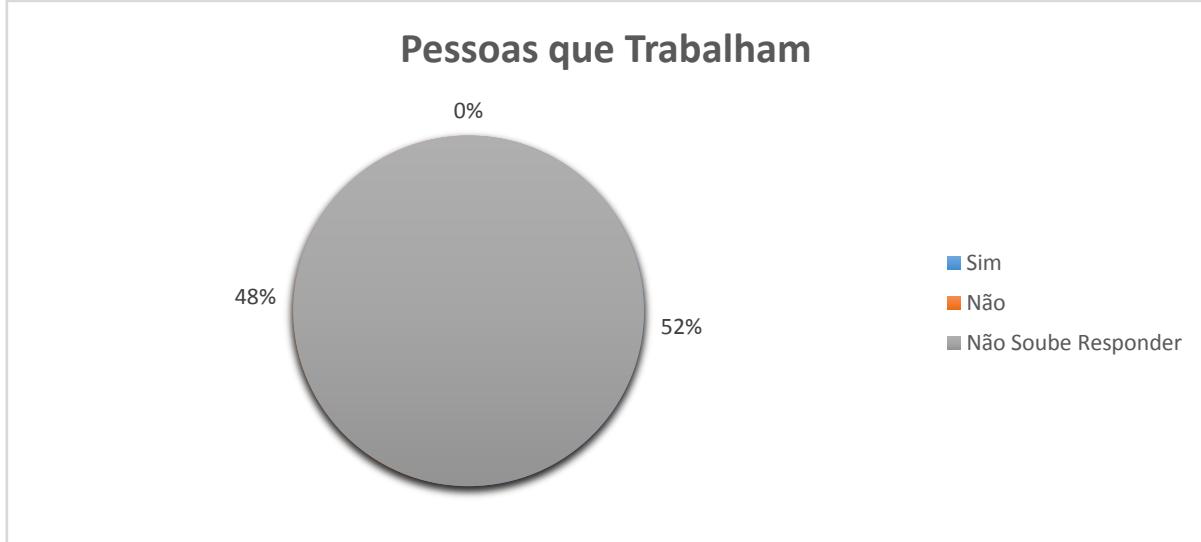


Gráfico 1 Entrevistados que Trabalham.

O Gráfico 2 mostra que 38% dos entrevistados trabalham em algo que envolve esforço físico, e 57% não. 5% dos entrevistados não souberam responder.



Gráfico 2 Esforço Físico no Trabalho.

O Trabalho de 38% dos entrevistados oferece algum tipo de risco à saúde, já de outros 52% dos entrevistados não, e 10% não souberam responder à questão, como pode ser visto no Gráfico 3.

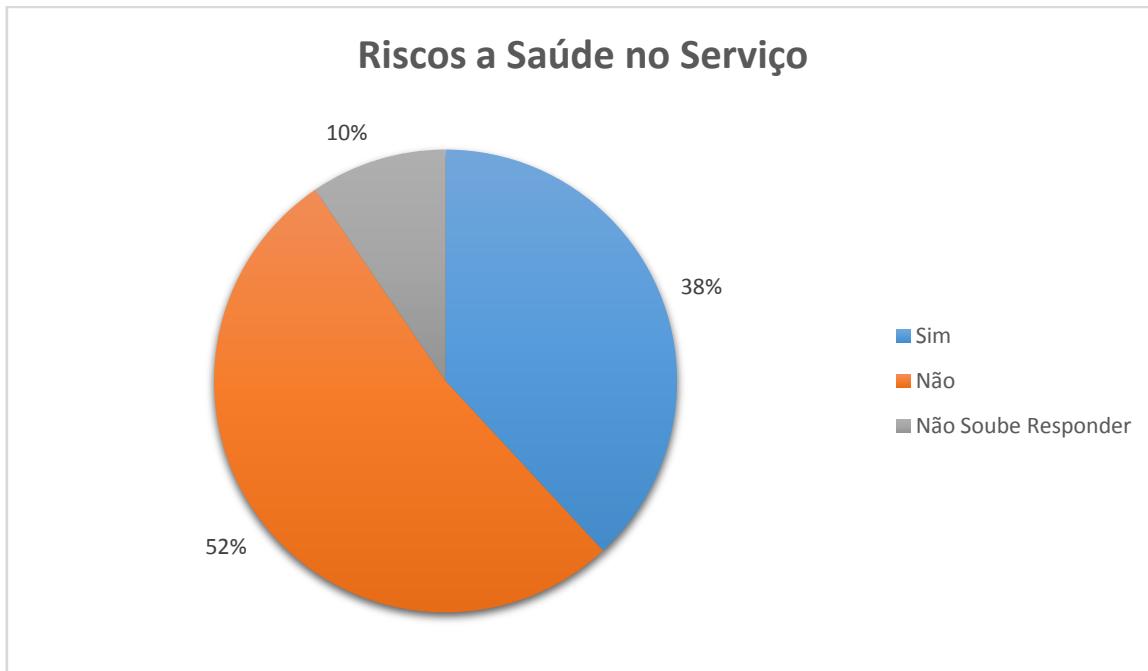


Gráfico 3 Risco a Saúde no Serviço.

O Gráfico 4 apresenta que 36% dos que participaram, trabalham em linha de produção, 57% não, e 7% não soube responder.

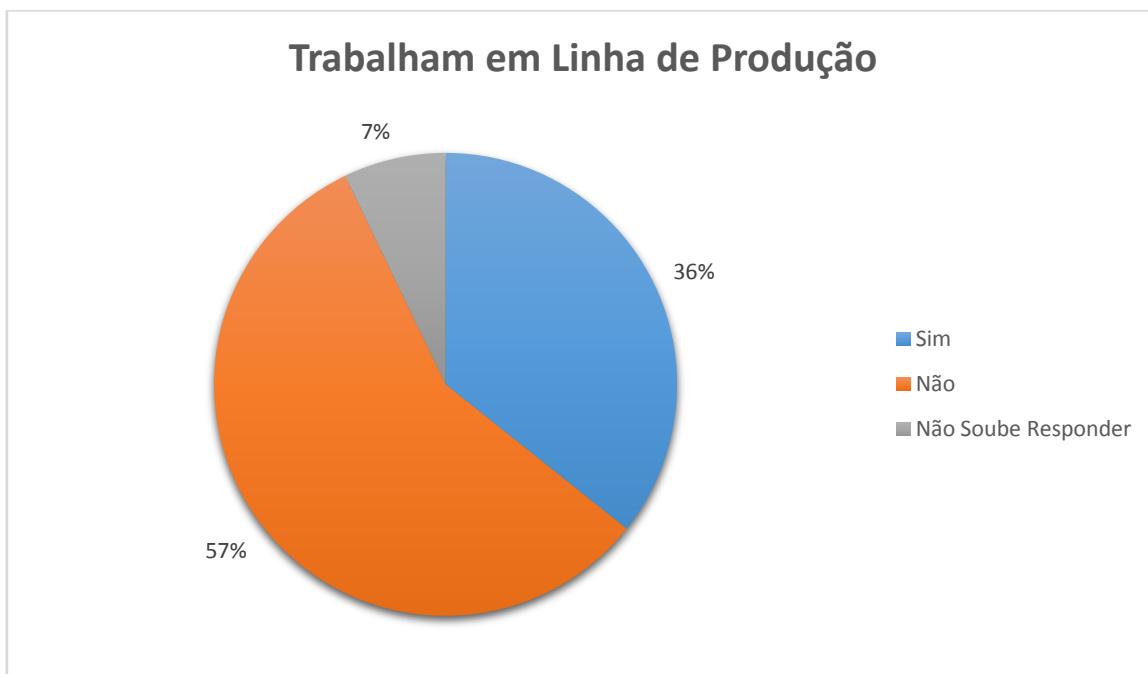


Gráfico 4 Trabalham em Linha de Produção.

Apenas 10% dos participantes da pesquisa são chefes de uma empresa ou microempresa, como apresenta o Gráfico 5. 88% não são chefes de nada e 2% não souberam responder.

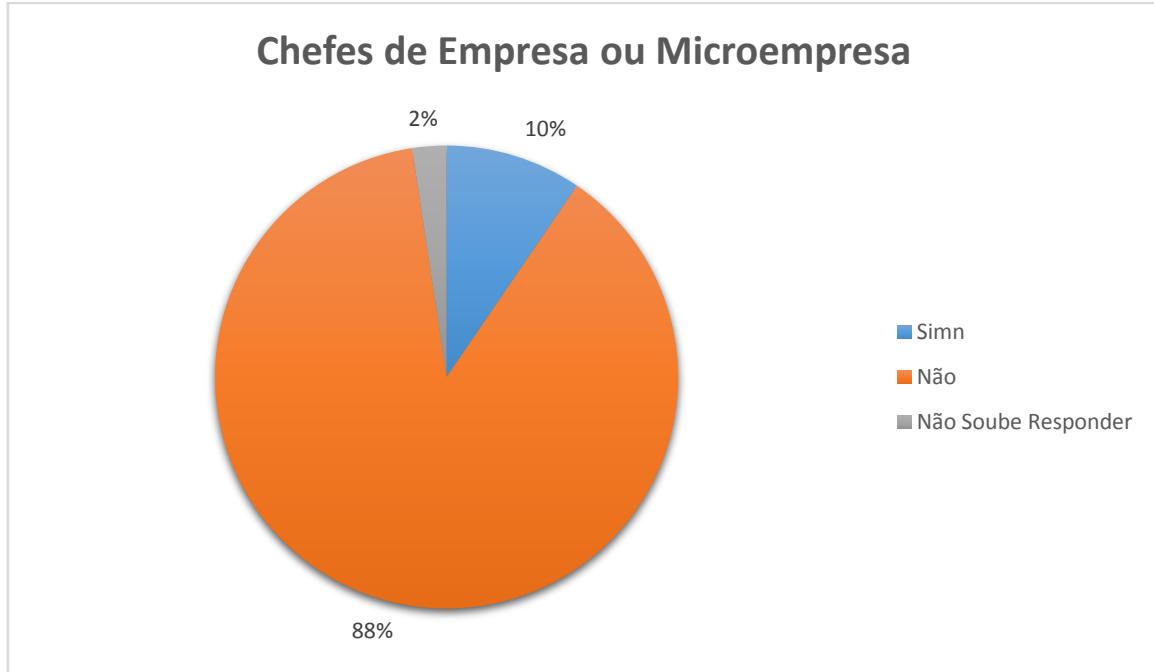


Gráfico 5 Chefes de Empresa ou Microempresa.

O Gráfico 6, mostra que 55% dos participantes tem interesse em aumentar seus lucros, desenvolver sua empresa e melhorar as condições de serviço, 19% não tem esse interesse, e 26% não souberam responder.

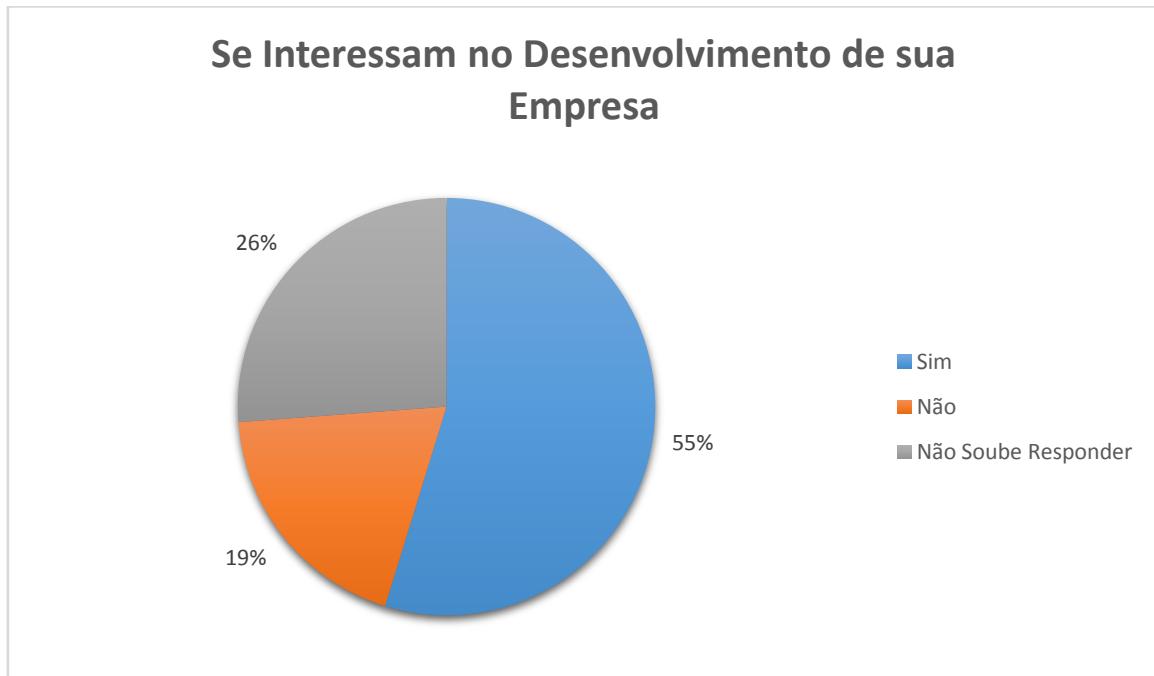


Gráfico 6 Se Interessam no Desenvolvimento de sua Empresa.

É apresentado no Gráfico 7 que 64% dos entrevistados comprariam o Equipamento caso se encaixassem nas situações citadas acima, 24% não souberam responder e apenas 12% dos Participantes não comprariam.

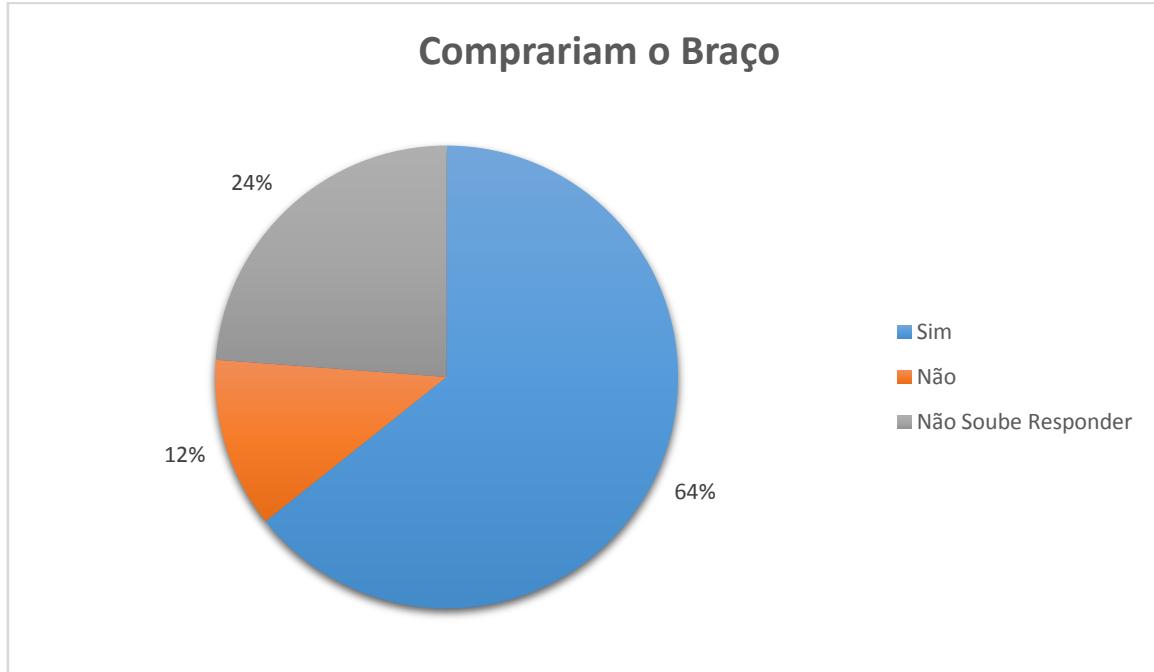


Gráfico 7 Pessoas que Compraria o Braço.

81% dos Participantes afirmaram que acreditam em nosso Projeto, como mostra o Gráfico 8. 12% não souberam responder e apenas 7% desacreditam de nosso projeto.

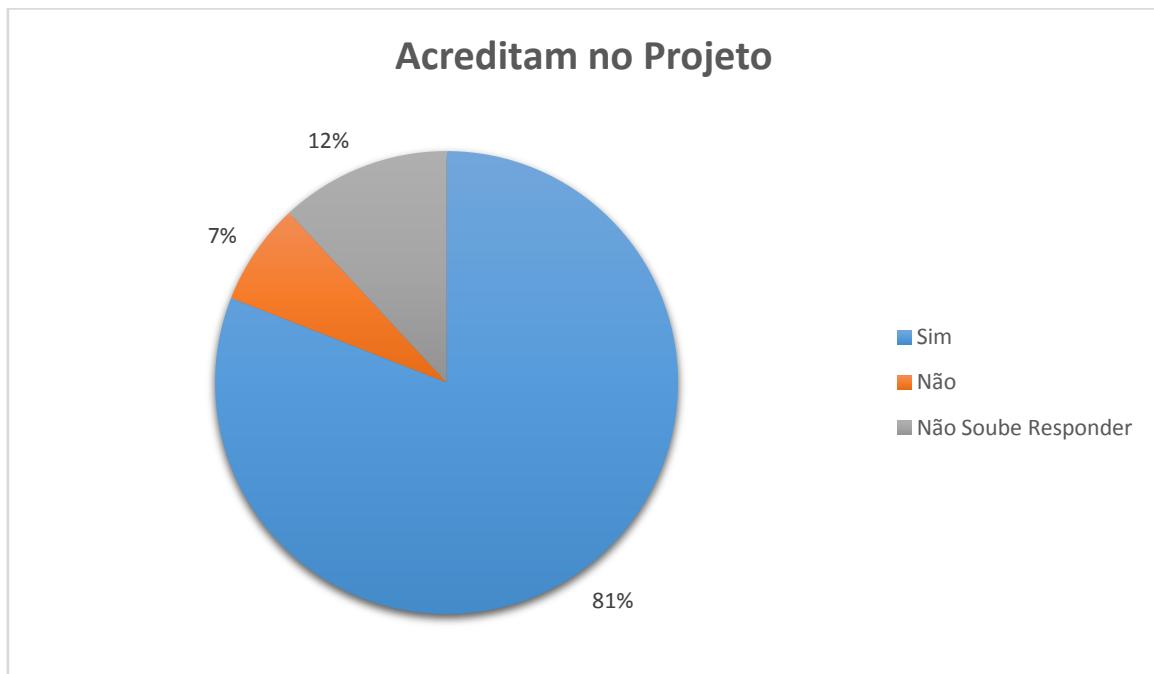


Gráfico8 Acreditam no Projeto.

