



Fundação Educacional do Município de Assis  
Instituto Municipal de Ensino Superior de Assis  
Campus "José Santilli Sobrinho"

**PEDRO VICENTE PRATA FAZANO JUNIOR**

**PROJETO DOMÓTICO PARA AMBIENTES INTELIGENTES  
BASEADO NAS TECNOLOGIAS ARDUINO E GOOGLE ANDROID**

Assis  
2013

**PEDRO VICENTE PRATA FAZANO JUNIOR**

**PROJETO DOMÓTICO PARA AMBIENTES INTELIGENTES  
BASEADO NAS TECNOLOGIAS ARDUINO E GOOGLE ANDROID**

Trabalho de Conclusão de Curso apresentado ao  
Instituto Municipal de Ensino Superior de Assis,  
como requisito do Curso de Graduação

**Orientador:** Prof. Esp. Guilherme de Cleve Farto

**Área de Concentração:** Informática

Assis  
2013

## **FICHA CATALOGRÁFICA**

FAZANO JR., Pedro Vicente Prata

Projeto Domótico Para Ambientes Inteligentes Baseado Nas Tecnologias  
Arduino E Google Android / Pedro Vicente Prata Fazano Junior. Fundação  
Educativa do Município de Assis – FEMA – Assis, 2013.  
93p.

Orientador: Prof. Esp. Guilherme de Cleve Farto

Trabalho de Conclusão de Curso – Instituto Municipal de Ensino Superior de  
Assis – IMESA

**1.** Domótica **2.** Arduino **3.** Google Android

CDD: 001.6  
Biblioteca da FEMA

# **PROJETO DOMÓTICO PARA AMBIENTES INTELIGENTES BASEADO NAS TECNOLOGIAS ARDUINO E GOOGLE ANDROID**

**PEDRO VICENTE PRATA FAZANO JUNIOR**

Trabalho de Conclusão de Curso apresentado ao  
Instituto Municipal de Ensino Superior de Assis,  
como requisito do Curso de Graduação, analisado  
pela seguinte comissão examinadora:

**Orientador:** Prof. Esp. Guilherme de Cleve Farto

**Analisador (1):** Prof. Dr. Alex Sandro Romeo de Souza Poletto

Assis  
2013

## DEDICATÓRIA

Dedico este trabalho à minha família, amigos e  
todas as pessoas que acreditaram em meus  
sonhos e anseios, apoiando-me com a força  
necessária para que pudesse realizá-los

## **AGRADECIMENTOS**

A todos os professores (Leonor, Marisa, Diomara, Almir, Regina, Guilherme, Fábio, Cleiton, Douglas, Fernando, Alex, Begosso, Talo e Domingos) que estiveram comigo nessa caminhada e que contribuíram de alguma maneira para o meu crescimento pessoal e profissional.

À minha noiva Barbara e a minha família, por estar sempre ao meu lado me apoiando e fortalecendo.

Aos amigos que cultivei durante o curso que me ajudaram nessa caminhada.

Ao meu orientador professor Guilherme de Cleva Farto, pelo apoio nas pesquisas e pelo conhecimento transmitido para que esse trabalho pudesse ser realizado.

Por fim agradeço a todos que contribuíram diretamente ou indiretamente para a minha formação e na execução deste trabalho, deixo aqui meu muito obrigado a todos.

*“Ele não sabia que era impossível. Foi lá e fez.”*

Jean Cocteau (1889 -1963)

## RESUMO

Atualmente o setor domótico está crescendo em todo o mundo e o Brasil não está de fora desse crescimento. Nos últimos quatro anos, o setor domótico no Brasil cresceu aproximadamente 300% de acordo com dados da Aureside (Associação Brasileira de Automação Residencial) e cerca de 250 empresas atualmente estão no ramo da domótica. Devido à alta procura nesse setor e surgindo muitas oportunidades no ramo, a proposta deste trabalho foi a de mostrar o desenvolvimento de um protótipo miniaturizado de um projeto domótico de baixo custo, alto poder computacional, fácil interatividade e gerenciado por dispositivos móveis como *smartphones* e *tablets* por meio de tecnologias *open-source*. Para tal desenvolvimento foi necessário pesquisar e compreender os conceitos da domótica fazendo-se uso da plataforma *Arduino* para a criação do protótipo miniaturizado e para o gerenciamento dos sensores e atuadores, assim como o desenvolvimento de um aplicativo baseado na plataforma *Google Android* chamado *ArduDomus* capaz de controlar e gerenciar por meio do WiFi, criando uma comunicação entre o morador e usuário com sua residência.

**Palavras-chave:** *Domótica; Arduino; Google Android.*



## **ABSTRACT**

Currently the sector home automation is growing around the world and Brazil is not out of this growth. Over the past four years, the home automation industry in Brazil grew by approximately 300 % according to data from Aureside (Brazilian Association of Residential Automation) and about 250 companies are currently in the field of home automation. Due to high demand in this sector and many emerging opportunities in the business, the purpose of this study was to show the development of a prototype miniaturized of a design home automation low cost, high computational power, easy interactivity and managed by mobile devices such as smartphones and tablets through open-source technologies. For such development is necessary to search out and understand the concepts of home automation by making use of the Arduino platform for creation of prototype miniaturized and for managing sensors and actuators, as well as the development of an application based on the Google Android platform called ArduDomus able to control and manage through WiFi, creating a communication between the dweller and user with your residence.

**Keywords:** *Home Automation; Arduino; Google Android.*

## LISTA DE ILUSTRAÇÕES

Figura 1 – Aplicações da Domótica (Adaptado de CEDOM, 2008) .....	23
Figura 2 – Sensores (CASTILLO, 2009) .....	30
Figura 3 – Atuadores (CASTILLO, 2009) .....	31
Figura 4 – Arquitetura Centralizada (Adaptado de HUIDOBRO et al., 2007) .....	32
Figura 5 – Arquitetura Descentralizada (Adaptado de HUIDOBRO et al., 2007) .....	33
Figura 6 – Arquitetura Distribuída (Adaptado de HUIDOBRO et al., 2007) .....	34
Figura 7 – Rede Doméstica (CASTILLO, 2009) .....	35
Figura 8 – Equipe Arduino (Cuartielles, 2008).....	43
Figura 9 – Hardware Arduino .....	45
Figura 10 – Arduino Mega (Arduino, 2013) .....	47
Figura 11 – Arduino Nano (ARDUINO, 2013).....	48
Figura 12 – Arduino LilyPad (ARDUINO, 2013) .....	49
Figura 13 – Program-ME (GLOBAL CODE, 2013) .....	49
Figura 14 – IDE Arduino .....	50
Figura 15 – Botões IDE Arduino.....	51
Figura 16 – Sketch, Barra de status e Área de notificação do Arduino .....	52
Figura 17 – Empilhamento de Shields.....	54
Figura 18 – Motor Shield, SD Shield, Ethernet Shield e Wave Shield.....	55
Figura 19 – Emulador Android.....	61
Figura 20 – Camadas da Arquitetura Android (ANDROID DEVELOPERS, 2013) ....	63
Figura 21 – Arquitetura simplificada das tecnologias utilizadas .....	70
Figura 22 – Arquitetura Sistema Domótico.....	71
Figura 23 – Planta baixa da maquete.....	72
Figura 24 – Esboço das medidas para o corte da madeira .....	73

Figura 25 – Maquete do Protótipo Domótico .....	74
Figura 26 – LDR .....	75
Figura 27 – Sensor DHT11.....	76
Figura 28 – Sensor Reed Switch (sensor magnético) .....	76
Figura 29 – Servomotor.....	77
Figura 30 – Sensor Umidade Solo .....	78
Figura 31 – Mini bomba de água.....	79
Figura 32 – Ligação paralelo relé e interruptor.....	79
Figura 33 – Shield Ethernet com roteador WiFi.....	80
Figura 34 – Interface para o usuário (página html).....	81
Figura 35 – ArduDomus .....	83
Figura 36 – Configuração do ArduDomus .....	84
Figura 37 – Sincronização do ArduDomus.....	85

## **LISTA DE TABELAS**

Tabela 1 - Meios de Transmissão .....	37
Tabela 2 - Padrões da Domótica (BOLZANI, 2004), (ARAUJO; PEREIRA, 2004) e (MACEDO, 2009) .....	39
Tabela 3 - Resumo da placa Arduino (ARDUINO, 2013) .....	44
Tabela 4 - Versões do Android (Android Developers, 2013) .....	59

## SUMÁRIO

<b>1 – INTRODUÇÃO .....</b>	<b>16</b>
1.1 - OBJETIVOS .....	18
<b>1.1.1 Objetivos específicos.....</b>	<b>19</b>
1.2 – JUSTIFICATIVAS .....	19
1.3 – MOTIVAÇÃO .....	20
1.4 - ESTRUTURA DO TRABALHO.....	20
<b>2 – DOMÓTICA.....</b>	<b>21</b>
2.1 – HISTÓRIA PRIMORDIOS.....	21
2.2 – APLICAÇÕES DA DOMÓTICA .....	22
2.3 – BENEFÍCIOS DA DOMÓTICA.....	26
<b>2.3.1 - Gestão de energia .....</b>	<b>27</b>
<b>2.3.2 - Conforto .....</b>	<b>27</b>
<b>2.3.3 - Segurança.....</b>	<b>27</b>
2.3.3.1 - Segurança de bens.....	27
2.3.3.2 - Segurança pessoal .....	28
2.3.3.3 - Incidentes e falhas .....	28
<b>2.3.4 - Comunicações.....</b>	<b>28</b>
2.4 – EQUIPAMENTOS UTILIZADOS .....	29
<b>2.4.1 – Controlador.....</b>	<b>29</b>
<b>2.4.2 – Sensor .....</b>	<b>29</b>
<b>2.4.3 – Atuador.....</b>	<b>30</b>
<b>2.4.4 – Interface .....</b>	<b>31</b>
2.5 – ARQUITETURA DA DOMÓTICA .....	31
<b>2.5.1 – Centralizada .....</b>	<b>32</b>
<b>2.5.2 – Descentralizada .....</b>	<b>33</b>
<b>2.5.3 – Distribuída (híbrida) .....</b>	<b>34</b>
2.6 – REDE DOMÉSTICA .....	35
2.7 – MEIOS DE TRANSMISSÃO .....	36

2.8 – ACESSIBILIDADE .....	38
2.9 – PROTOCOLOS E PADRÕES EXISTENTES.....	38
2.10 – DOMÓTICA NO BRASIL .....	40
<b>3 - COMPUTAÇÃO FÍSICA COM ARDUINO .....</b>	<b>42</b>
3.1 – HISTÓRIA DO ARDUINO .....	43
3.2 – HARDWARE ARDUINO .....	44
<b>3.2.1 - Entrada e Saída .....</b>	<b>46</b>
<b>3.2.2 – Comunicação.....</b>	<b>46</b>
<b>3.2.3 – Modelos De Arduino.....</b>	<b>46</b>
3.2.3.1 – Arduino Mega.....	47
3.2.3.1 – Arduino Nano .....	48
3.2.3.1 – Arduino LilyPad.....	48
3.2.3.1 – Program-ME.....	49
3.3 – IDE ARDUINO .....	50
3.4 - LINGUAGEM ARDUINO .....	52
<b>3.4.1 – Funções no Arduino.....</b>	<b>52</b>
<b>3.4.2 – Bibliotecas .....</b>	<b>54</b>
3.5 - SHIELDS .....	54
3.6 - HARDWARE E SOFTWARE OPEN-SOURCE.....	56
3.7 – COMUNIDADE ARDUINO.....	57
<b>4 - TECNOLOGIA GOOGLE ANDROID .....</b>	<b>58</b>
4.1 - OPEN HANDSET ALLIANCE .....	60
4.2 - DESENVOLVIMENTO DE APLICATIVOS ANDROID .....	61
4.3 - Máquina Virtual Dalvik.....	62
4.4 - ARQUITETURA ANDROID .....	63
4.4.1 – Linux Kernel.....	64
<b>4.4.2 – Libraries e Android Runtime .....</b>	<b>64</b>
<b>4.4.3 – Application Framework.....</b>	<b>65</b>
<b>4.4.4 – Applications.....</b>	<b>65</b>
4.5 - COMPONENTES BÁSICOS DE UM APLICATIVO ANDROID.....	65