4 DESENVOLVIMENTO

4.1 INSERÇÃO DO MÓDULO WI-FI

A única alteração feita no hardware do braço foi a ligação do Módulo Wi-Fi e de uma fonte externa para a alimentação do módulo e do próprio braço, já que não receberia mais sua alimentação através da porta USB.

As ligações feitas entre a protoboard, placa arduino e módulo podem ser vistas na Figura 44.

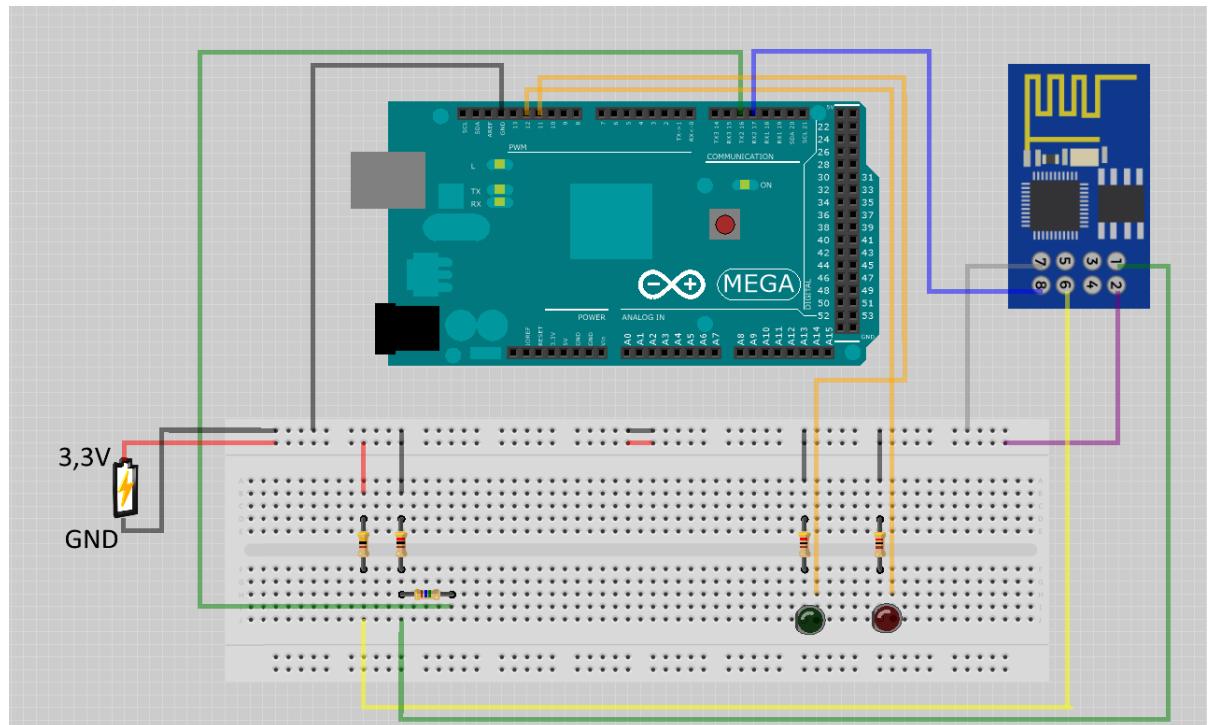


Figura 44 Ligações do Módulo.

4.2 INSTALAÇÃO DO VIRTUALIZADOR

Inicialmente foi necessária a instalação de um virtualizado, neste caso, foi utilizado o Oracle VM VirtualBox.

Iniciou-se o instalador do VirtualBox, como apresenta a Figura 45.

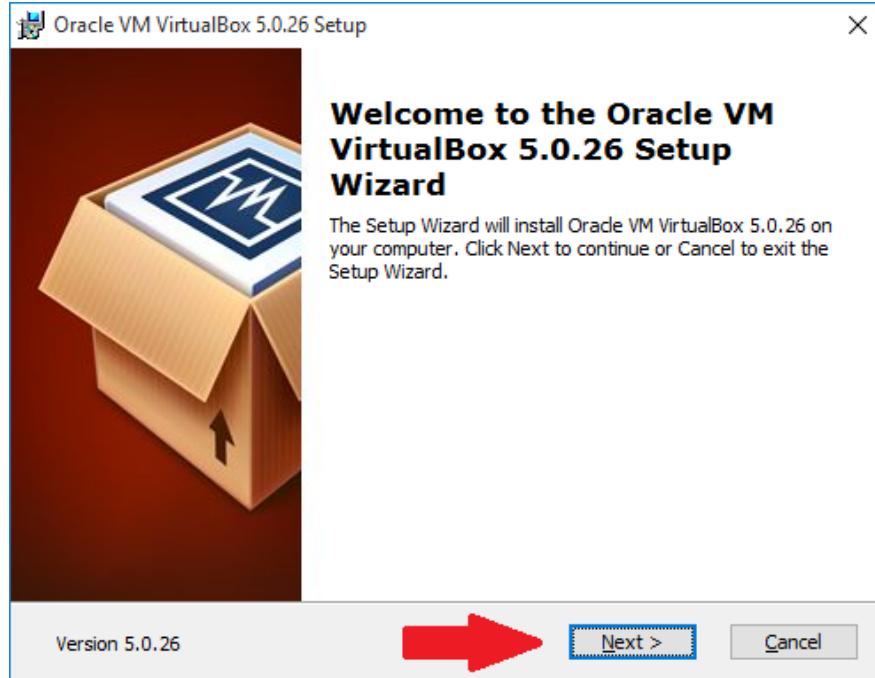


Figura 45 Instalação do Virtual Box (Parte 1).

Como na Figura 46, selecionou-se o local de instalação.

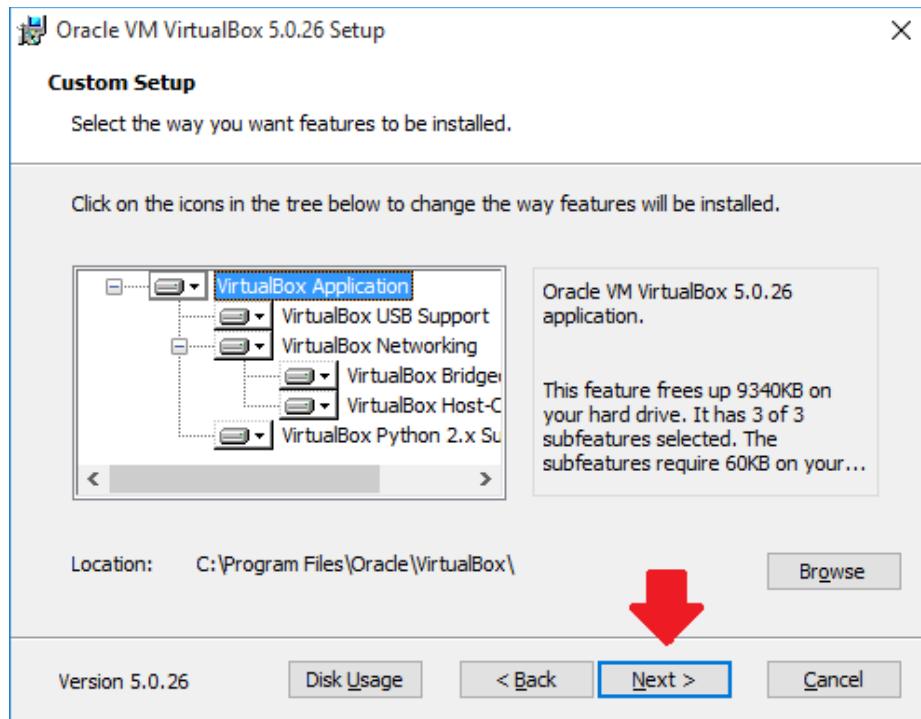


Figura 46Instalação do Virtual Box (Parte 2).

Para dinamizar o uso da feramente criou-se atalhos na área de trabalho e no menu iniciar durante a instalação, como apresenta a Figura 47.

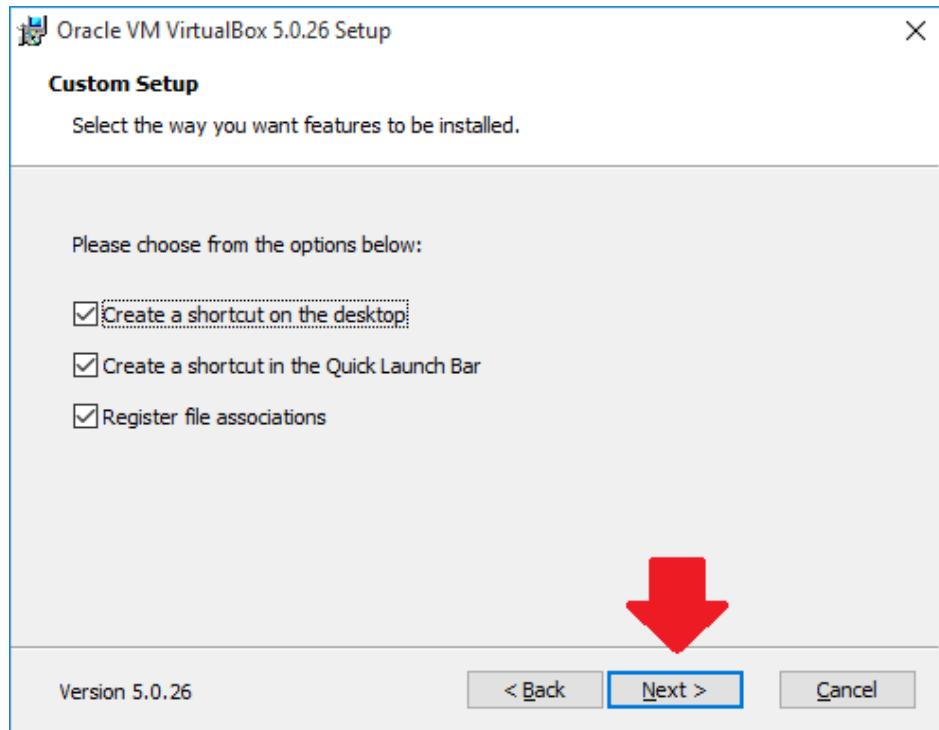


Figura 47Instalação do Virtual Box (Parte 3).

Deu-se início à instalação do programa de acordo com as Figuras 48, 49 e 50.

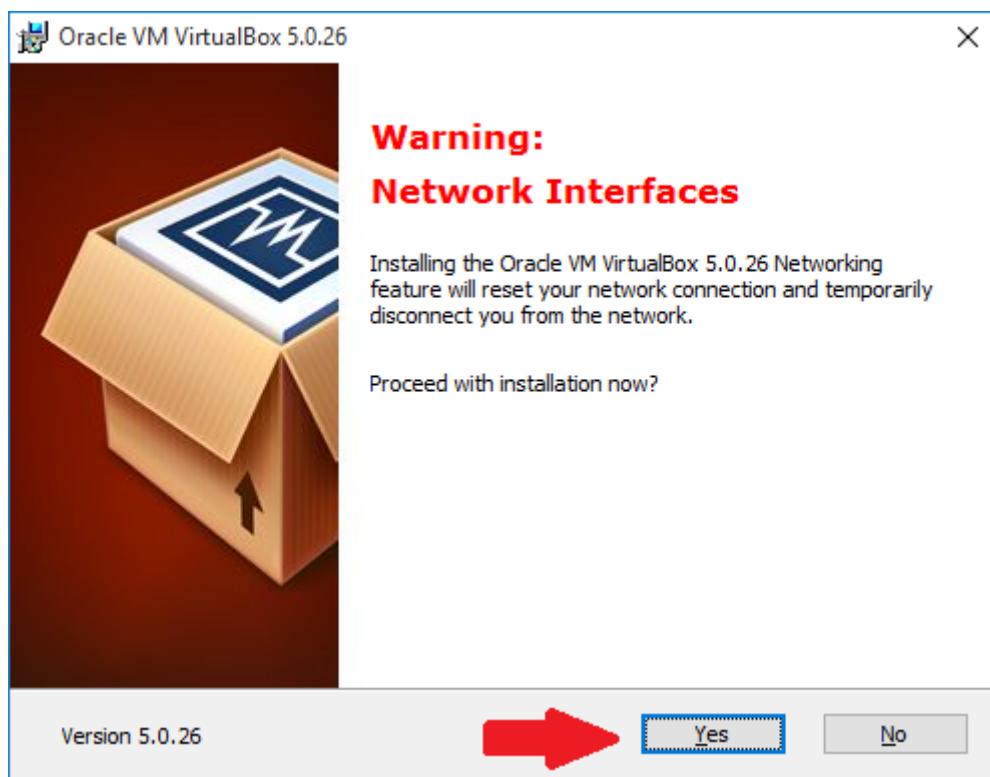


Figura 48Instalação do Virtual Box (Parte 4).

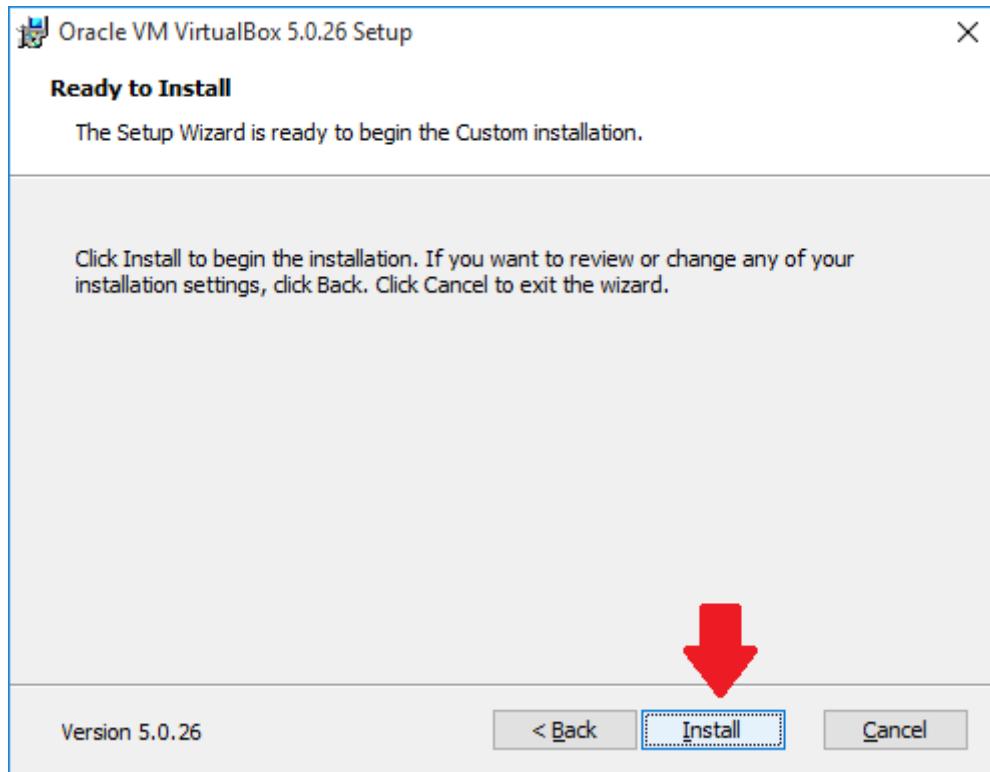


Figura 49Instalação do Virtual Box (Parte 5).

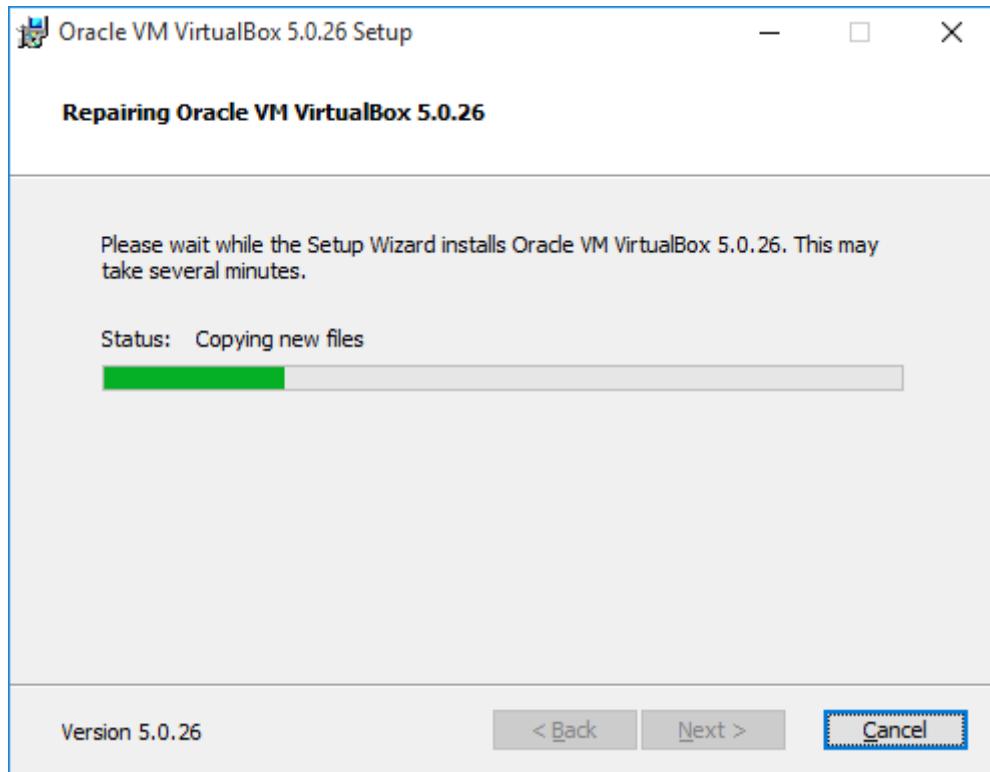


Figura 50Instalação do Virtual Box (Parte 6).

Após aguardar o tempo de instalação, o processo foi finalizado, como mostra a Figura 51.

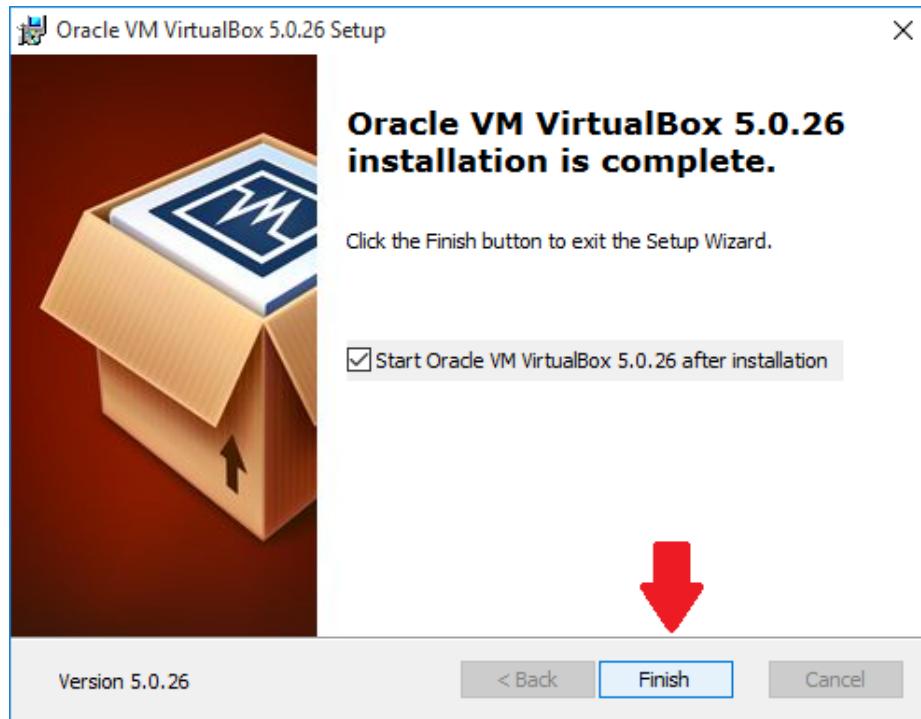


Figura 51 Instalação do Virtual Box (Parte 7).

4.3 CRIAÇÃO DE UMA MAQUINA VIRTUAL

O Segundo passo do projeto foi criação de uma máquina virtual para a instalação do sistema operacional, como na Figura 52.

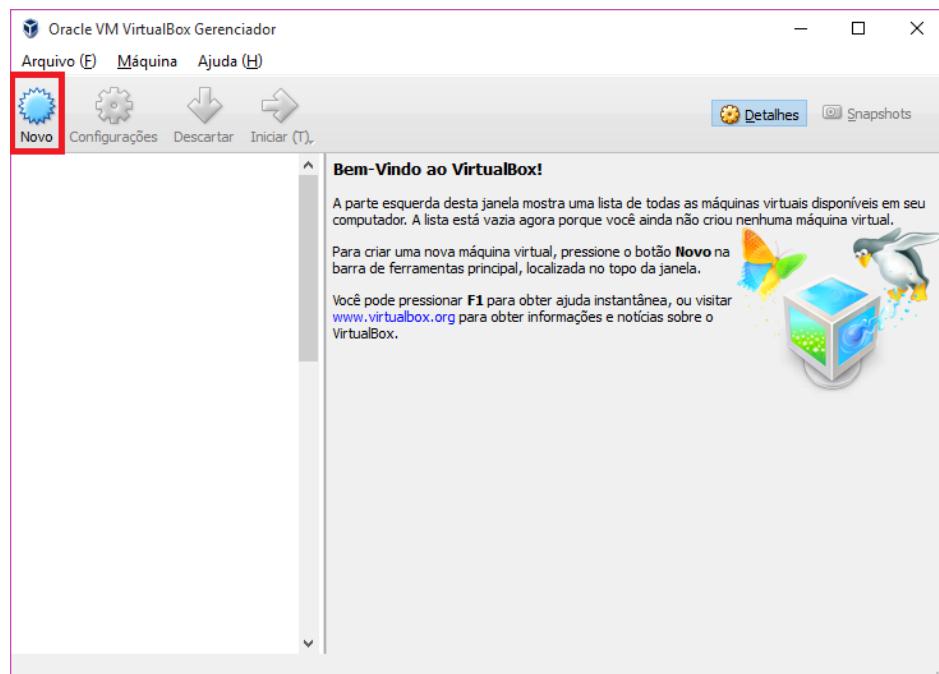


Figura 52 Criação de VM (parte 1).

Selecionou-se o sistema operacional e foi nomeada a máquina virtual, como apresenta a Figura 53.

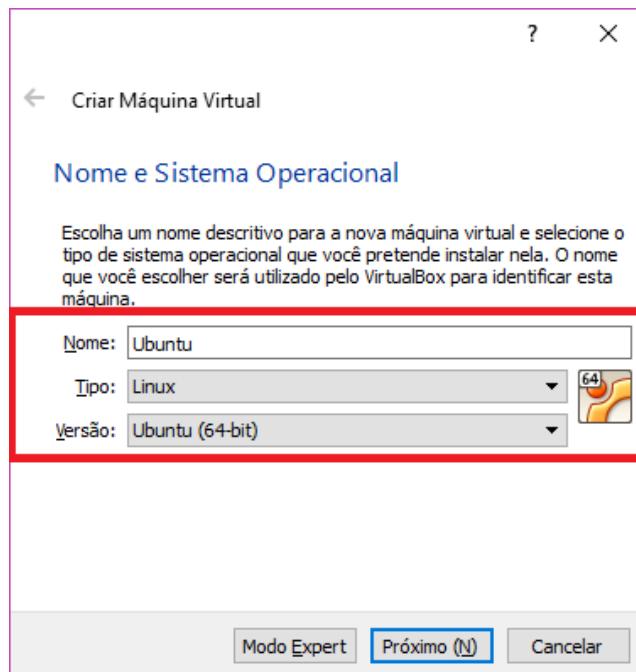


Figura 53 Criação de VM (parte 2).

Considerando que os serviços que serão instalados na máquina não exigem uma máquina potente, alocou-se apenas dois gigabytes de memória RAM para a máquina virtual, como é possível ver na Figura 54.

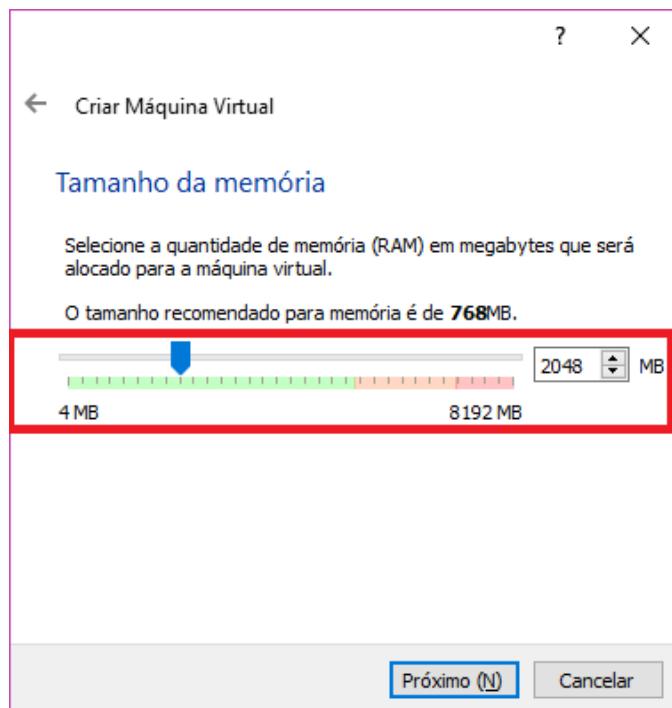


Figura 54 Criação de VM (parte 3).

As Figuras 55 e 56 mostram a criação de um novo disco rígido virtual do tipo VDI (VirtualBox Disk Image).

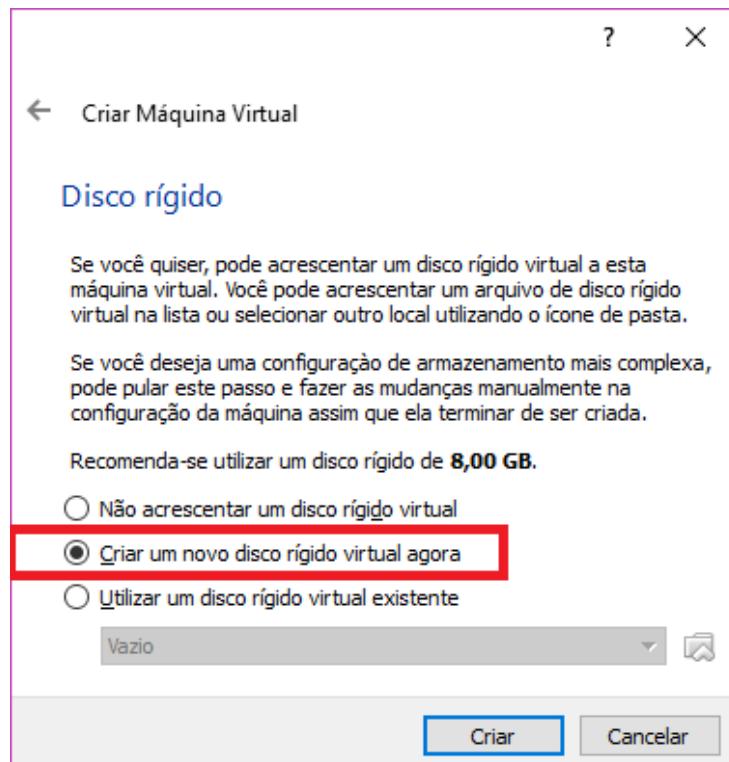


Figura 55 Criação de VM (parte 4).

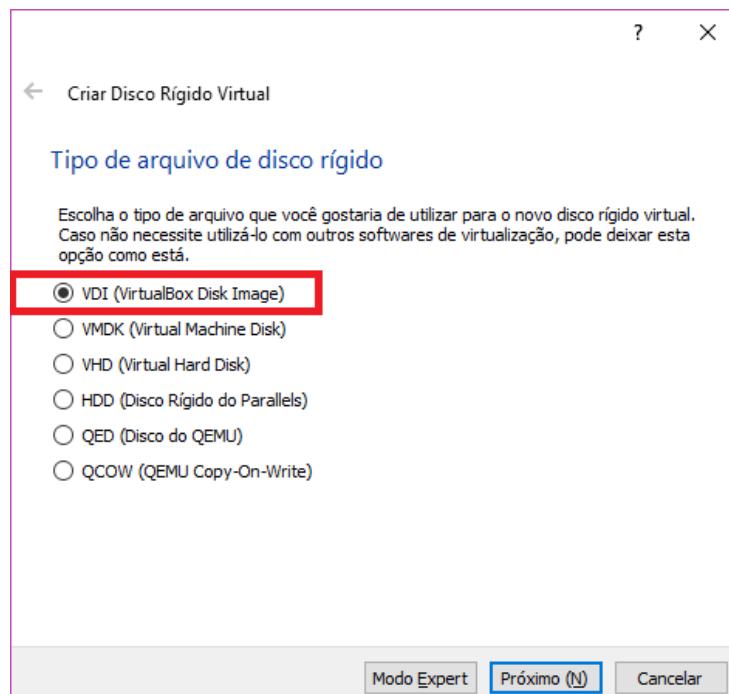


Figura 56 Criação de VM (parte 5).

O Modo de armazenamento escolhido foi dinamicamente alocado, como pode ser observado na Figura 57, para economizar espaço no disco rígido físico e caso fosse possível aumentar o espaço caso houvesse a necessidade.

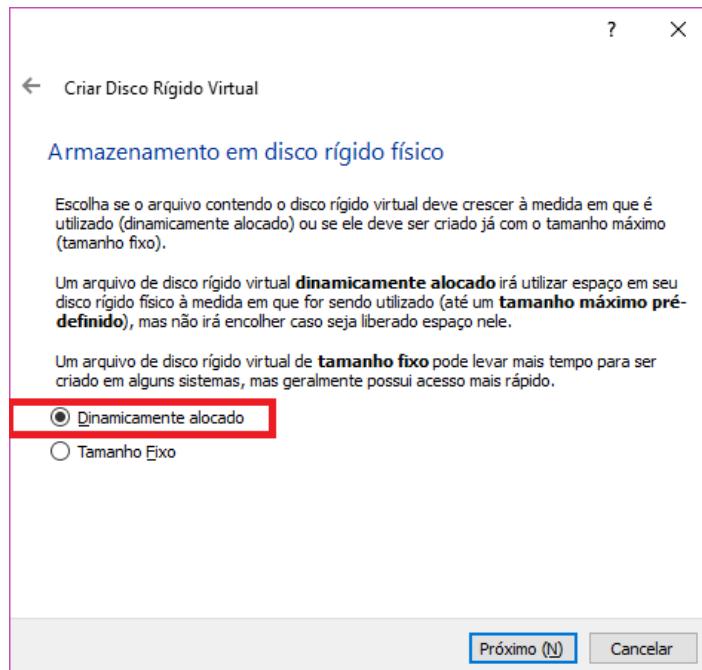


Figura 57 Criação de VM (parte 6).

Um espaço de 25 Gigabytes foi reservado para esta máquina virtual, como mostra a Figura 58.

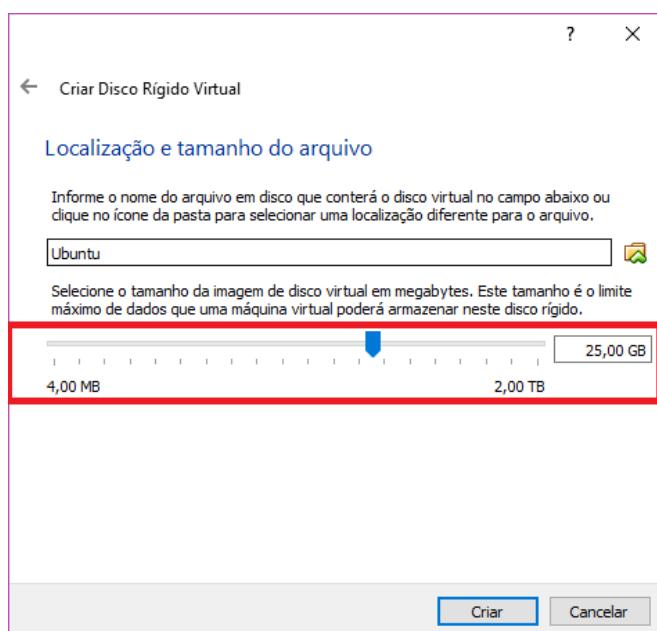


Figura 58 Criação de VM (parte 7).

4.4 INSTALAÇÃO DO SISTEMA OPERACIONAL

O Terceiro passo do projeto foi instalar o Sistema Operacional, e para isso, iniciou-se a máquina virtual criada, como mostra a Figura 59.

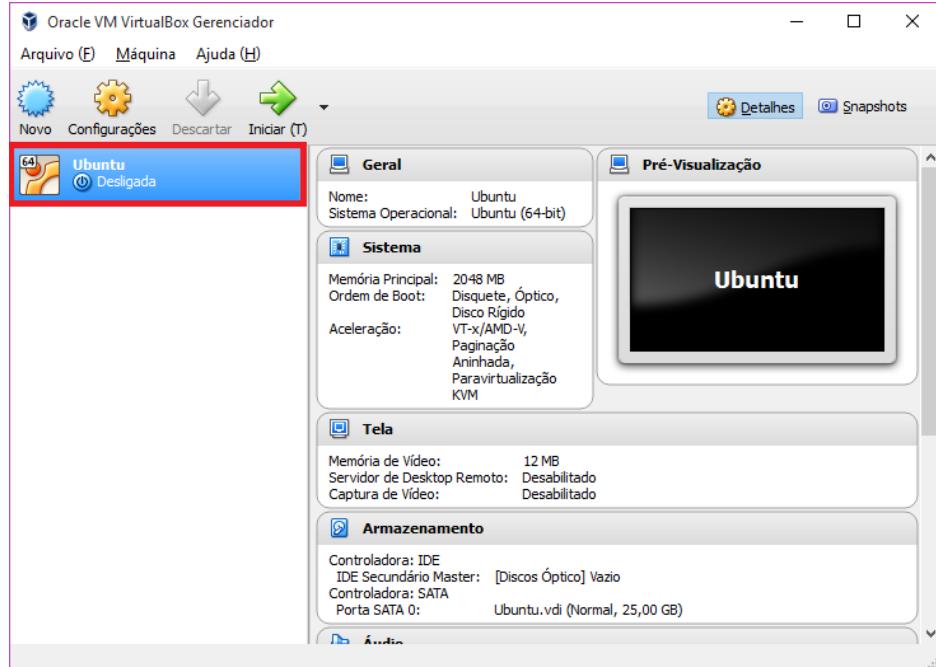


Figura 59 Instalação do Ubuntu (parte 1).