4.5.1 – Activities .....	66
4.5.2 – Intents.....	66
4.5.3 - Broadcast Receivers.....	67
4.5.4 – Service.....	67
4.5.5 - Content Provider .....	68
<b>5 – PROPOSTA DE TRABALHO.....</b>	<b>69</b>
5.1 – DEFINIÇÃO DA PROPOSTA DO TRABALHO .....	69
5.2 – ARQUITETURA DO PROJETO DOMÓTICO.....	70
5.3 – PLANTA DA MAQUETE .....	71
<b>6 - ESTUDO DE CASO .....</b>	<b>73</b>
6.1 – CONSTRUÇÃO DA MAQUETE .....	73
6.2 – PARTE FÍSICA DO PROJETO.....	74
6.3 – PARTE MÓVEL (INTERFACE) DO PROJETO.....	81
6.3.1 - ArduDomus.....	82
<b>7 – CONCLUSÃO .....</b>	<b>88</b>
7.1 – TRABALHOS FUTUROS.....	88
<b>REFERÊNCIAS.....</b>	<b>90</b>

## 1 – INTRODUÇÃO

Historicamente, o homem vem procurando “controlar e aperfeiçoar” o tempo desde a primeira utilização da roda entre 3500 e 3200 a.C. e, com isso, aprendera que, simplificando e substituindo o trabalho braçal por outros meios e mecanismos, era capaz de aproveitar o tempo ganho com demais afazeres e, dessa forma, criar novos sistemas de automatização, chegando a grandes descobertas que marcaram época. Como, por exemplo, na Inglaterra em 1086, período em que os moinhos de água produziam energia de forma revolucionária e barata, no século VII quando o arado foi uma das principais invenções para a agricultura e a descoberta da agulha magnética que possibilitou a navegação para novos continentes. Já na era das indústrias automatizadas, o homem criou máquinas que passaram a executar os trabalhos repetitivos com o objetivo de elevar a taxa da produtividade e qualidade nesse setor (SILVEIRA; SANTOS, 2009).

Recentemente, com a evolução da computação e com a miniaturização dos componentes eletrônicos fora possível começar a automatizar as residências e tornar os ambientes mais dinâmicos e inteligentes. Esse tipo de automatização residencial também é conhecido como Domótica e é um neologismo da palavra em latim “domus”, que significa casa ou residência, com robótica (BOLZANI, 2004).

A domótica é uma nova e emergente área de tecnologia que possibilita controlar diversos ambientes de uma residência e, segundo Chamusca (2006), deve satisfazer três necessidades básicas das pessoas: o conforto, a segurança e as comunicações. Os moradores podem interagir com a residência controlando seus sensores e atuadores por meio de dispositivos interligados ao ambiente como, por exemplo, utilizar de um *smartphone* ou *tablet* para gerenciar diversos componentes e módulos dos cômodos de sua casa e escritório tais como a iluminação, a climatização, o acionamento de persianas das janelas entre outros sistemas que podem ser automatizados.

Essa moderna tecnologia possui a capacidade de melhorar, e muito, a qualidade de vida de pessoas portadoras de deficiência, tornando possível o controle de



equipamentos de suas residências mesmo quando não é possível se locomover até à área desejada por estar em uma cadeira de rodas ou acamado, assim como também ocorre comumente com pessoas idosas (BOLZANI, 2004).

Com a popularização dos dispositivos portáteis como *smartphones* e *tablets* e o crescente uso e popularização desses aparelhos, ainda tem-se feito pouco uso de tecnologias móveis para permitir a interação com diversos outros equipamentos por meio da Internet, redes *WiFi*, componentes e módulos *Bluetooth*, viabilizando a concepção, modelagem e desenvolvimento de ambientes inteligentes que fazem uso de automação residencial com centrais de gerenciamento móveis disponíveis em *smartphones* e *tablets*.

A escolha das plataformas para desenvolvimento ocorrera levando-se em consideração diversos detalhes entre os quais se destacam a utilização de projetos *open-source* como forma de promover *hardware* e *software* livres assim como a abordagem e uso de conceitos e tecnologias emergentes e em alta nas áreas acadêmica, científica e profissional tais como computação física, ambientes inteligentes, *Arduino*, Java e *Google Android*.

Neste trabalho fora utilizada a plataforma *Arduino*, projeto este que teve início em 2005, na Itália, e que é amparado pelos conceitos de desenvolvimento *open-source*, podendo ser utilizado para a criação de protótipos eletrônicos, e baseia-se em *hardware* e *software* flexíveis e de fácil utilização. Inicialmente, o objetivo do projeto *Arduino* era apenas o de facilitar o ensino de eletrônica e programação para pessoas que não fossem familiarizadas com a área de engenharia elétrica assim como promover o acesso a placas, componentes e dispositivos mais baratos, pois as plataformas existentes possuíam um valor de custo muito alto (BANZI, 2012).

A partir da concepção e construção das primeiras placas *Arduino*, tornou-se possível que um grande número de pessoas interessadas pudesse iniciar a criação de projetos eletrônicos e objetos interativos de forma fácil, rápida e com um valor de investimento menor (MCROBERTS, 2011) e (BANZI, 2012).

Far-se-á uso também do sistema operacional e plataforma de desenvolvimento *open-source Google Android*. O *Android* é uma plataforma para implementação de aplicações móveis de código-aberto criadas inicialmente pela Google, porém

atualmente mantida pela *Open Handset Alliance* (OHA), grupo este formado por mais de trinta empresas unidas com o objetivo de inovar e acelerar o desenvolvimento de aplicações e serviços para dispositivos móveis, principalmente no contexto de tecnologia da informação (TI) e telefonia.

O *Android* foi desenvolvido baseado no *kernel Linux* e suas aplicações são desenvolvidas na linguagem *Java* junto com bibliotecas específica para ele, que rodam em uma máquina virtual própria desenvolvida para dispositivos moveis chamada *Dalvik*, que permite que as aplicações possam rodar em qualquer dispositivo *Android* independentemente de sua arquitetura. (DIMARZIO, 2008), (FARIA, 2008), (ABLESON; COLLINS; SEN, 2009) e (LECHETA, 2009).