A Imagem ISO do sistema operacional foi selecionada, iniciando-se então a máquina, conforme as Figuras 60, 61 e 62.

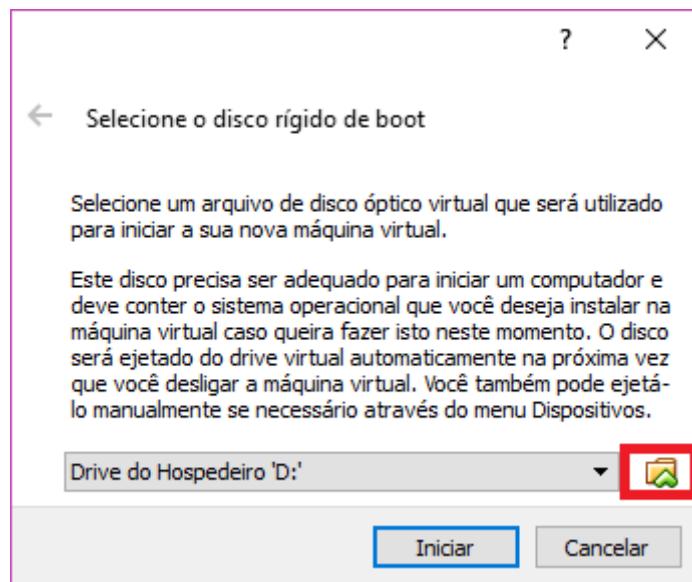


Figura 60 Instalação do Ubuntu (parte 2).

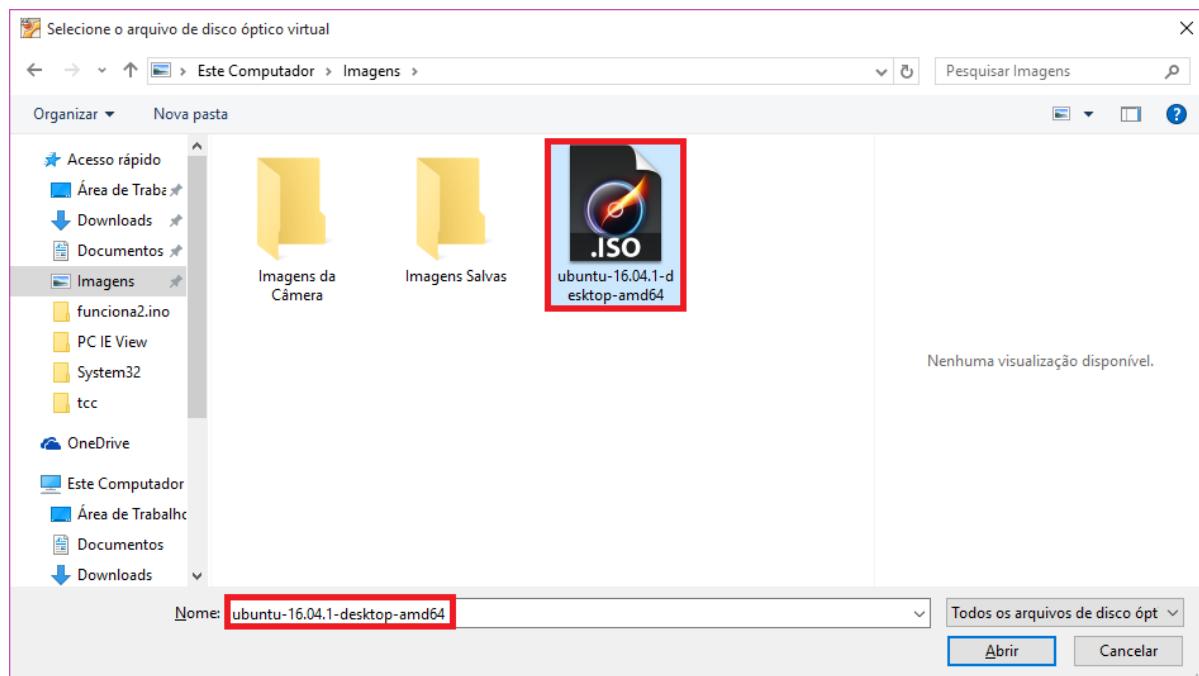


Figura 61 Instalação do Ubuntu (parte 3).

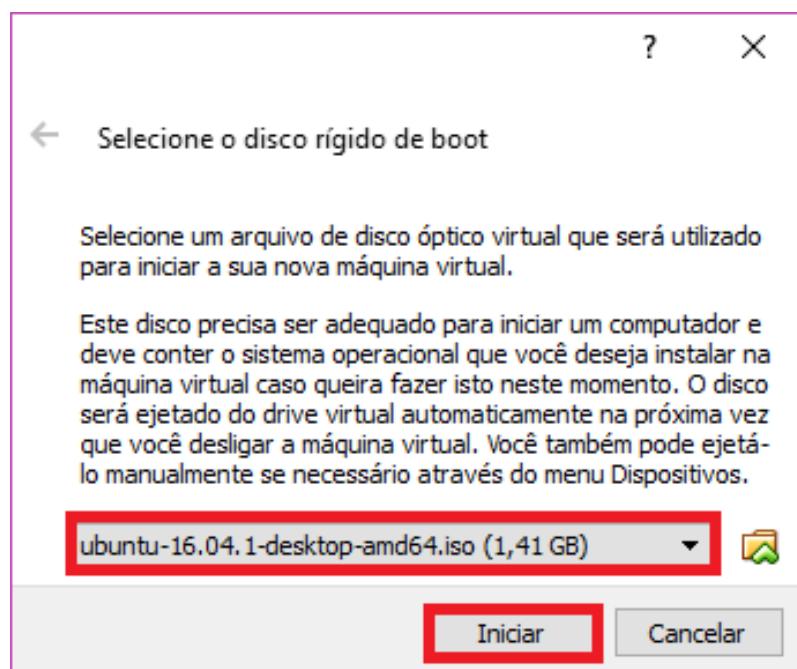


Figura 62 Instalação do Ubuntu (parte 4).

Após a seleção da Linguagem inicia-se a instalação do Ubuntu como na Figura 63.



Figura 63 Instalação do Ubuntu (parte 5).

As atualizações foram instaladas, durante essa etapa pouparando assim tempo. Para agilizar o processo, não foram instalados softwares de terceiros, como apresenta a Figura 64.

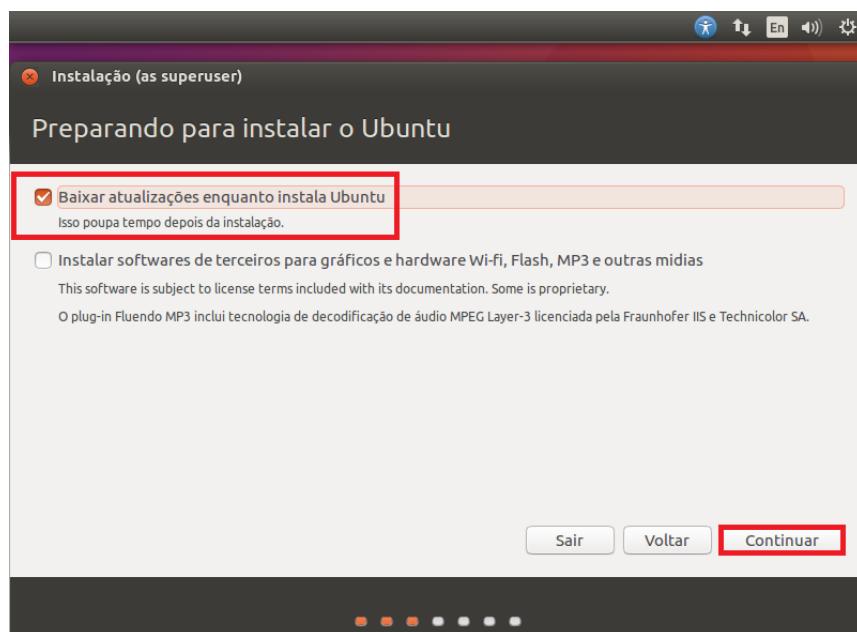


Figura 64 Instalação do Ubuntu (parte 6).

Na Figura 65, mostra que o disco foi formatado para garantir o sucesso da instalação.

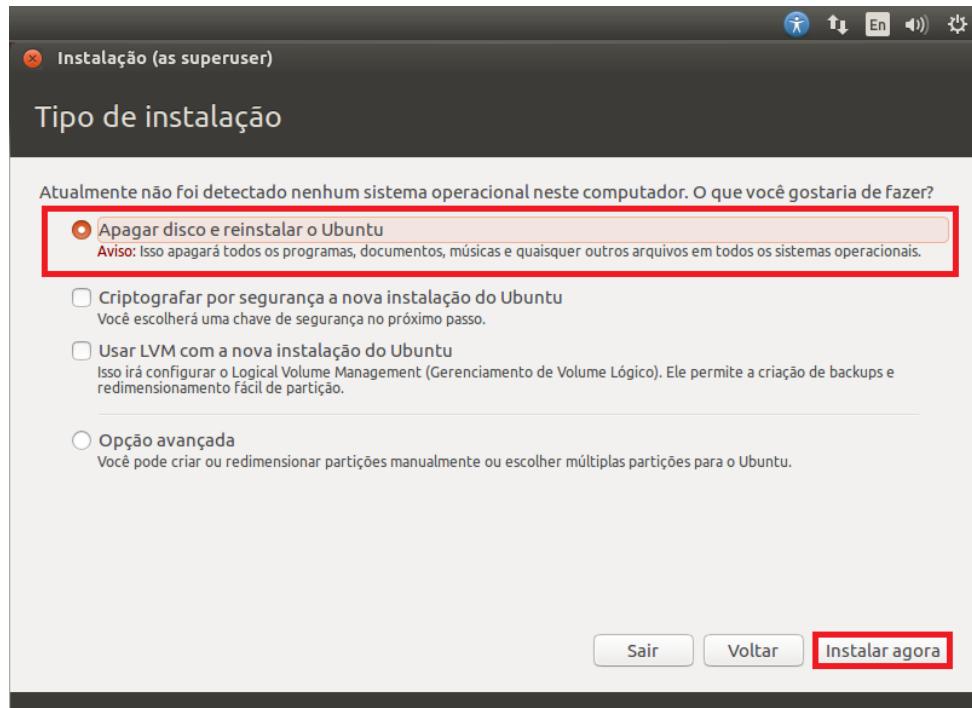


Figura 65 Instalação do Ubuntu (parte 7).

Para finalizar as configurações feitas durante a instalação, selecionou-se a localização da máquina, como na Figura 66, para o cálculo do fuso horário.

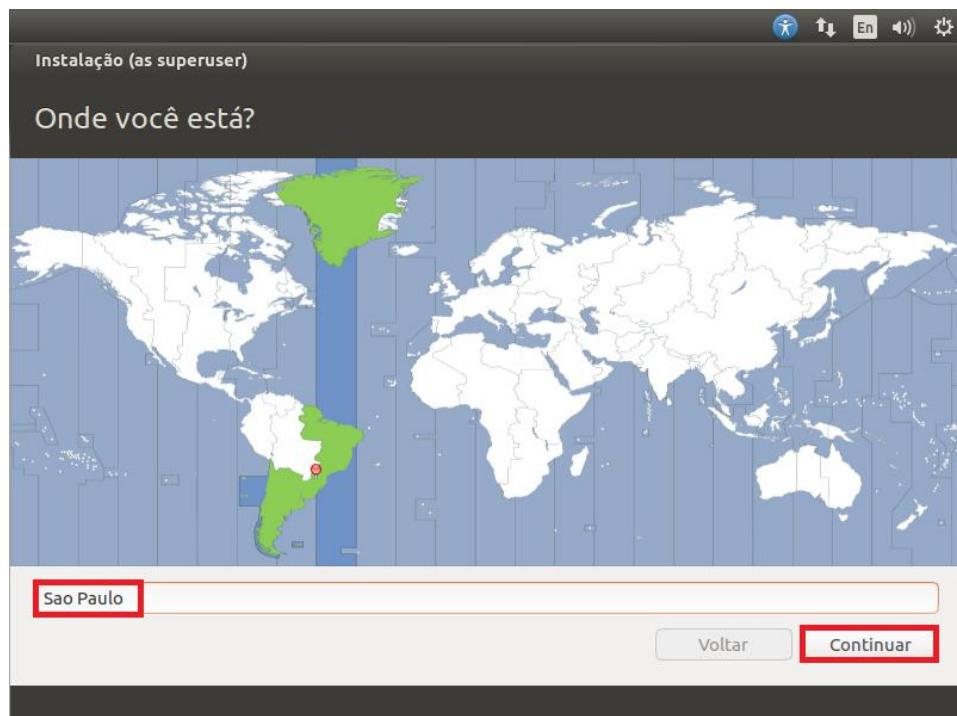


Figura 66 Instalação do Ubuntu (parte 8).

A Figura 67, mostra o final da instalação do sistema operacional.

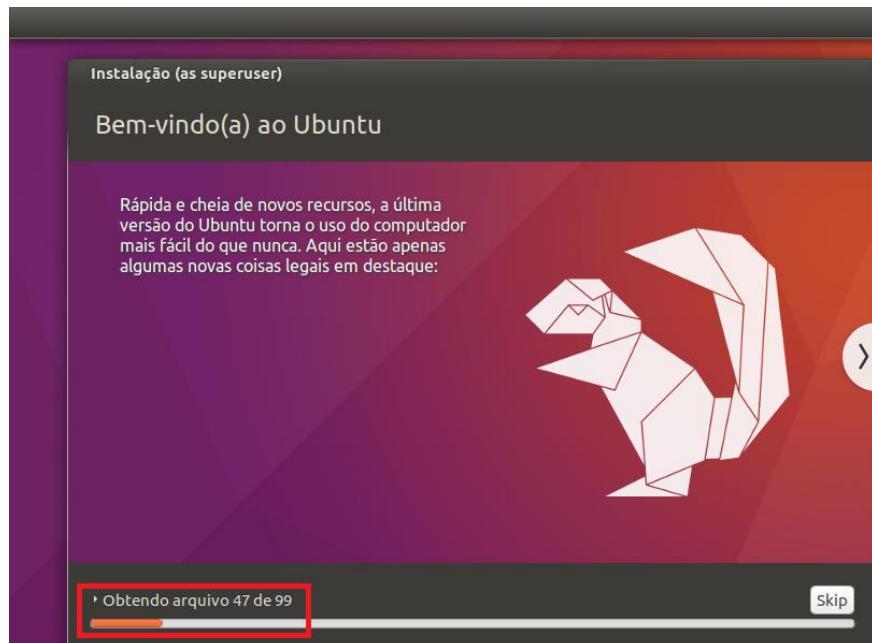


Figura 67 Instalação do Ubuntu (parte 9).

Com o Termo, o sistema foi iniciado, tendo em tela algo como na Figura 68.



Figura 68 Área de trabalho da máquina virtual.

4.5 INSTALAÇÃO DO LAMP-SERVER

Para a instalação do lamp-server utilizou-se os seguintes pacotes: apache2, php7.0, php7.0-mysql, libapache2-mod-php7.0 e mysql-server.

A Instalação desses pacotes foram feitas com o seguinte comando:

```
apt-get install apache2 php7.0 libapache2-mod-php7.0 mysql-server php7.0-mysql
```

Durante a instalação dos pacotes foi definida uma palavra-passe para o utilizador administrativo do MySQL, como mostra a Figura 69.

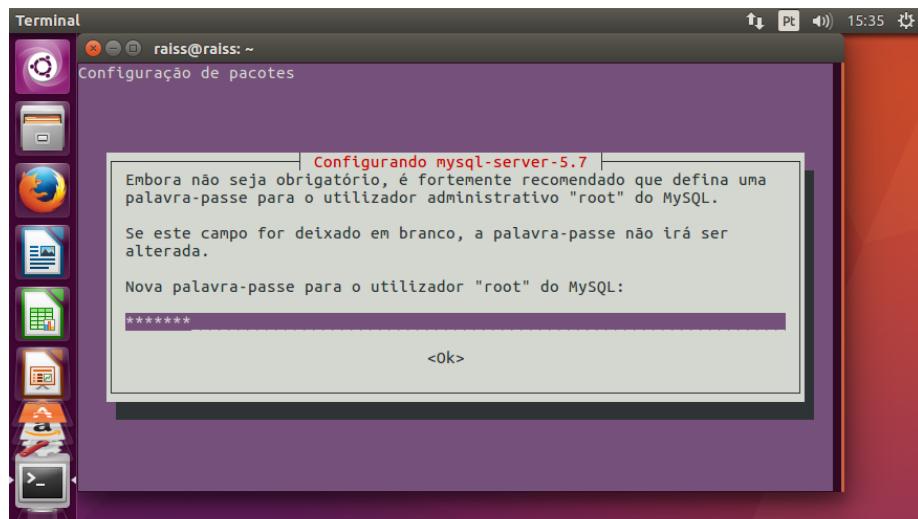


Figura 69 Instalação do LAMP.

A palavra passe foi repetida por segurança.Uma tela similar à da Figura 70 foi exibida ao final da instalação.

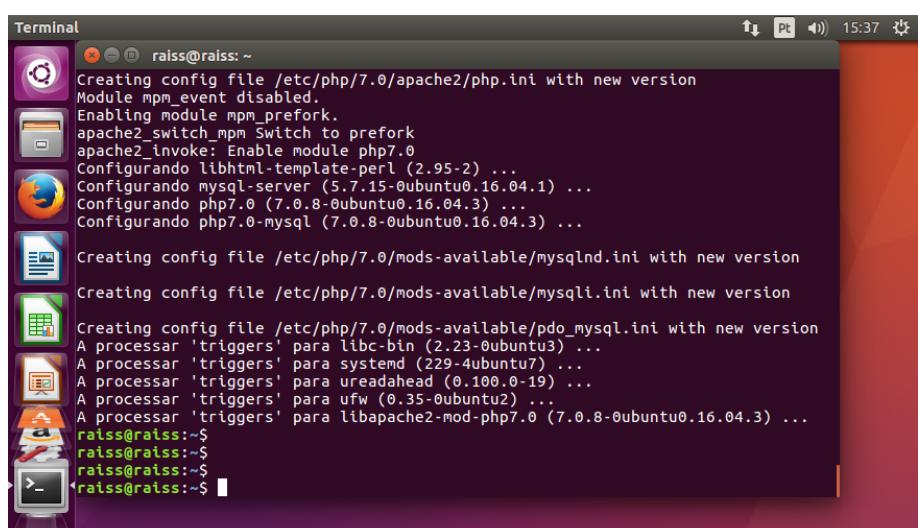


Figura 70 Finalização da Instalação do LAMP.

Para conferir o funcionamento do Apache, abriu-se o navegador e digitou-se na barra de endereço o IP local da máquina (que também poderia ser substituído por “*http://localhost*”), e o resultado foi o que mostra a Figura 71, mostrando que o servidor está funcionando corretamente.

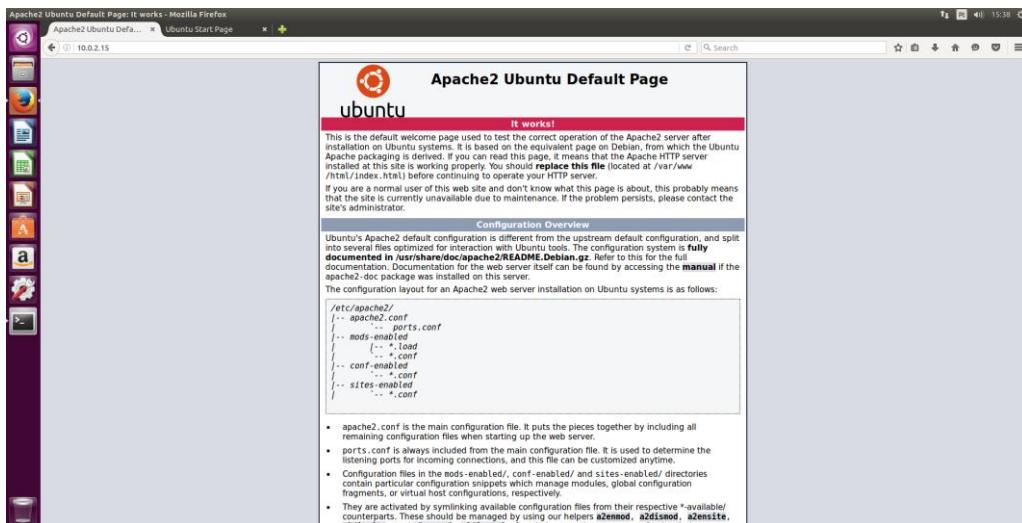


Figura 71 Resultado da Instalação do LAMP.

4.6 CRIAÇÃO DO BANCO DE DADOS NO MYSQL

Para o acesso ao MySQL no terminal do servidor, utilizou-se o

```
mysql -u root -p -h localhost
```

seguinte comando:

Foi digitada a senha de root do MySQL para fazer o acesso. Após isso digitou-se o comando conforme a Figura 72, para criar banco de dados *raiss* no servidor.

```
raiss@raiss:~$ mysql -u root -p -h localhost
Enter password:
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 7
Server version: 5.7.13-0ubuntu0.16.04.2 (Ubuntu)

Copyright (c) 2000, 2016, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> create database raiss;
Query OK, 1 row affected (0,00 sec)

mysql> ■
```

Figura 72 Criação de um banco de dados.

Ainda como root criou-se um novo usuário com permissão de acesso, de leitura e gravação ao banco de dados raiss, logo após foi encerrado o terminal do MySQL, como pode ser visto na Figura 73.

```
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 5.7.13-0ubuntu0.16.04.2 (Ubuntu)

Copyright (c) 2000, 2016, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

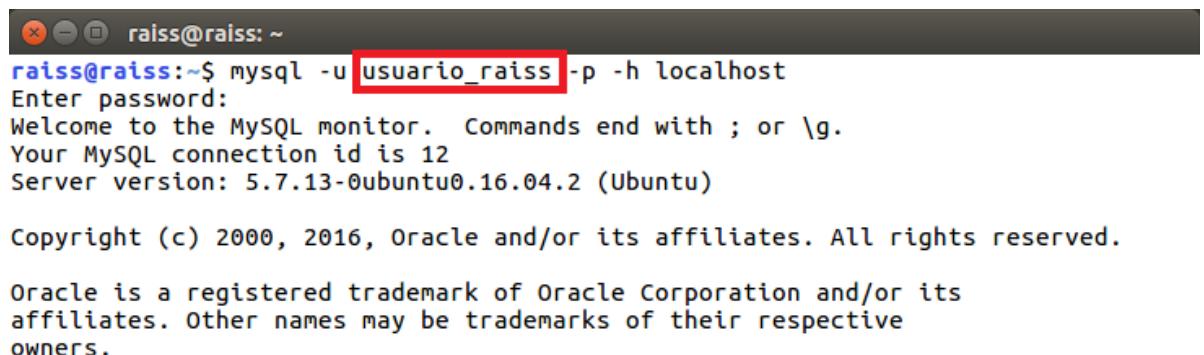
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> GRANT ALL PRIVILEGES ON raiss.* TO 'usuario_raiss'@localhost IDENTIFIED BY 'projeto' WITH GRANT OPTION;
Query OK, 0 rows affected, 1 warning (0,00 sec)

mysql> quit
Bye
```

Figura 73 Criação do usuário do banco de dados.

Novamente no terminal do servidor foi digitado o comando para abrir o terminal do MySQL, porém com o acesso do usuário criado anteriormente, como mostra a Figura 74.



```
raiss@raiss:~$ mysql -u usuario_raiss -p -h localhost
Enter password:
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 12
Server version: 5.7.13-0ubuntu0.16.04.2 (Ubuntu)

Copyright (c) 2000, 2016, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.
```

Figura 74 Acesso local ao MySQL.

Com o usuário atual acessou-se o banco de dados criado na sessão do MySQL anterior, e dentro dele foi criado uma tabela com os campos ID, Login, Senha, Nível, Ativo, Tentativas e Cadastro como pode ser observado na Figura 75.

- ID – Número de identificação do usuário;
- Login – Nome utilizado para o acesso do usuário;
- Senha – Senha utilizada para o acesso do usuário;
- Nível – Nível do usuário, o padrão é ‘1’ (usuário comum = ‘1’, usuário administrador = ‘2’);
- Ativo – Mostra se usuário está ativo, o padrão é ‘0’ (usuário inativo = ‘0’, usuário ativo = ‘1’);
- Tentativas – Número de tentativas sem sucesso de acesso do usuário (máximo de cinco tentativas);
- Cadastro – Data de cadastro do usuário.

```
mysql> use raiss;
Database changed
mysql> show tables;
Empty set (0,00 sec)

mysql> CREATE TABLE usuarios(
-> ID int(10) UNSIGNED NOT NULL AUTO_INCREMENT,
-> Login varchar(30) NOT NULL,
-> Senha varchar(80) NOT NULL,
-> Nivel int(1) UNSIGNED NOT NULL DEFAULT '1',
-> Ativo bool NOT NULL DEFAULT '0',
-> Tentativas int(1) UNSIGNED NOT NULL DEFAULT '0',
-> Cadastro datetime NOT NULL,
-> PRIMARY KEY ( ID ),
-> UNIQUE KEY Login ( Login )
-> DEFAULT CHARSET=utf8;
Query OK, 0 rows affected (0,30 sec)
```

Figura 75 Criação de tabela no MySQL.

Após criar a tabela inseriu-se uma linha na tabela recém-criada, como pode ser vista na Figura 76.

```
mysql> INSERT INTO usuarios ( Login, Senha, Nivel, Ativo, Cadastro )
-> VALUES ( 'Administrador', 'Senha', 2 , 1,NOW());
Query OK, 1 row affected (0,05 sec)

mysql> select * from usuarios;
+----+-----+-----+-----+-----+
| ID | Login      | Senha | Nivel | Ativo | Tentativas | Cadastro |
+----+-----+-----+-----+-----+
| 1  | Administrador | Senha |    2 |     1 |          0 | 2016-11-14 20:38:40 |
+----+-----+-----+-----+-----+
1 row in set (0,00 sec)

mysql> ■
```

Figura 76 Inserção de dados na tabela.

4.7 CRIAÇÃO DAS PÁGINAS DO SERVIDOR

Após a criação do LAMP server e da tabela no MySQL, foram feitas as páginas para controle do braço, para acesso e autenticação, e por final a página de gerência de usuários.

4.7.1 Página de acesso (index.html)

A página index.html localizada na página inicial do servidor é a página que envia os dados para o acesso, ela recebe o nome de utilizador e a senha do usuário e os envia através de um formulário, pelo método POST para uma página escrita em PHP. O código da página index.html pode ser visto nas Figuras 77 e 78.

Figura 77 Página de acesso (parte 1).

```
39 <div class="thumbnail12">
40   Somos estudantes do curso técnico de Redes de Computadores oferecido pelo SENAI - Santos Dumont de São José dos Campos, orientados pelos
41   professores Airton Cézar Zombardi, Josemar Monteiro da Silva, Kleber Gelli, Paulo Eduardo Gauvão e Wellington Carlos Joffre, e coordenados por José
42   Rogério Chavier.
43   <p></p>
44   Este site é parte de nosso Trabalho de Conclusão de Curso, é a interface web desenvolvida para controlar remotamente um Braço Robótico, que
45   funciona através de um modulo Wi-Fi ligado a um arduino, que esta implementado no robô. Esse é apenas um protótipo de braço que pode ser
46   aprimorado, e esse projeto pode ser adaptado a outros braços de difetenter portes com diferentes funções.
47   <p></p>
48   O projeto foi feito pensando nos diversos trabalhadores que se encontram em situações de risco diariamente em seu serviço, que realizarão o mesmo
49   serviço sem um contato com esses riscos a sua integridade, no bem estar e facilitado desses funcionários, que poderão trabalhar sem sair de suas
50   casas, e no crescimento dessas empresas, que poderão ter crescimento na produção e nao terão mais gastos com funcionários acidentados.
51   <p></p>
52   Equipe: André Lucas Maegima, Breno Henrique Borges Santos, Bruno dos Santos Mauricio, Guilherme Rios da Cunha e Leonardo Ribeiro dos Santos.
53   </div>
54 </div>
55 <footer>
56   <h3 class="logo">&copy;2016 - RAISS</h3>
57   <div>
58     <h3 class="logo">SENAI</h3>
59   </div>
60 </footer>
61 </body>
62 </html>
```

Figura 78 Página de acesso (parte 2).

A página gerada por esse código pode ser vista na Figura 79.

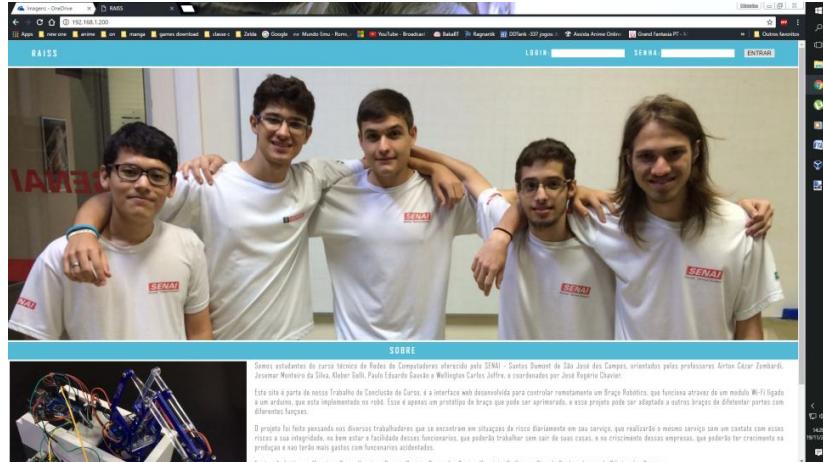


Figura 79 Layout da página de acesso.

A página login.php é a página que recebe os dados enviados do index.html, essa página é responsável por verificar no banco de dados se as informações recebidas são válidas, se forem então ela cria uma sessão para o usuário e o redireciona para a página principal do servidor. O código desta página pode ser visto na Figura 80.

```

1  <?php
2  include 'MySQL_func.php';
3  include 'vars.php';
4  session_start();
5  $link = conecta_mysql($servidor,$usuario,$senhadb,$db);
6  $tentativas = procura_mysql($link,'tentativas',$tabela,$login);
7  $ativo = procura_mysql($link,'ativo',$tabela,$login);
8  if( $tentativas === NULL || $ativo === NULL ){
9      mysqli_close($link);
10     header("location:/");
11 }
12 if( $tentativas >= 5 ){
13     mysqli_close($link);
14     echo "Número de tentativas excedidas.<br>\n";
15     exit(0);
16 }
17 if( $ativo === 0 ){
18     mysqli_close($link);
19     echo "Usuário Inativo.<br>\n";
20     exit(0);
21 }
22 if( $tentativas < 5 && $ativo == 1 ){
23     if( mysqli_real_query($link, "SELECT * FROM $tabela WHERE login = '$login' AND senha = '$senhaSH'")){
24         if( $result = mysqli_store_result($link) ){
25             echo $result->num_rows;
26             if($result->num_rows > 0){
27                 $_SESSION['Login'] = $login;
28                 $_SESSION['senha'] = $senha;
29                 mysqli_real_query($link, "UPDATE $tabela SET tentativas = 0 WHERE login = '$login' AND senha = '$senhaSH'");
30                 mysqli_close($link);
31                 header("location:/PHP/site.php");
32             }
33         else{
34             mysqli_real_query($link, "UPDATE $tabela SET tentativas = tentativas + 1 WHERE login = '$login'");
35             mysqli_close($link);
36             header("location:/");
37         }
38         mysqli_free_result($result);
39     }
40 }
41 }
42 else{
43     mysqli_close($link);
44     header("location:/");
45 }
?>
```

Figura 80 login.php.

Na segunda e terceira linhas são incluídas algumas funções criadas para trabalhar com o MySQL (MySQL_func.php), e algumas variáveis pré-definidas como \$servidor, \$usuario, \$senhadb e \$db (vars.php). O restante do código

trabalha usando funções para obter dados do servidor, compara-las com os dados recebidos pela página e tomar ações como enviar uma mensagem de erro ou redirecionar o usuário.

4.7.2 Página de controle do braço (site.php)

Após o usuário fazer o acesso ele é redirecionado para a página site.php, esta página inclui o verifica_con.php um script responsável em verificar se o usuário iniciou uma sessão em alguma página anterior, se a sessão não foi iniciada o script redireciona o usuário para a página de acesso. Esse script pode ser visto na Figura 81.

```

1 <?php
2 session_start();
3 if((!isset ($_SESSION['login']) == true) and (!isset ($_SESSION['senha']) == true))
4 {
5     unset($_SESSION['login']);
6     unset($_SESSION['senha']);
7     header('location:/');
8 }
9 $logado = $_SESSION['login'];
10 ?>

```

Figura 81 verifica_con.php.