Segundo estatísticas da *Strategy Analytics*, em 2012, dos 700 milhões de *smartphones* vendidos, cerca de 68,4% eram dispositivos com *Google Android*, 19,4% com *Apple iOS* e os outros 12,2% divididos entre outros sistemas operacionais disponíveis no mercado (MARTINS, 2013).

## 1.1 - OBJETIVOS

O objetivo geral deste trabalho é pesquisar e compreender os conceitos da domótica fazendo-se uso da plataforma *Arduino* para o gerenciamento dos sensores e atuadores e da tecnologia *Google Android* para a comunicação com o ambiente residencial inteligente, com a finalidade de modelar e construir um protótipo miniaturizado de um projeto domótico de baixo custo, alto poder computacional, fácil interatividade e gerenciado por dispositivos móveis como *smartphones* e *tablets*.

A partir da elaboração deste trabalho, fora possível alcançar conhecimentos necessários para conceber, modelar e desenvolver um projeto domótico fazendo-se uso de *Arduino* e *Google Android*.

### 1.1.1 Objetivos específicos

Os objetivos específicos deste trabalho são mostrados a seguir:

- Pesquisar e compreender os conceitos da Domótica;
- Pesquisar e analisar a tecnologia Arduino;
- Pesquisar e analisar a tecnologia *Google Android*;
- Modelar e construir um protótipo domótico miniaturizado;
- Desenvolver a arquitetura do projeto domótico;
- Desenvolver um *aplicativo para gerenciar o projeto* domótico;
- Testar, validar e descrever os resultados obtidos.

## 1.2 – JUSTIFICATIVAS

Devido à grande necessidade de segurança e conforto das pessoas em suas residências, a domótica estabelece-se como uma ótima opção relacionada a esse anseio, fornecendo mecanismo de gestão inteligente de uma residência, escritório e quaisquer ambientes que podem ser automatizados e gerenciados por meio de sensores e atuadores.

Além de possibilitar segurança e conforto maiores, a domótica também visa o auxílio e a execução de tarefas na vida de muitas pessoas, inclusive e principalmente para indivíduos portadores de deficiência física e mental assim como idosos, objetivando facilitar a interação com ambientes de uma forma mais independente e cômoda, contribuindo diretamente para uma vida mais saudável e tranquila para aqueles que já não dispõem de tanta capacidade motora e mental.

### 1.3 – MOTIVAÇÃO

Atualmente, a domótica vem crescendo muito no mercado juntamente com as oportunidades nessa área, porém a maioria dos serviços existentes são de alto custo inviabilizando o acesso a essa tecnologia para pessoas de baixa renda.

O desenvolvimento deste trabalho consiste no fato de a domótica ser um tema que está em constante crescimento e que engloba diversas tecnologias, porém com poucos trabalhos científicos e técnicos abordando o desenvolvimento de projetos eletrônicos residenciais integrados a dispositivos móveis.

Outra motivação são as consideráveis chances de atuar no mercado de trabalho, baseando-se neste tema. Um estudo realizado pela *Motorola Mobility* mostra que 78% dos brasileiros entrevistados disseram ter interesse em automação residencial. (AURESIDE, 2012).

### 1.4 - ESTRUTURA DO TRABALHO

Este trabalho será estruturado nos seguintes capítulos: Capítulo 1 – Introdução, Capítulo 2 – Domótica, Capítulo 3 – Computação Física com *Arduino*, Capítulo 4 – Tecnologia *Google Android*, Capítulo 5 – Proposta de Trabalho, Capítulo 6 – Estudo de Caso, Capítulo 7 – Conclusão e por último o capítulo de Referências.

## 2 – DOMÓTICA

O termo Domótica está relacionado à ciência multidisciplinar que estuda e implementa a relação do homem e a casa, utilizando processos automatizados responsáveis pelo controle e gerenciamento de uma residência, proporcionando aos seus moradores maior conforto, comodidade e segurança.

Podem-se utilizar outras denominações para a Domótica, tais como, Automação Residencial, Automação Doméstica, Residências Inteligentes, Casa do Futuro entre outras.

Entende-se por Domótica o conjunto de sistemas capazes de automatizar uma casa, provendo serviços de segurança, gestão de energia, conforto e comunicação, e que podem estar integrados por meio de redes internas e externas de comunicação, com ou sem fio, e cujo controle tem certa ubiquidade, a partir de dentro e fora da casa. Pode ser definida como a integração da tecnologia no design inteligente de um ambiente (CIEC, 2011).

Segundo Meyer (2008) Domótica é o uso de equipamentos específicos para controlar eletrodomésticos, lâmpadas, ar condicionado, aquecedores, e até mesmo perceber a localização das pessoas dentro da residência. As soluções domóticas utilizam equipamentos microcontrolados que se comunicam através de meios de comunicação homogêneos ou heterogêneos, trocando informações e tomando decisões orientadas pelo usuário ou a ele para assegurar seu conforto, segurança e acessibilidade.

### 2.1 – HISTÓRIA PRIMORDIOS

Um marco na história das residências foi à chegada da eletricidade dentro das residências, transformando não somente o modo como eram iluminadas através de lamparinas e lampiões, para a facilidade de se iluminar um ambiente com lâmpadas elétricas, mas também abrindo portas para o desenvolvimento de novas tecnologias,

serviços e outras aplicações, alterando o modo de vida das famílias em suas residências (BOLZANI, 2010).

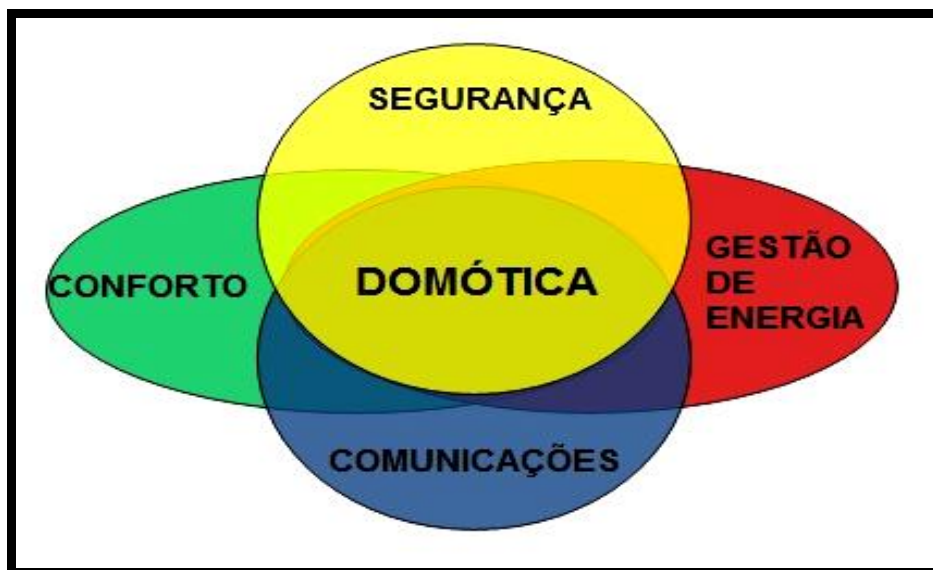
“As primeiras formas de controle residencial surgiram em meados do século XX das mãos de aficionados por eletricidade, como por exemplo, a casa de Jack Fletcher em 1954.” (STIMSON, 1954 apud BOLZANI, 2010).

Na década de 1970 começaram os primeiros esforços de algumas empresas para a produção de equipamentos domóticos. O x10 foi um dos primeiros equipamentos, desenvolvido pelo Pico Electronics de Glenrothes na Escócia em 1975, utilizava um protocolo de comunicação que trafegava através da rede elétrica e se tornou um dos padrões mais utilizados na domótica atualmente, devido ao seu preço mais baixo em relação a outros produtos.

A partir dos primeiros equipamentos desenvolvidos várias empresas criaram seus padrões e soluções para a domótica. Apesar do grande empenho de algumas empresas em padronizar os protocolos de comunicação entre dispositivos de várias marcas, isso ainda não foi alcançado. Um padrão que está sendo muito utilizado atualmente é o TCP/IP, alguns fabricante de eletrodomésticos como televisão e geladeira estão adotando esse padrão para a comunicação e troca de dados entre diversos dispositivos residenciais e acesso à internet (RILEY, 2012).

## 2.2 – APLICAÇÕES DA DOMÓTICA

As aplicações da Domótica abrangem quatro categorias principais a fim de suprir as necessidades de seus usuários, são à base de todo sistema domótico, independentemente da tecnologia utilizada, e serve de ponto de partida para realizar qualquer tipo de instalação de forma eficaz. Estes são: segurança, conforto, gestão energética e comunicações, como são representados na Figura 1 e descritos a seguir (CIEC, 2012) (CEDOM, 2008).



**Figura 1 – Aplicações da Domótica (Adaptado de CEDOM, 2008)**

Segurança: Consiste em um sistema encarregado de proteger não só os bens patrimoniais com prover segurança pessoal.

- Sistema de prevenção
  - ✓ Prevenção à intrusão: Controle de acesso por código e chave eletrônica; controle de acesso por impressão digital; controle de acesso por reconhecimento facial; vídeo porteiro; sistema de vídeo vigilância.
  - ✓ Prevenção Técnica: Detector de incêndio, manter alimentação em caso de falha; válvulas de corte de gás; válvulas de corte de água; sistema de luz de emergência; sensores de vento chuva para fechar as janelas e recolher toldos.
  - ✓ Simulação de presença: Relacionada à iluminação; motorização de aberturas e acender/desligar eletrodomésticos.
- Sistema de aviso para resolução de problemas
  - ✓ Aviso de intrusão: Alarme com conexão a central; Alarme com conexão comunitária, detector de movimento; sirene interior e exterior; detector de abertura de portas e janelas; sensores de quebra vidros.
  - ✓ Outros avisos: Botão do pânico para pedir ajuda.

- ✓ Avisos de problemas técnicos: Detector de inundação; detector de vazamento de gás, detecção de falha na energia; detector de incêndio e fumaça.

Gestão energética: Gestão de energia não é algo tangível, mas um conceito que pode ser alcançado de diversas maneiras. Em muitos casos não é necessário substituir eletrodomésticos ou outros sistemas que consomem menos, mas sim fazer uma gestão eficaz dos mesmos.

- Economia de Energia

- ✓ Economia de energia elétrica: Automatizar o desligamento do circuito elétrico de acordo com certos critérios; abertura e fechamento automático de aberturas para aproveitar condições externas de luminosidade, temperatura e outras; gerenciar tarifas energéticas (ligando dispositivos ou circuitos em horários de tarifas reduzidas); monitoramento do consumo de energia elétrica.
- ✓ Economia de combustível: Aviso de portas ou janelas abertas quando este ligado o aquecimento; regulação do aquecimento de acordo com a temperatura exterior, a hora do dia, a presença de pessoas e assim por diante; monitoramento do consumo de gás.
- ✓ Economia de água: Torneiras inteligentes; chuveiros inteligentes.

- Eficiência Energética

- ✓ Eficiência elétrica: Corretor de potência; controle de geradores.

- Geração de energia e uso de energias renováveis

- ✓ Geração elétrica: Geração elétrica com geradores; geração elétrica solar; geração elétrica eólica.
- ✓ Geração de calor: Controle do aquecimento de água com energia solar.
- ✓ Geração de água: Sistema automático de reciclagem da água.

Conforto: O conforto envolve todas as ações que podem ser realizadas para melhorar o bem estar em uma casa. Tais ações podem ser passivas, ativa ou mista.



- Controle de dispositivos
  - ✓ Controle de luminosidade: Ligar e desligar para cada ponto de luz; ligar e desligar luzes mediante sensores de luminosidade; controlar intensidade das luzes interiores e exteriores; acender ou apagar todas as luzes da casa.
  - ✓ Controle do clima: Controle da refrigeração; controle do aquecimento; controle da umidade.
  - ✓ Controle de aberturas: Controle de persianas; controle de toldos; controle de cortinas; controle de portas e janelas.
  - ✓ Controle de irrigação: Irrigação automática por horário; irrigação automática de acordo com a umidade da terra.
  - ✓ Controle do ambiente multimídia: Controle do áudio ambiente; controle do vídeo ambiente.
  - ✓ Geração e programação de cenas: Cenas de luminosidade; cenas de clima; cenas de aberturas, cenas de multimídia.
- Elementos Auxiliares
  - ✓ Comandos a distancia
  - ✓ Temporizadores
  - ✓ Controle via Internet

Comunicações: Sistemas ou infraestrutura de comunicações que a residência possui.