A página de controle do braço possui *links* para páginas que enviam os comandos pré-definidos ao módulo WiFi que encaminha as instruções ao arduino que então move o robô. O código da página pode ser visto nas Figuras 82, 83 e 84.

```

1 <?php include "include/verifica_con.php"; ?>
2 <!DOCTYPE html>
3 <html>
4   <head>
5     <meta charset="utf-8">
6     <meta name="viewport" content="width=device-width, initial-scale=1">
7     <title>RAISS</title>
8     <link href="/css/singlePageTemplate.css" rel="stylesheet" type="text/css">
9   </head>
10 <body>
11   <header>
12     <h3 class="logo">INTERFACE PPARA CONTROLE DO BRAÇO ROBÓTICO</h3>
13     <nav>
14       <ul>
15         <li><?php echo "Logado como $logado"; ?></li>
16         <li>&ampnbsp&ampnbsp&bull;&ampnbsp&nbsp;</li>
17         <li><a href="registros.php">GERENCIAR USUÁRIOS</a></li>
18         <li>&ampnbsp&ampnbsp&bull;&ampnbsp&nbsp;</li>
19         <li><a href="include\logoff.php">SAIR</a></li>
20       </ul>
21     </nav>
22   </header>
23   <section class="hero">
24     <h2 class="hero_header">
25       <table>
26         <tr bgcolor="#52bad5">
27           <td width="33%">
28             <table>
29               <tr>
30                 <td width="50%"><a href="Comandos/Reseta.php" target="_blank"><div class="link">RESET</div></a></td>
31                 <td width="50%"> <a href="Comandos/Avanco-1.php" target="_blank"><div class="link">Avanço 50°</div></a></td>
32               </tr>
33               <tr>
34                 <td width="50%"><a href="Comandos/Garra-1.php" target="_blank"><div class="link">Garra Aberta</div></a></td>
35                 <td width="50%"> <a href="Comandos/Avanco-2.php" target="_blank"><div class="link">Avanço 100°</div></a></td>
36               </tr>
37             </table>
38           </td>
39         </tr>
40       </table>
41     </h2>
42   </section>
43 </body>
44 </html>

```

Figura 82 site.php (parte 1).

```

39      <tr>
40        <td> <a href="Comandos/Garra-2.php" target="_blank"><div class="link">Semi Aberta</div></a></td>
41        <td> <a href="Comandos/Avanco-3.php" target="_blank"><div class="link">Avanco 150°</div></a></td>
42      </tr>
43      <tr>
44        <td> <a href="Comandos/Garra-3.php" target="_blank"><div class="link">Garra Fechada</div></a></td>
45        <td> <a href="Comandos/Avanco-4.php" target="_blank"><div class="link">Avanco 200°</div></a></td>
46      </tr>
47    </table>
48  </td>
49  <td width="28.5%">
50    <iframe src="http://192.168.0.100:8080/browserfs.html"></iframe>
51  </td>
52  <td width="33%">
53    <table>
54      <tr>
55        <td width="50%"> <a href="Comandos/Mergulho-1.php" target="_blank"><div class="link2">Mergulho 0°</div></a></td>
56        <td width="50%"> <a href="Comandos/Base-1.php" target="_blank"><div class="link2">Base 0°</div></a></td>
57      </tr>
58      <tr>
59        <td> <a href="Comandos/Mergulho-2.php" target="_blank"><div class="link2">Mergulho 25°</div></a></td>
60        <td> <a href="Comandos/Base-2.php" target="_blank"><div class="link2">Base 45°</div></a></td>
61      </tr>
62      <tr>
63        <td> <a href="Comandos/Mergulho-3.php" target="_blank"><div class="link2">Mergulho 50°</div></a></td>
64        <td> <a href="Comandos/Base-3.php" target="_blank"><div class="link2">Base 90°</div></a></td>
65      </tr>
66      <tr>
67        <td> <a href="Comandos/Mergulho-4.php" target="_blank"><div class="link2">Mergulho 75°</div></a></td>
68        <td> <a href="Comandos/Base-4.php" target="_blank"><div class="link2">Base 135°</div></a></td>
69      </tr>
70      <tr>
71        <td> <a href="Comandos/Mergulho-5.php" target="_blank"><div class="link2">Mergulho 100°</div></a></td>
72        <td> <a href="Comandos/Base-5.php" target="_blank"><div class="link2">Base 180°</div></a></td>
73      </tr>
74    </table>
75  </td>
76  </tr>
77 </table>
78 </h2>
79 </section>
80

```

Figura 83 site.php (parte 2).

```

81  <div class="galleryt">
82    <div class="thumbnail">
83      <h3>GARRA</h3>
84    </div>
85    <div class="thumbnail">
86      <h3>AVANÇO</h3>
87    </div>
88    <div class="thumbnail">
89      <h3>MERGULHO</h3>
90    </div>
91    <div class="thumbnail">
92      <h3>BASE</h3>
93    </div>
94  </div>
95  <div class="gallery">
96    <div class="thumbnail">
97      
98    </div>
99    <div class="thumbnail">
100      
101    </div>
102    <div class="thumbnail">
103      
104    </div>
105    <div class="thumbnail">
106      
107    </div>
108  </div>
109 </div>
110 <h3 class="logo">INTERFACE PARA CONTROLE DO BRAÇO ROBÓTICO</h3>
111 <nav>
112   <ul>
113     <li><?php echo "Logado como $logado"; ?></li>
114     <li>&ampnbsp&ampnbsp&nbsp;&nbsp;</li>
115     <li><a href="registros.php">GERENCIAR USUARIOS</a></li>
116     <li>&ampnbsp&ampnbsp&nbsp;&nbsp;</li>
117     <li><a href="include/logout.php">SAIR</a></li>
118   </ul>
119 </nav>
120 </div>
121 </body>
122 </html>
123

```

Figura 84 site.php (parte 3).

Na Figura 85 pode ser visto a página gerada pelo código acima.

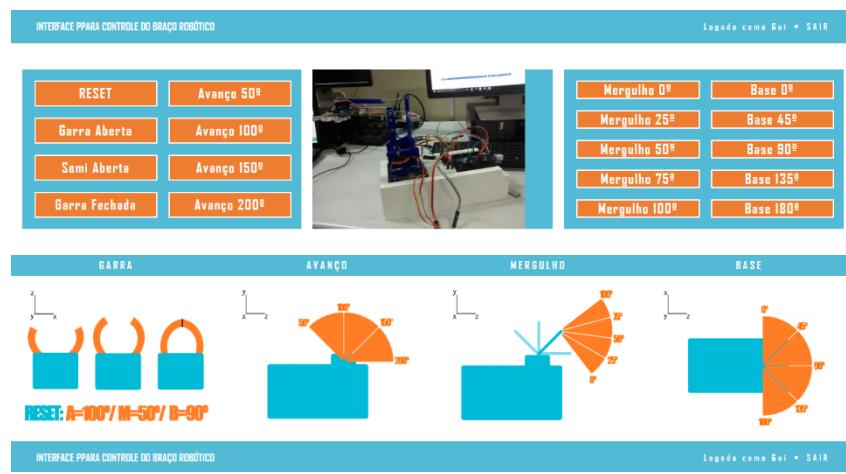


Figura 85 Layout do site.php.

4.7.3 Página de gerencia de usuários (registros.php)

A página para a gerência de usuários registros.php, inclui em seu cabeçalho o arquivo php gerenciar.php, este arquivo é responsável em verificar se o usuário possui ou não permissão para mudar as definições de outros usuários.

Caso o usuário tenha a permissão para alterar os dados da tabela de usuários, a página registros.php é carregada, caso contrário o usuário recebe uma mensagem de erro e é redirecionado para a página site.php. O script gerenciar.php pode ser visto nas Figuras 86, 87 e 88.

```

1  <?php
2  include 'verifica_con.php';
3  include 'MySQL_func.php';
4  include 'vars.php';
5  $link = conecta_mysql($servidor,$usuario,$senhadb,$db);
6  $nivel = procura_mysql($link, 'nivel', $tabela, $logado);
7  if($nivel !== 2){
8      mysqli_close($link);
9      echo "Erro usuário não tem permissão para acessar página\n";
10     echo "<meta http-equiv='refresh' content='5;URL=/PHP/site.php'>\n";
11     exit(0);
12 }
13 $login_derr="";
14 if($_SERVER['REQUEST_METHOD'] == 'POST'){
15     if( !empty($login_d) ){
16         if(!preg_match("/^[_a-zA-Z0-9_-]+$/",$login_d)){
17             $login_derr = "Erro: Caracter Inválido.\n";
18         }
19         if( !in_db($link,$login_d) && empty($login_derr) ){
20             $login_derr = "Erro: Usuário não existe.\n";
21         }
22         if( empty($login_derr) ){
23             del_user_db($link,$login_d);
24         }
25     }
26 }
```

Figura 86 gerenciar.php (parte 1).

```

27 if( $_POST['login'] != NULL ){
28     if($_SERVER['REQUEST_METHOD'] == 'POST'){
29         $login_err="";
30         if(!preg_match("/^[_a-zA-Z0-9_-]+$/,$login)){
31             $login_err = "Erro: Caracter Inválido.\n";
32         }
33         if( !in_db($link,$login) && empty($login_err) ){
34             $login_err = "Erro: Usuário não existe.\n";
35         }
36         if( empty($login_err) ){
37             mysqli_real_query($link, "UPDATE $tabela SET tentativas = 0 WHERE login = '$login'");
38             if($lv == '2' || $lv == '1'){
39                 if(!empty($login)){
40                     mysqli_real_query($link, "UPDATE $tabela SET nivel = $lv WHERE login = '$login'");
41                 }
42             }
43             if($ativa == '1' || $ativa == '0'){
44                 if(!empty($login)){
45                     mysqli_real_query($link, "UPDATE $tabela SET ativo = $ativa WHERE login = '$login'");
46                 }
47             }
48         }
49     }
50 }
```

Figura 87 gerenciar.php (parte 2).

```

51 if( $_POST['login_a'] != NULL ){
52     $senha_aerr=$login_aerr=$confirm_aerr="";
53     if($_SERVER['REQUEST_METHOD'] == 'POST'){
54         if( !empty($login_a) ){
55             if(!preg_match('/^([a-zA-Z0-9-_]+$/',$login_a)){
56                 $login_aerr = "Erro: Caracter Inválido.\n";
57             }
58             if( in_db($link, $login_a)){
59                 $login_aerr = "Erro: Nome de usuario '$login_a' já existe.\n";
60             }
61         }
62     }else{
63         $login_aerr = "Erro: Campo login não pode estar vazio.\n";
64     }
65     if( empty($senha) ){
66         $senha_aerr = "Erro: Campo senha não pode estar vazio.\n";
67     }
68     if( $confirm !== $senha ){
69         $confirm_aerr = "Erro: Senhas não coincidem.\n";
70     }
71     if( empty($login_aerr) && empty($senha_aerr) && empty($confirm_aerr) ){
72         $comando = "INSERT INTO usuarios ( login , senha , cadastro ) VALUES ( '$login_a' , '$senhaSH' , NOW() )";
73         if(!mysql_real_query($link,$comando)){
74             if( $link->errno == "1062" ){
75                 $login_aerr = "Erro: Nome de usuario '$login_a' já existe.\n";
76             }
77         }
78     }
79 }
80 ?>

```

Figura 88 gerenciar.php (parte 3).

O script acima tem também como função receber os dados enviados pela página registros.php e executar uma ação de acordo com eles. O código do registros.php pode ser visto na Figura 89.

```

1 <?php
2 include 'include/gerenciar.php';
3 ?>
4 <!DOCTYPE html>
5 <html>
6 <head>
7     <title>Registros Usuarios</title>
8     <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
9     </head>
10 <body>
11     <h1>Registros Usuarios</h1>
12     <!-- Mostra tabela com os dados dos usuarios, classe da tabela usuarios -->
13     <?php TabelaUsuarios($link,$tabela); ?>
14     <!-- Daqui pra baixo os php são para mostrar erros se houverem não remover -->
15     <h2>Adicionar novo banco de dados:</h2>
16     <form action=<?php echo $self; ?> method="post" >
17         Inform seu Login
18         <input class="navy-tema" type="text" name="login_a" >
19         * <?php echo $login_aerr; ?><br>
20         Inform sua Senha:
21         <input class="navy-tema" type="password" name="senha" >
22         * <?php echo $senha_aerr; ?><br>
23         Confirme a Senha :
24         <input class="navy-tema" type="password" name="confirm" >
25         * <?php echo $confirm_aerr; ?><br>
26         <input type="submit" ><br>
27     </form>
28     <h2>Mudar permissões do usuário:</h2>
29     <form action=<?php echo $self; ?> method="post" >
30         Inform o Login do Usuario:<input class="navy-tema" type="text" name="login" >
31         * <?php echo $login_err; ?><br>
32         <input type="radio" name="nivel" value="1" checked>Usuario comun <br>
33         <input type="radio" name="nivel" value="2" >Administrador <br>
34         <input type="radio" name="ativa" value="1" checked>Ativa <br>
35         <input type="radio" name="ativa" value="0" >Desativa <br>
36         <input type="submit" value="enviar" >
37     </form>
38     <h2>Remover usuário:</h2>
39     <form action=<?php echo $self; ?> method="post" >
40         Inform o Login do Usuario a ser removido:
41         <input class="navy-tema" type="text" name="login_d" >
42         * <?php echo $login_derr; ?><br>
43         <input type="submit" >
44     </form>
45     <a href="site.php">Clique aqui para voltar para página principal</a><br>
46 </body>
47 </html>

```

Figura 89 registros.php.

A função “TabelaUsuarios(\$link,\$tabela)”, tem a função de todos os campos da tabela de usuários, isto é, mostrar na página a tabela contendo todos os campos de dados dos usuários no banco de dados. Logo após mostrar a tabela de usuários, a página mostra as opções de gerenciamento como adicionar, mudar a permissão de acesso ou excluir um usuário. Na Figura 90 pode ser vista a página no navegador.

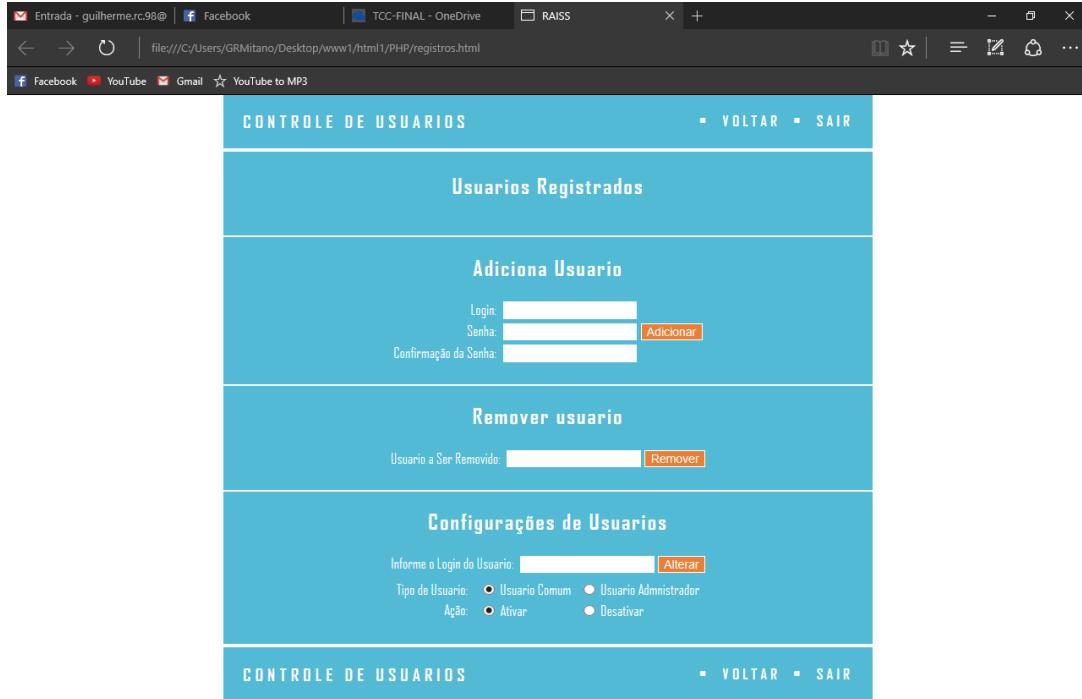


Figura 90 Layout do registros.php.

4.7.4 Páginas para envio de comando ao servidor

As páginas para envio de comandos ao servidor foram criadas para fazer a conexão com uma porta de determinado servidor e enviar um pré-definido comando a ele. Essas páginas possuem em seu cabeçalho um arquivo php que verifica se há uma sessão iniciada para então conectar ao servidor e enviar a mensagem especificada em seu corpo. O código de um exemplo do arquivo php pode ser visto na Figura 91.

```

1 <?php
2 include "include/verifica_con.php";
3 error_reporting(E_ALL);
4 $socket = socket_create(AF_INET, SOCK_STREAM, SOL_TCP);
5 if ($socket === false) {
6     echo "Erro: " . socket_strerror(socket_last_error());
7     exit(0);
8 }
9 $address = "127.0.0.1";
10 $service_port = 2500;
11 $result = socket_connect($socket, $address, $service_port);
12 if ($result === false) {
13     echo "Erro: ($result) " . socket_strerror(socket_last_error($socket));
14     exit(0);
15 }
16 $in = "Exemplo\r\n";
17 socket_write($socket, $in, strlen($in));
18 socket_close($socket);
19 echo '<script>window.close();</script>';
20 header('location:../site.php');
21 ?>
```

Figura 91 Página que envia comando.

4.8 COMUNICAÇÃO DO SERVIDOR COM O BRAÇO ROBÓTICO

Para a comunicação do braço robótico com o servidor foi utilizado um servidor TCP/IP. O servidor abre uma porta TCP e espera a primeira conexão, após ela ser feita o servidor aguarda uma segunda conexão e espera uma mensagem dessa segunda conexão, o servidor então envia essas mensagens para a primeira conexão, esse programa foi escrito na linguagem C o início de seu código pode ser visto na Figura 92.

```

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4 #include <unistd.h>
5 #include <sys/types.h>
6 #include <sys/socket.h>
7 #include <netinet/in.h>
8 #include <arpa/inet.h>
9 #include <pthread.h>
10 #define MAX 2
11
12 int on = 0, sockfd;
13
14 typedef struct{
15     int id;
16     int value;
17 } TipoCliente;
18
19 typedef struct{
20     TipoCliente data[MAX];
21 } cliente;

```

Figura 92 servidor.c (inclusão das bibliotecas/definição das estruturas).

A figura acima mostra a inclusão das bibliotecas, a criação de duas variáveis globais inteiras e a definição das estruturas TipoCliente e cliente. Após a criação das estruturas foram criadas também as diversas funções, Tabela 3, para facilitar o gerenciamento e manutenção do programa.

Função	Descrição
void error(const char *msg)	Imprime na tela uma mensagem de erro termina o programa.
void inicializaCliente(cliente *sock)	Inicializa as variáveis da estrutura cliente.
void abreSocket(int *sockfd)	Abre um socket no modo <i>stream</i> .
void amarraSocket(int *sockfd, int portno)	Amarra o socket a um endereço e a uma porta do servidor.
void clienteSocket(TipoCliente *cliente)	Espera uma conexão de um cliente.
void mensagemInicial(TipoCliente *cliente)	Envia uma mensagem a um cliente quando após a conexão ser feita.
int timeoutConexao(struct timeval *timeout, fd_set *u_set)	Espera uma mensagem vindo do cliente por um certo período.
int verificaConexao(TipoCliente *cliente, int conexao)	Verifica se o cliente está conectado.
void fechaConexao(TipoCliente *cliente)	Fechá a conexão do cliente com o servidor.
void *connection(void *sock_void)	Função que é chamada quando uma nova conexão é requisitada.
void novaConexao(cliente *client, int id)	Inicia uma nova conexão em um novo processo do sistema.
void envia(cliente *client)	Envia as mensagens do cliente 2 para o

	cliente 1.
--	------------

Tabela 3 Funções do programa servidor.c.

A Figura 93, mostra as funções error, inicializaCliente e abreSocket.

```

22 void error(const char *msg){
23     perror(msg);
24     exit(1);
25 }
26
27 void inicializaCliente(cliente *sock){
28     int i;
29     for(i = 0; i < MAX; i++){
30         sock->data[i].id = -1;
31         sock->data[i].value = -1;
32     }
33 }
34
35
36 void abreSocket(int *sockfd){
37     int optval = 1;
38     *sockfd = socket(AF_INET, SOCK_STREAM, 0);
39     if (*sockfd < 0) error("ERRO ao abrir socket");
40     setsockopt(*sockfd, SOL_SOCKET, SO_REUSEADDR,
41                 (const void *)&optval, sizeof(int));
42 }
43

```

Figura 93 Funções erro, inicializaCliente e abreSocket.

Na Figura 94, pode ser vistas as funções amarraSocket e clienteSocket.

```

44 void amarraSocket(int *sockfd, int portno){
45     struct sockaddr_in serv_addr;
46     bzero((char *) &serv_addr, sizeof(serv_addr));
47     serv_addr.sin_family = AF_INET;
48     serv_addr.sin_addr.s_addr = INADDR_ANY;
49     serv_addr.sin_port = htons(portno);
50     if (bind(*sockfd, (struct sockaddr *) &serv_addr, sizeof(serv_addr)) < 0)
51         error("ERRO ao estabelecer ligacao");
52 }
53
54 void clienteSocket(TipoCliente *cliente){
55     socklen_t clilen;
56     struct sockaddr_in cli_addr;
57     listen(sockfd, 5);
58     clilen = sizeof(cli_addr);
59     cliente->value = accept(sockfd, (struct sockaddr *) &cli_addr, &clilen);
60     if (cliente->value < 0) error("ERRO ao aceitar");
61     printf("nova conexao %s porta:%hu id %d\n",
62           inet_ntoa(cli_addr.sin_addr), cli_addr.sin_port, cliente->id);
63 }
64

```

Figura 94 Funções abreSocket e clienteSocket.

As funções mensagemInicial e timeoutConexao podem ser vistas na Figura 95.

```

65 void mensagemInicial(TipoCliente *cliente){
66     int n;
67     char msg[50];
68     sprintf(msg, "Bem-vindo ao servidor TCP/IP, o seu id e %d.\n", cliente->id);
69     n = write(cliente->value, msg, strlen(msg));
70     if (n < 0) error("ERRO ao escrever no socket");
71     n = write(cliente->value, "Para sair digite 'quit' ou digite 'shutdown'", 44);
72     if (n < 0) error("ERRO ao escrever no socket");
73     n = write(cliente->value, "para desconectar o servidor.\n\n", 30);
74     if (n < 0) error("ERRO ao escrever no socket");
75 }
76
77 int timeoutConexao(struct timeval *timeout, fd_set *u_set){
78     int n;
79     timeout->tv_sec = 10;
80     timeout->tv_usec = 0;
81     n = select(FD_SETSIZE, u_set, NULL, NULL, timeout);
82     return n;
83 }
84

```

Figura 95 Funções mensagemInicial e timeoutConexao.

Na Figura 96, pode ser visto a definição das funções verificaConexao e fechaConexao.

```

85 int verificaConexao(TipoCliente *cliente, int conexao){
86     int n;
87     char buffer[2048];
88     bzero(buffer, 2048);
89     if (conexao == -1) error("ERRO select");
90     else if (conexao == 0){
91         printf("%d timed out\n", cliente->id);
92         return 0;
93     }
94     else{
95         n = read(cliente->value, buffer, 2048);
96         if (n < 0) error("ERRO ao ler socket");
97         if( strcmp(buffer, "VIVO\r\n") == 0 ){
98             printf("V_OK\n");
99             return 1;
100        }
101    }
102    return 0;
103 }
104
105 void fechaConexao(TipoCliente *cliente){
106     close(cliente->value);
107     cliente->id = -1;
108     cliente->value = -1;
109     on--;
110 }
111

```

Figura 96 Funções VerificaConexao e fechaConexao.

As funções *connection* e *novaConexao* podem ser vistas na Figura 97.

```

112 void *connection(void *sock_void){
113     int u_on = 1, conexao;
114     struct timeval timeout;
115     fd_set u_set;
116     TipoCliente *newsock = (TipoCliente *)sock_void;
117     clienteSocket(newsock);
118     mensagemInicial(newsock);
119     FD_ZERO(&u_set);
120     FD_SET(newsock->value, &u_set);
121     if( newsock->id == 1 ){
122         do{
123             conexao = timeoutConexao(&timeout, &u_set);
124             u_on = verificaConexao(newsock, conexao);
125         }
126         while(u_on);
127         fechaConexao(newsock);
128     }
129     return NULL;
130 }
131
132 void novaConexao(cliente *client, int id){
133     pthread_t sock_thread;
134     printf("%d, %d\n",client->data[id].id,id);
135     client->data[id].id = id;
136     if(pthread_create(&sock_thread, NULL, connection, &client->data[id]))
137         error("Erro ao criar Thread");
138     on++;
139 }
140

```

Figura 97 Funções *connection* e *novaConexao*.

Na Figura 98, pode ser vista a função *envia*.

```

141 void envia(cliente *client){
142     char mensagem[25];
143     int n, len;
144     bzero(mensagem,25);
145     if( client->data[0].id != -1 && client->data[0].value != -1){
146         n = read(client->data[0].value,mensagem,25);
147         if (n < 0) error("ERRO ao ler socket 0");
148         len = strlen(mensagem);
149         printf("len: %d\n",len);
150         if( client->data[1].id != -1 && client->data[1].value != -1){
151             if(len < 25){
152                 n = write(client->data[1].value,mensagem,len);
153                 if (n < 0) error("ERRO ao escrever no socket");
154                 printf("mensagem len: %d enviada.\n",len);
155             }
156             else{
157                 printf("Muito grande\n\n");
158             }
159         }
160         fechaConexao(&client->data[0]);
161     }
162 }
163

```

Figura 98 função *envia*.

A parte principal do programa o *main* tem como parâmetro *int arg* que guarda a quantidade de parâmetros fornecidos ao programa e *char *argv[]* um ponteiro que guarda as *strings* referentes a esses parâmetros, a Figura 99 mostra o *main* do programa arquivo *servidor.c*.

```

164 int main(int argc, char *argv[])
165 {
166     int portno;
167     cliente client;
168     if (argc < 2) {
169         fprintf(stderr,"ERRO, nenhuma porta fornecida\n");
170         exit(1);
171     }
172     portno = atoi(argv[1]);
173     inicializaCliente(&client);
174     abreSocket(&sockfd);
175     amarraSocket(&sockfd, portno);
176     do{
177         envia(&client);
178         if( client.data[1].id == -1 && client.data[1].value == -1 && on < MAX ){
179             novaConexao(&client,1);
180         }
181         if( client.data[1].id > -1 && client.data[1].value > -1 && on < MAX ){
182             if( client.data[0].id == -1 && client.data[0].value == -1 && on < MAX ){
183                 novaConexao(&client,0);
184             }
185         }
186     }while(1);
187     close(sockfd);
188     return 0;
189 }
190

```

Figura 99 main do arquivo servidor.c.

A comunicação do braço robótico com o servidor é feita a través do módulo WiFi conectado ao arduino, o arduino Mega manda e recebe as mensagens do módulo através de suas portas seriais. Para a conexão do módulo ao servidor foi criado um programa no arduino para este se conectar ao servidor através do módulo WiFi para receber os comandos movimentar o braço.

O módulo primeiramente tenta-se conectar à rede fornecida pelo programa do arduino e logo após ele se conecta e fica esperando as mensagens vindas do servidor. O arduino constantemente checa a serial conectada ao módulo para ler o *buffer* de dadosse ele estiver disponível. A Figura 100 mostra a inclusão das bibliotecas utilizada e definição das funções do programa braco_wifi.ino.

```

#include <Servo.h>
#define DEBUG true
#define NO_DEBUG false
#define L_RED 11
#define L_GREEN 12

bool conectado = false;
bool erro = false;
Servo base, avanco, mergulho, garra;
float baseP, avancoP, mergulhoP, garraP, Delay;

void pisca(int led, const int timeout);
String getString(String padrao, String str);
String RespostaServidor(String str);
String sendData(String command, const int timeout, boolean debug);
String conectaRoteador(String ssid, String senha);
boolean iniciaConexaoTCP(String ip, String porta, boolean debug);
boolean checaConexao(String *conn);
String enviaMensagem(String mensagem);
String recebeMensagem(boolean debug);
void Alive();
void Desconecta(String *conn);
void startServos();
void moveServo(Servo *servo, int posicao);
void posiciona(Servo *motor1, Servo *motor2, Servo *motor3, float np1, float np2, float np3);

```

Figura 100 braco_wifi.ino.

A função pisca, é utilizada para fazer um led conectado na porta fornecida piscar.

As funções “getString” e “RespostaServidor”, são funções que recebem uma *string* corta partes dela de acordo com um padrão e retorna.

A função “sendData”, envia um comando para o módulo WiFi e retorna resposta como uma *string*.

A função “conectaRoteador”, faz a conexão a um *acess point*.

A função “iniciaConexao”, inicia uma conexão TCP com o servidor especificado.

A função “checaConexao”, verifica a conexão do módulo com o servidor.

As funções “enviaMensagem” e “recebeMensagem”, enviam e recebem mensagens através do módulo respectivamente.

A função “Alive”, manda uma mensagem para mostrar ao servidor que a conexão está ativa.

A função “Desconecta”, tem a função de desconectar o módulo do servidor.

A função “startServos”, define as portas de controle dos servos motores.

As funções “moveServo” e posicionar, são utilizadas para controlar os servos motores.

O *setup* do arduino inicia as Seriais, define os pinos que serão utilizados para os leds, reinicia as configurações do módulo e tenta iniciar a comunicação com o servidor. A Figura 101 mostra o código do *setup* do arduino.

```

void setup() {
    Serial.begin(9600);
    Serial2.begin(115200);
    pinMode(L_GREEN, OUTPUT);
    pinMode(L_RED, OUTPUT);
    sendData("AT+RST\r\n", 2000, DEBUG); // rst
    String AP = conectaRoteador("dlink", "12345678");
    if (AP != "OK") {
        Serial.println("Erro na Conexao com o roteador.");
        erro = true;
    }
    else {
        long int tempo = millis() + 30000;
        while (!conectado && tempo > millis()) {
            sendData("AT+CIPCLOSE\r\n", 1000, DEBUG);
            pisca(L_GREEN, 3000);
            conectado = iniciaConexaoTCP("192.168.0.200", "2500", DEBUG);
        }
        if (conectado) {
            digitalWrite(L_GREEN, HIGH);
            startServos();
            Delay = 50;
        }
        else erro = true;
    }
}

```

Figura 101 *setup* do arduino.

O *loop* do arduino verifica a cada vez que o código dentro do *loop* se repete o arduino verifica a conexão com o servidor verifica se houve algum erro de conexão e espera pela próxima mensagem do servidor para executar uma ação, o código do *loop* pode ser visto nas figuras 102 e 103.

```

void loop() {
    String resp = "";
    if (erro) {
        Serial.println("Erro na Conexao com o servidor.");
        digitalWrite(L_GREEN, LOW);
        digitalWrite(L_RED, HIGH);
        while (true);
    }
    if (!conectado) {
        Serial.println("Fim da Conexao.");
        digitalWrite(L_GREEN, LOW);
        digitalWrite(L_RED, HIGH);
        while (true);
    }
    if (tempo < millis()) Alive();
    resp = recebeMensagem(NO_DEBUG);
    resp = RespostaServidor(resp);
    Serial.print(resp);
}

```

Figura 102 *loop* do arduino (parte 1).

```
if (resp == "0\n") {  
    posiciona(&base, &avanco, &mergulho, 100, 100, 100);  
    moveServo(&garra, 50);  
}  
else if (resp == "") Serial.print(resp);  
else if (resp == "G-1\n") moveServo(&garra, 0);  
else if (resp == "G-2\n") moveServo(&garra, 50);  
else if (resp == "G-3\n") moveServo(&garra, 120);  
else if (resp == "B-1\n") moveServo(&base, 0);  
else if (resp == "B-2\n") moveServo(&base, 45);  
else if (resp == "B-3\n") moveServo(&base, 90);  
else if (resp == "B-4\n") moveServo(&base, 135);  
else if (resp == "B-5\n") moveServo(&base, 180);  
else if (resp == "A-1\n") moveServo(&avanco, 50);  
else if (resp == "A-2\n") moveServo(&avanco, 100);  
else if (resp == "A-3\n") moveServo(&avanco, 150);  
else if (resp == "A-4\n") moveServo(&avanco, 200);  
else if (resp == "M-1\n") moveServo(&mergulho, 10);  
else if (resp == "M-2\n") moveServo(&mergulho, 35);  
else if (resp == "M-3\n") moveServo(&mergulho, 55);  
else if (resp == "M-4\n") moveServo(&mergulho, 80);  
else if (resp == "M-5\n") moveServo(&mergulho, 120);  
}
```

Figura 103/loop do arduino (parte 2).

5 RESULTADOS E DISCUSSÃO

O desenvolvimento foi dividido em duas partes para agilizar e facilitar os procedimentos necessários, sendo elas o desenvolvimento do servidor e o desenvolvimento do braço robótico.

Os primeiros passos do projeto foram a instalação do Virtual box e a criação de uma nova máquina virtual para hospedar o Ubuntu. Esse processo ocorreu sem qualquer erro.

Com a máquina virtual criada, o Ubuntu foi instalado, e em seguida o Apache2, o MySQL e o PHP, todo esse procedimento também foi um sucesso.

A implementação do SSL no servidor web foi iniciada em seguida, porém alguns problemas ocorreram no acesso do servidor devido a mudança da porta 80 para a porta 443. Aproveitando a situação, foram realizados testes de scripts e configurações do apache, do MySQL e do PHP, para buscar uma melhor performance do servidor.

Feitos esses testes, com o projeto ainda no início e o cronograma em dia, toda a instalação do LAMP e as configurações foram refeitas de forma definitiva, para evitar problemas futuros com tantas alterações que já haviam sido feitas.

Com todas as configurações realizadas e o servidor web funcionando, a programação das páginas em html, e PHP, e a integração do PHP com o MySQL para o funcionamento do sistema de usuários foram preparados e incluídos no servidor corretamente.

A primeira parte do trabalho foi finalizada rapidamente com poucos problemas e a segunda parte foi iniciada com a chegada do braço robótico.

Inicialmente ocorreram problemas com o funcionamento do equipamento devido ao cabo serial com mau contato, mas que foram logo trocados e o braço funcionou de acordo com o esperado.

Problemas de logística vieram em seguida, com a compra de um modulo errado, não suficiente para nosso projeto.

Quando foi iniciada a programação e configuração do modulo Wi-Fi, controlado pelo arduino, ocorreram alguns problemas. O modulo Wi-Fi não apresentava as suas configurações e nem se conectava a rede. A versão do firmware foi atualizada, porém não ouve nenhum resultado positivo, mantendo a

versão 1.1. Após uma semana verificando as possibilidades de erros, como erros no circuito, modulo danificado, tensão insuficiente, foi verificado que o erro estava no código de comunicação entre modulo e arduino. A correção foi feita, e a configuração entre o modulo e o arduino apresentou resultado positivo.

Após o modulo se conectar a redes Wi-Fi, foi iniciado a programação para a comunicação entre arduino e servidor, o servidor e arduino não conseguiam se conectar de princípio e o teste “ping” não conseguia respostas. Dentro de 2 semanas foram feitos ajustes no servidor e correções no código, a conexão entre modulo e servidor foi estabelecida mas o teste ping ainda continuava sem respostas.

Foi dada continuidade a programação visando a troca de mensagens e comandos entre modulo e servidor, de início as mensagens não eram entregues para ambos os lados, tanto entre modulo para servidor, como do servidor para o modulo. Após correções no código e alguns ajustes no servidor, as mensagens do modulo através do monitor serial eram entregues para o servidor, mas as mensagens do servidor não eram recebidas pelo modulo. O problema foi detectado na função que recebia a mensagem, com o erro reparado a comunicação entre ambos os lados passou a funcionar.

A Interface do servidor foi criada com facilidade, e foram adotados alguns comandos pré-definidos, inseridos em botões para a movimentação do braço. Com isso o braço já estava operacional e a comunicação, apesar de um pouco lenta, funcionando corretamente.

Mesmo operacional, o servidor se encontrava instável e o braço travando eventualmente, mas com pequenas alterações no código como o *delay*, o problema foi corrigido. Para melhorar a resposta do braço foi inserida uma fonte de 5 volts no arduino.

Com a sobra de tempo após a conclusão do projeto foi feita uma melhoria na interface, deixando mais agradável e fácil para o usuário. As Figuras 104 e 105 mostram a evolução da interface.



Figura 104 Primeira Interface para controle do braço robótico.