- Tele assistência - Ubiquidade
  - ✓ Controle a distância: Controles remotos via RF; controles remotos vi infravermelho; controle via internet; controle via linha telefônica; controle via linha celular/SMS/MMS, controle via *WiFi*
  - ✓ Monitoramento à distância: Monitoramento remoto por RF; monitoramento remoto por infravermelho; monitoramento pela internet;

monitoramento pela linha telefônica; Monitoramento pela linha celular/SMS/MMS, monitoramento via *WiFi*.

- Conectividade e Interatividade
  - ✓ Rede LAN ou *WiFi*
  - ✓ Vídeo conferência
  - ✓ Transmissão de dados: Transmissão interna entre diferentes sistemas
  - ✓ Transmissão a sistemas externos: Internet.
  - ✓ Televisão por cabo, *Pay per View*, *video on demand*.

Na realidade, existem poucas diferenças entre uma residência tradicional e outra com equipamento domótico. Trata-se da mesma residência, com equipamento semelhante, e com o mesmo desenho arquitetônico. A diferença está na incorporação de uma mínima tecnologia, que permite gerenciar de forma mais eficiente os distintos equipamentos e instalações domésticas que são instalados na residência (HUIDOBRO et al., 2007)

## 2.3 – BENEFÍCIOS DA DOMÓTICA

Segundo Bolzani (2004), os benefícios da domótica inicialmente podem ser recebidos apenas como um símbolo de status e modernidade, por ser uma novidade e causar perplexidade pelo seu alto grau tecnológico, mas assim que os moradores perceberem o conforto e conveniência que ela proporciona e que todos desejam, a domótica se torna o maior e melhor dos sonhos de consumo desejados e posteriormente se tornará uma necessidade vital a qualquer pessoa.

A domótica se concentra em sistemas integrados que incluem diferentes aplicações nos campos de: gestão de energia, conforto, segurança e comunicação. Com o objetivo principal de incrementar a qualidade de vida de seus usuários, tornando o gerenciamento da residência mais simples e de fácil acesso a qualquer morador (HUIDOBRO et al., 2007)

### **2.3.1 - Gestão de energia**

Neste campo, a automação é responsável por gerenciar o consumo de energia, usando temporizadores, interruptores de tempo, termostato e outros. Ele também aproveita os horários de tarifa reduzida para ligar aparelhos como a máquina de lavar para economizar, utilizando esses aparelhos nos horários de redução. Utiliza reguladores de intensidade para a iluminação de acordo com a incidência de luz exterior. A energia é usada somente quando e onde é necessária eliminando gastos desnecessários (CEDOM, 2008).

### **2.3.2 - Conforto**

Automação residencial oferece uma série de comodidades, através de interfaces limpas e amigáveis para os moradores que podem através de um simples comando controlar os mais diversos serviços oferecidos dentro da residência, tais como serviços de controle automático: aquecimento, água quente, resfriamento, iluminação e gestão de elementos, tais como portas, persianas, toldos, janelas, sistema de irrigação e outros. (CEDOM, 2008).

### **2.3.3 - Segurança**

A segurança que um sistema domótico oferece é maior do que qualquer outro sistema pode fornecer, pois integra três áreas de segurança, que normalmente são controladas por diferentes sistemas (DOMÍNGUEZ; VACAS, 2006).

#### **2.3.3.1 - Segurança de bens**

Controle e gerenciamento do acesso e presença de pessoas entre os ambientes residenciais, bem como a simulação de presença manipulando as luzes, o aquecimento, a ventilação, televisores e rádios para fazer com que a residência

pareça e soe estar ocupado em situações onde os moradores estão fora, alarmes contra invasões, circuito fechado de TV (CFTV) que interage com o sistema domótico.

#### 2.3.3.2 - Segurança pessoal

Especialmente para os idosos, doentes e portadores de deficiência. Alarmes pessoais para avisos de pânico ou SOS, usado para emergências como intrusão, roubo ou ataques pessoais, e também serviço de apoio para chamar a atenção em caso de necessidade de assistência pessoal ou acidentes dentro da residência e automaticamente fazer uma chamada para os serviços de ambulância e policial.

#### 2.3.3.3 - Incidentes e falhas

Utilizando sensores, ele pode detectar incêndios e vazamentos de gás e água, e pelo telefone acionar o alarme para os bombeiros, por exemplo. Se houver vazamento de gás cortar automaticamente a saída de gás. Em caso de detectar fumaça abrir as janelas para a fumaça sair, alertar a falta de energia e falha na linha telefônica.

### 2.3.4 - Comunicações

Os serviços e aplicativos de comunicação permitem a troca de mensagens entre pessoas e equipamentos dentro da casa e da casa para o exterior, a comunicação com o exterior é essencial para acessar a variedade de serviços oferecidos pelas operadoras de telecomunicações. Uma das características fundamentais da Domótica, que é a integração de sistemas, existe graças ao gateway residencial, que interliga a rede domótica com diferentes dispositivos, tais como Internet, rede de telefonia e rede de dados. (CEDOM, 2008).

## 2.4 – EQUIPAMENTOS UTILIZADOS

Há diversos dispositivos diferentes em um sistema de automação residencial desde centrais de controle para sistemas centralizados a um simples controle de uma lâmpada automática. O número de dispositivos domóticos distribuídos pela residência só é superado por nossas necessidades. Todos esses dispositivos domóticos podem ser utilizados e combinados de diferentes formas, dependendo das necessidades desejadas dos moradores e funcionalidades do sistema domótico. Esses diferentes dispositivos do sistema domótico podem ser classificados de acordo com os seguintes grupos: Controlador, Sensor, Atuador e Interface.

### 2.4.1 – Controlador

O controlador é o elemento que gerencia o sistema domótico, podendo ser um único controlador ou múltiplos distribuídos pela residência. Nele reside toda a inteligência do sistema, normalmente todos os outros elementos do sistema se conectam ao controlador enviando e recebendo informações. O controlador é responsável pela administração dos atuadores e sensores e das respostas retornadas para as interfaces (CASTILLO, 2009) e (BOLZANI, 2004).