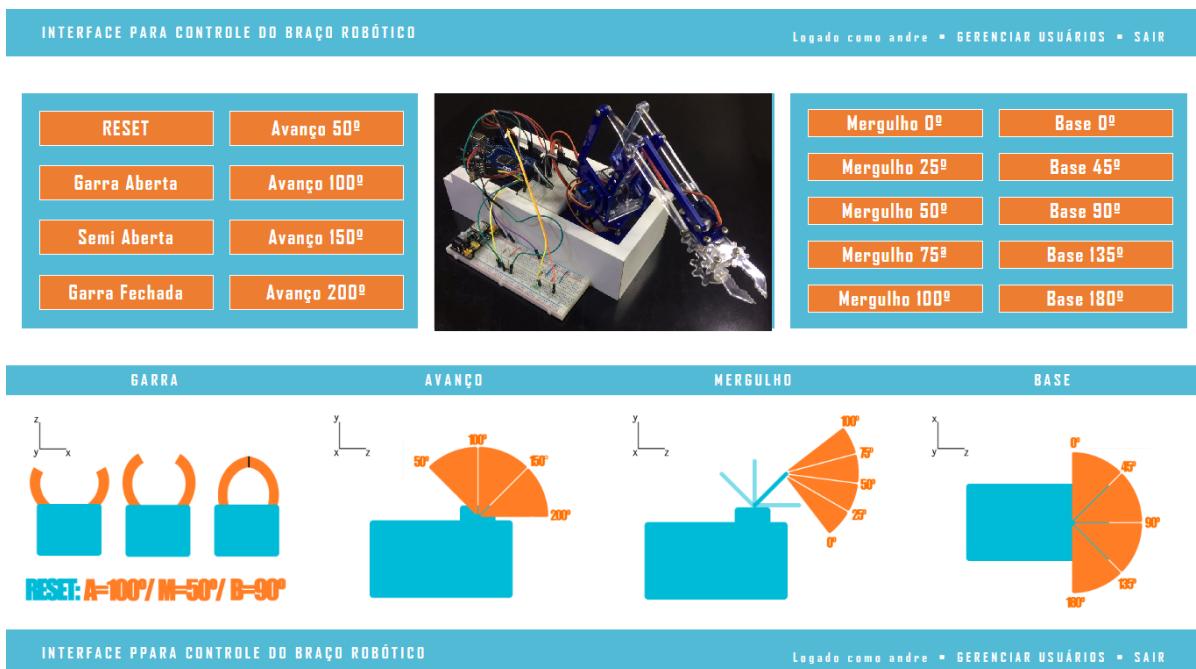


Figura 105 Interface Finalizada.

6 CONCLUSÃO

Neste trabalho foi abordado o assunto de aprimoramento em uma interface para controlar remotamente o protótipo de um braço robótico já existente, implementar em um servidor web para o acesso de qualquer lugar, e trabalhar em um sistema de segurança para o equipamento, e foi concluído, com base nos resultados da pesquisa de campo e nos resultados obtidos, que esse trabalho é essencial e viável nos dias de hoje, também pelo seu custo de implementação é muito acessível se comparado aos benefícios e segurança que esse projeto propõe no dia-a-dia de uma empresa e empregado.

Os resultados obtidos com a realização deste projeto foram excepcionais, como apresenta a Figura 106, já que, mesmo com dificuldades em relação com a comunicação entre servidor, módulo e arduino, foram concluídas e definidas operacionais, abrindo novas sugestões de melhorias e integrações de outros serviços.

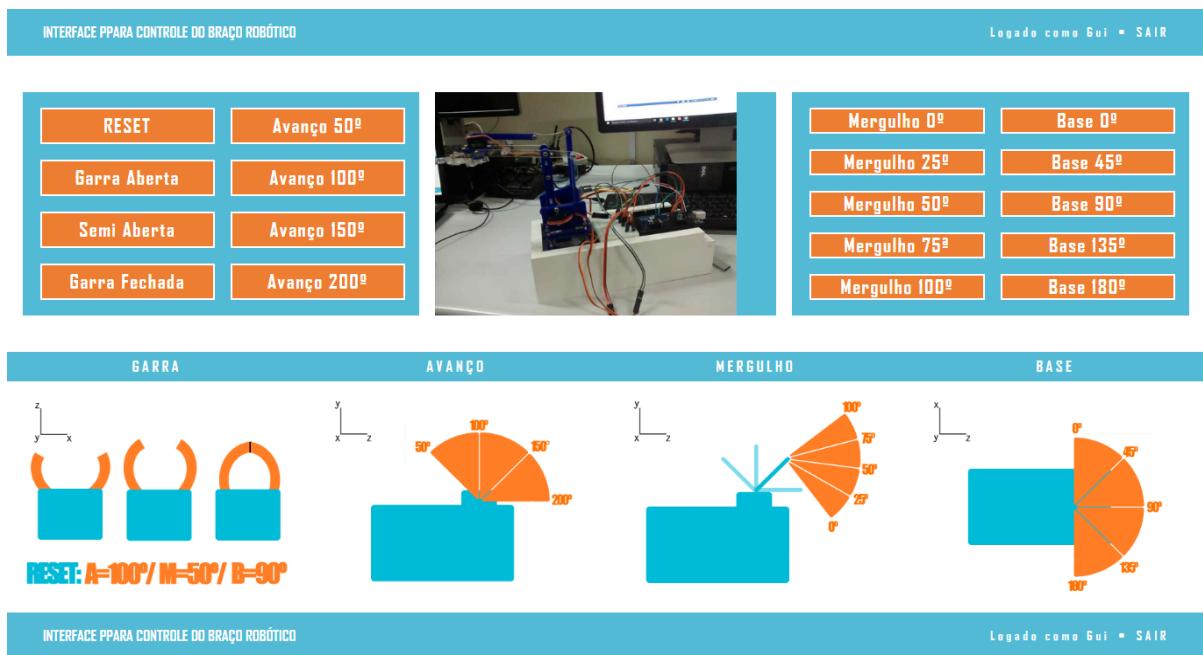


Figura 106Resultados do projeto.

Foi observado também que a implementação da câmera IP foi essencial em nosso projeto, para que de fato o usuário possa usar o equipamento de qualquer lugar, e obter uma resposta visual do que foi feito.

O desenvolvimento das industrias e de suas linhas de produção, trazem a necessidade de haver cada vez mais segurança e tempo disponível de serviços prestados com estabilidade de operação. O baixo custo de manutenção e

esforços humanos quase inexistentes devido a sua interface de controle, fazem desse projeto algo de interesse em diversas áreas e ramos de trabalho.

Em suma, o objetivo específico proposto foi alcançado, e em decorrência do mercado de trabalho e o projeto foi considerado viável, satisfatório e essencial para uma melhor segurança em atividades de risco ou que envolvam esforço físico, inibindo aspectos preocupantes para as empresas e inseguranças do operador e empregado, além de trazer um aumento no desenvolvimento da sociedade, considerando que não haverá aumento no número de desempregados, as empresas terão mais lucros, e o governo poderá investir mais em educação e hospitais.

7 RECOMENDAÇÕES

Com o termo do projeto, as implementações de outros trabalhos podem vir a ser interessantes e viáveis. A implementação de um controle Joystick arduino para a movimentação do braço robótico pode ser uma proposta simples e eficaz para facilitar o manuseio. Com sensores de movimento e com ferramentas de monitoramento como o Zabbix, níveis de óleo, temperatura de operação, erros no circuito, problemas nos servos, entre outros podem ser visualizados através desses sensores e capitados pela para facilitar o uso do usuário. Um serviço de log pode ser implementado para razões como horário de início de movimentação, termo de atividade, tempo de utilização, comandos disparados e imagens da câmera. Ferramenta de monitoramento. Também podem ser realizadas melhorias na interface, adaptando para aparelhos celulares, aumentando a precisão da movimentação do braço, ou podendo simplesmente também deixar mais agradável e fácil ao usuário.

REFERÊNCIAS

ARRUDA, Felipe. **A história da interface gráfica.** 08 de abr. 2011 Disponível em: <<https://www.tecmundo.com.br/historia/9528-a-historia-da-interface-grafica.htm>>. Acesso em: 22 de nov. 2016.

ASCENCIO, Ana Fernanda Gomes; CAMPOS, Edilene Aparecida Veneruchi de. **Fundamentos da programação de computadores:** Algoritmos, Pascal, C/C++ e Java. 2. ed. São Paulo: Pearson Prentice Hall, 2007.

AULDS, Charles. **Linux Apache Web Server Administration.** Alameda: Sybex Inc, 2000. (Craig Hunt Linux Library).

BARRETO, Carlos Alberto Alves. **História dos motores elétricos.** 13 de set. 2009. Disponível em: <<http://www.ebah.com.br/content/ABAAAATzQAF/historia-dos-motores-eletricos>>. Acesso em 01 de dez. 2016.

Canonical. **The Ubuntu story.** Ubuntu. 2016. Disponível em: <<https://www.ubuntu.com/about/about-ubuntu>>. Acesso em: 20 de nov. 2016.

Canonical. **Ubuntu Flavors:** Introduction to flavors. Wiki Ubuntu. Nov. 2016. Disponível em: <https://wiki.ubuntu.com/UbuntuFlavors?_ga=1.187045288.147214052.1473270466>. Acesso em: 20 de nov. 2016.

CASTRO, Maria Alice Soares de. **Apostila de Introdução à Linguagem HTML:** Disponibilização de Conteúdos na WEB. Mar. 2007. Disponível em: <https://fit.faccat.br/~fpereira/pagina/informatica/apostilas/apostila_html_mar2007.pdf>. Acesso em: 22 de nov. 2016

CENTRAL SERVER. **Certificado ssl.** 5 de jun. 2012 Disponível em: <<http://www.centralserver.com.br/servicos/ferramentas/certificado-ssl/>>. Acesso em: 04 out. 2016.

CERTBR. **Cartilha de segurança.** 3 de jun. 2012 Disponível em: <<http://cartilha.cert.br/redes/>>. Acesso em: 20 set. 2016.

CISCO. **O que é um firewall?** 2 de abr. 2016 Disponível em: <http://www.cisco.com/c/pt_br/products/security/firewalls/what-is-a-firewall.html>. Acesso em: 14 out. 2016.

CLÁUDIO, Arilo. **Bancos de Dados Relacionais – Artigo Revista SQL Magazine 86.** 5 de abr. 2011. Disponível em: <<http://www.devmedia.com.br/bancos-de-dados-relacionais-artigo-revista-sql-magazine-86/20401>>. Acesso em 22 de nov. 2016.

CONTI, Fátima. **Interfaces.** 6 de mar. 2011. Disponível em: <<http://www.ufpa.br/dicas/linux/li-lint.htm>>. Acesso em: 22 de nov. 2016.

DATE, C.j. **Introdução a Sistemas de Banco de Dados.** 7. ed. Rio de Janeiro: Campus Ltda: 1999. Tradução da 7. ed. Americana.

Debian Documentation Team. **A Brief History of Debian:** Chapter 1 - Introduction -- What is the Debian Project?. Debian. Abr. 2015. Disponível em: <<https://www.debian.org/doc/manuals/project-history/ch-intro.en.html>>. Acesso em: 20 de nov. 2016.

Debian Documentation Team. **A Brief History of Debian:** Chapter 4 - A Detailed History. Debian. Abr. 2015. Disponível em: <<https://www.debian.org/doc/manuals/project-history/ch-detailed.en.html>>. Acesso em: 20 de nov. 2016.

DUMAS, Véronique. **A origem da internet.** 8 de jan. 2010. Disponível em: <http://www2.uol.com.br/historiaviva/reportagens/o_nascimento_da_internet.html>. Acesso em 22 de nov. 2016

FERREIRA, Rubem E. **Linux Guia do Administrador do Sistema:** Linux. 2. ed. São Paulo: Novatec, 2010.

Free Software Foundation. Dica de Leitura: <<https://www.gnu.org/philosophy/free-sw.en.html>>. Acesso em: 20 de nov. 2016.

HARRIS, Tom. **O Braço robótico.** 17 out. de 2015. Disponível em: <<http://tecnologia.hsw.uol.com.br/robos2.htm>>. Acesso em: 22 de nov. 2016.

HIGA, Paulo. **As 10 principais diferenças entre o Windows e o Linux.** 4 de mai. 2011. Disponível em: <<http://www.guiadopc.com.br/artigos/3394/as-10-principais-diferencias-entre-o-windows-e-o-linux.html>>. Acesso em 22 de nov. 2016.

JMV Technology. **Câmera IP – Tudo Sobre Câmera IP.** 11 de dez. 2010. Disponível em: <<http://www.sitehosting.com.br/camera-ip/>>. Acesso em 22 de nov. 2016.

KERNIGAN, Brian W; RITCHIE, Dennis M. **The C programming language.** 2. ed. New Jersey: Prentice Hall Ptr, 1988.

LIU, Cricket; ALBITZ, Paul. **DNS and BIND.** 3. ed. Sebastopol: O'reilly Media, 1998.

LJUBUNCIC, Igor. **Apache Web server – Complete Guide,** 2011. Disponível em: <<https://sites.duke.edu/workblog/files/2014/12/www.dedoimedo.com-apache-web-server-lm.pdf>>. Acesso em: 20 de nov. 2016.

LONGMAN, Addison Wesley. **A history of HTML.** 1998. Disponível em: <<http://www.w3.org/People/Raggett/book4/ch02.html>>. Acesso em 22 de nov. 2016.

MCKIE, Stewart. **Everything You Ever Wanted To Know About Client/Server Computing But Were Afraid To Ask,** 1995. Disponível em: <<http://businessfinancemag.com/technology/everything-you-ever-wanted-know-about-clientserver-computing-were-afraid-ask/>>. Acesso em: 20 de nov. 2016.

MCROBERTS, Michael, **Arduino basico** - Novatec Editora Ltda.2011.

MOTA FILHO, João Eriberto. **Descobrindo o Linux:** Entenda o sistema operacional GNU/Linux. 3. ed. São Paulo: Novatec, 2012.

MOURA, Edi Carlos. **Servidor Web Apache.** 28 de mar. 2011. Disponível em: <<http://softwarelivre.org/edicarlos/edi-carlos-moura/servidor-web-apache>>. Acesso em: 01 de dez. 2016.

MOURA, Isabella Mayer de. **A Pollux Automation mais que dobrou o faturamento ao apostar em soluções diferenciadas.** 17 de out. 2015. Disponível em: <<http://ndonline.com.br/joinville/noticias/empresa-de-joinville-investe-em-tecnologia-e-desenvolve-robos-colaborativos-para-industrias>>. Acesso em 22 de nov. 2016.

Mozilla Developer Network. **O que é CSS.** 15 de fev. 2016. Dica de leitura: <https://developer.mozilla.org/pt-PT/docs/Web/CSS/Como_com%C3%A7ar/O_que_%C3%A9_CSS>. Acesso em: 22 de nov. 2016.

NEVES, Pedro M. C. **O dia prático da HTML.** 1.ed. Lisboa: Inova, 2004.

OGATA, Katsuhiko. **Engenharia de Controle Moderno**. Tradução de Prof. Bernardo Severo. LTC Editora: Rio de Janeiro, 1998

PEREIRA, Adriana Soares; VISSOTO, Elisa Maria; FRANCISCATTO, Roberto. **Sistemas Operacionais**. 1. Ed. Frederico Westphalen. E-Tec Brasil, 2015.

PEREIRA, Ana Paula. **O que é CSS?**. 09 de set. 2009. Disponível em:
<http://www.tecmundo.com.br/programacao/2705-o-que-e-css-.htm>. Acesso em: 22 de nov. 2016.

PIXININE, Juliana. **Como um wi-fi funciona? entenda a tecnologia**. 21 de fev. 2015 Disponível em:
<http://www.techtudo.com.br/noticias/noticia/2015/02/como-um-wi-fi-funciona-entenda-tecnologia.html>. Acesso em: 10 set. 2016.

SCHILD, Herbert. **C**: Completo e total. 3. ed. São Paulo: Makron Books, 1996.

SOARES, Luiz de Jesus Peres. **Os impactos financeiros dos acidentes do trabalho no orçamento brasileiro**: uma alternativa política e pedagógica para redução dos gastos. Brasília : s.ed.2008.56f.

Software in Public Interest. **About Debian**: WHAT is Debian? Debian. Jul. 2016 Disponível em:
<https://www.debian.org/intro/about.en.html>. Acesso em: 20 de nov. 2016.

SOUZA, Airton Ribeiro de. **Evolução histórica das redes de computadores**. 23 de nov. 2008. Disponível em:
http://www.lanwan.com.br/Aulas_Senac/Tecnico_Redes_Noturno/Aula%2006052010%20-%20Historia%20das%20redes.pdf. Acesso em 22 de nov. 2016.

SOUZA, Airton Ribeiro de. **HISTÓRICO DAS REDES DE COMPUTADORES**. 23 de nov. 2008. Disponível em:
http://www.lanwan.com.br/Aulas_Senac/Tec_Redes_Final_Semana/Aula%20270310%20e%2010042010%20-%20Historia%20das%20redes.pdf. Acesso em 22 de nov. 2016.

The PHP Group. **História do PHP**. Disponível em: http://php.net/manual/pt_BR/history.php.php. Acesso em 31 de ago. 2016.

Tribunal Superior do Trabalho. **Dados dos acidentes do trabalho de 2011**. 14 de dez. 2015. Disponível em: <http://www.tst.jus.br/web/trabalhoseguro/dados-nacionais>. Acesso em: 22 de nov. 2016.

APÊNDICE A**PESQUISA DE CAMPO**

1. Você trabalha?

Sim.

Não.

Não sei responder.

2. Seu trabalho envolve esforço físico?

Sim.

Não.

Não sei responder.

3. Seu emprego oferece algum risco a segurança?

Sim.

Não.

Não sei responder.

4. Seu trabalho envolve linha de produção?

Sim.

Não.

Não sei responder.

5. Você é chefe de uma empresa ou microempresa?

Sim.

Não.

Não sei responder.

6. Você tem interesse e desenvolver e facilitar o serviço em sua empresa, aumentando a produção os lucros e a segurança aos funcionários?

Sim.

Não.

Não sei responder.

7. Você compraria um braço robótico que pode ser controlado de qualquer lugar, oferecendo segurança em trabalhos de risco e facilidade ao funcionário?

Sim.

Não.

Não sei responder.

8. Você acredita no sucesso desse projeto?

Sim.

Não.

Não sei responder.