### 2.4.2 – Sensor

O sensor é o dispositivo responsável por monitorar qualquer mudança física no ambiente, capturando as informações e transmitindo para o controlador, que desta forma pode interpretar o ambiente e saber se está claro ou escuro, frio ou calor, se há pessoas no ambiente, se as janelas e portas se encontram fechadas ou abertas. Fornecem comandos de entrada para o sistema (CASTILLO, 2009), (BOLZANI, 2004) e (HUIDOBRO et al., 2007) e (CALLONI, 2011).

A Figura 2 mostra alguns sensores.



**Figura 2 – Sensores (CASTILLO, 2009)**

Os sensores são um dos principais equipamentos necessários para um sistema domótico, pois é com ele que o sistema vai captar os diversos estímulos que ocorrem dentro de uma residência e converter estes estímulos em sinais elétricos e manda-los para o controlador, para que este mande um sinal para os atuadores executarem alguma ação de acordo com o que foi programado. Cada tipo de sensor é adequado para um determinado tipo de percepção e quanto mais sensores instalados melhor (HUIDOBRO et al., 2007)

### **2.4.3 – Atuador**

O atuador é um dispositivo capaz de receber comandos do controlador e transformar esses comandos em ações físicas no ambiente (ligar/desligar, subir/descer e abrir/fechar). Recebem comandos de saída do sistema (CASTILLO, 2009), (BOLZANI, 2004), (HUIDOBRO et al., 2007) e (CALLONI, 2011).

A Figura 3 mostra alguns atuadores.



**Figura 3 – Atuadores (CASTILLO, 2009)**

Os atuadores são distribuídos por toda a residência a fim de interpretar os comandos recebidos do controlador e efetuar a ação a que é destinado, através das interfaces os moradores tem controle a todos os atuadores, e podem administrar a residência de uma forma dinâmica.

#### **2.4.4 – Interface**

A interface refere-se aos dispositivos como telas, *smartphones*, *tablets*, internet e formatos como binário e áudio, em que se podem mostrar as informações do sistema domótico para os usuários ou outros sistemas, que assim podem interagir e comandar todo o sistema a partir dessas interfaces (HUIDOBRO et al., 2007) e (CALLONI, 2011).

### **2.5 – ARQUITETURA DA DOMÓTICA**

A arquitetura do sistema domótico, como em qualquer sistema de controle, aponta como os diferentes dispositivos de controle do sistema são distribuídos. Essas arquiteturas são: arquitetura centralizada, arquitetura descentralizada e arquitetura distribuída (híbrida).

Para definir o tipo de arquitetura utilizada no sistema domótico é preciso identificar primeiramente os tipos de equipamentos utilizados e as informações que eles vão

gerenciar. Em função destas informações analisadas pode-se definir o tipo de arquitetura utilizada no sistema domótico a ser implantado (DOMÍNGUEZ; VACAS, 2006).

### 2.5.1 – Centralizada

Nessas arquiteturas, há um controlador central, encarregado de processar a informação recebida dos vários sensores, dispositivos, interfaces e usuários, ou enviar comandos para os atuadores agir com o ambiente e para as interfaces. Se um sensor envia um sinal ao controlador central, isto de acordo com a programação definida pelo usuário este deve enviar uma série de comandos para os atuadores. Eles são sistemas modulares que podem continuar adicionando placas para expandir seu alcance (HUIDOBRO et al., 2007). A Figura 4 demonstra essa arquitetura.

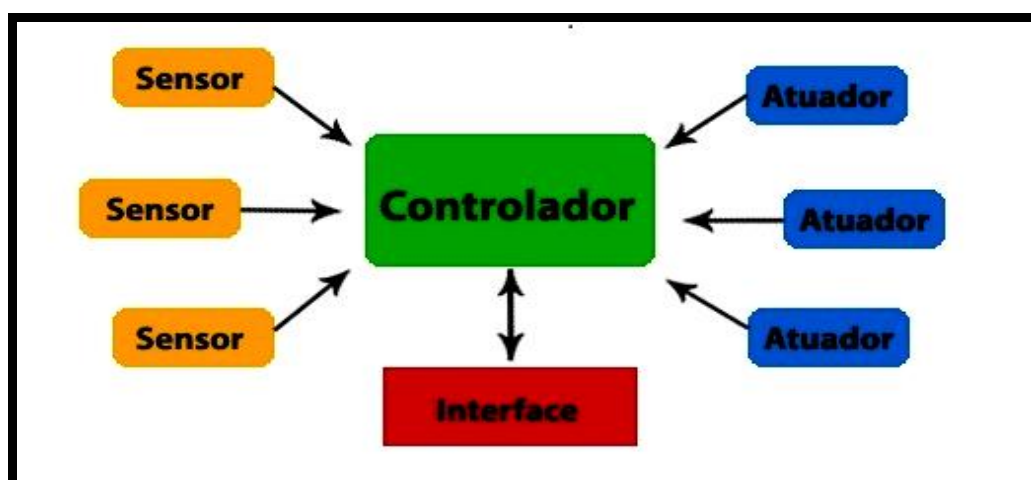


Figura 4 – Arquitetura Centralizada (Adaptado de HUIDOBRO et al., 2007)

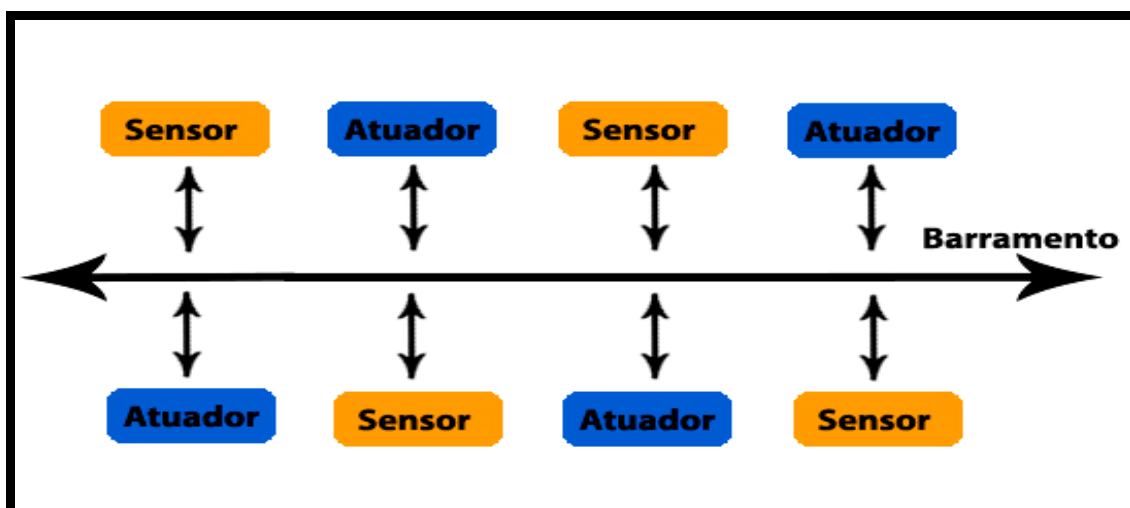
Algumas de suas vantagens são: dispositivos como sensores e atuadores são universais, custo reduzido ou moderado, fácil uso e formação e instalação simples.

E suas desvantagens são: cabeamento significativo, sistema dependente do funcionamento ideal da central, modularidade difícil, escalabilidade reduzida e capacidade do sistema (canais ou pontos).



### 2.5.2 – Descentralizada

São arquiteturas que permitem a execução de vários serviços. Neste sistema de barramento compartilhado, cada dispositivo tem a capacidade de tratar a informação recebida e podem atuar de forma independente, dispõe de funções de controle e comando. Possui elevado grau de flexibilidade, permitindo adicionar dispositivos ao barramento sem problemas e em qualquer lugar na rede. Ele pode captar informações próprias ou receber de outros dispositivos do sistema (HUIDOBRO et al., 2007). A Figura 5 demonstra essa arquitetura.



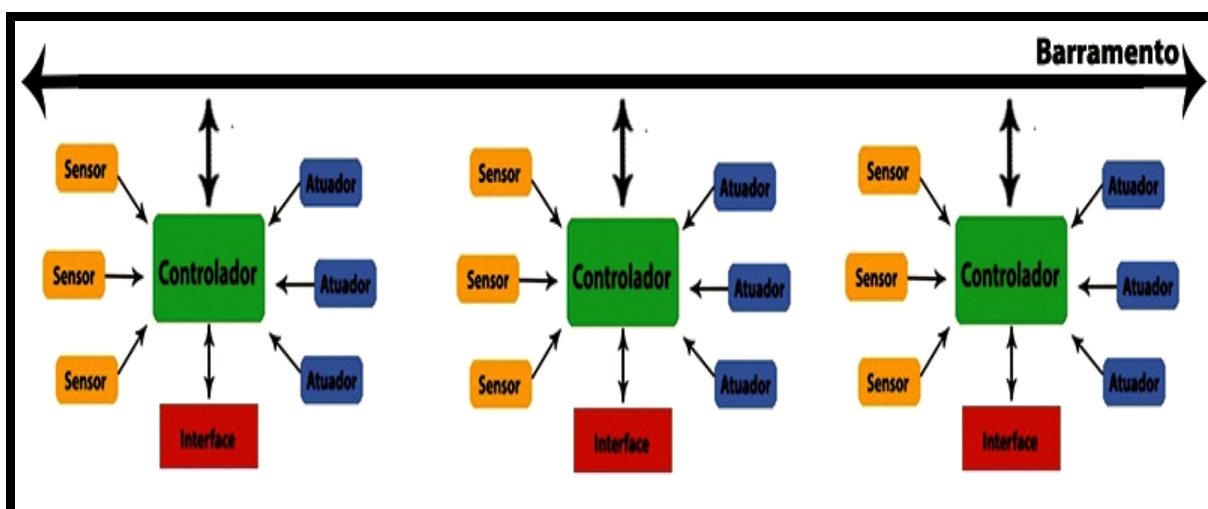
**Figura 5 – Arquitetura Descentralizada (Adaptado de HUIDOBRO et al., 2007)**

Algumas de suas vantagens são: segurança de funcionamento, possibilidade de reformulação da rede, fiação reduzida, confiabilidade de produtos e fácil expansibilidade.

E suas desvantagens são: dispositivos sensores não universais e limitadas variações, alto custo da solução, mais próximos de “edifícios inteligentes” que “casas inteligentes” e complexidade de programação.

### 2.5.3 – Distribuída (híbrida)

Na arquitetura distribuída (híbrida) são combinadas as arquiteturas centralizadas e descentralizadas. Seu controle está localizado em cada controlador que tem acesso a uma serie limitada de dispositivos da rede domótica. Para a comunicação entre os diversos controladores é preciso um protocolo para que suas ações sejam coordenadas (HUIDOBRO et al., 2007). A Figura 6 demonstra essa arquitetura.



**Figura 6 – Arquitetura Distribuída (Adaptado de HUIDOBRO et al., 2007)**

Algumas de suas vantagens são: segurança de funcionamento, possibilidade de reformulação da rede, fácil expansibilidade e sensores e atuadores do tipo universal (econômicos e de grande oferta).

E sua desvantagem é: como único inconveniente destacado é que exige programação ou configuração.

## 2.6 – REDE DOMÉSTICA

Segundo Bonzani (2004) “Rede doméstica é aquela que provê a interligação entre os equipamentos do assinante, como computadores, dispositivos inteligentes, sensores, atuadores e o *gateway* residencial, como o modem ADSL”.

A rede doméstica cabeada muito utilizada para interligar computadores, impressoras, scanners. Também pode ser utilizada para a domótica interligar seus dispositivos dentro da residência utilizando uma estrutura já disponível em muitas casas atualmente. O *WiFi* está se popularizando e sendo muito utilizado em diversos aparelhos como notebooks, televisores, *smartphones*, entre outros, interligando esses aparelhos em uma rede sem fio doméstica de comunicação e troca de dados.

A rede doméstica como vemos na Figura 7, pode ser classificada em rede de dados, rede multimídia, rede de segurança e rede domótica (HUIDOBRO et al., 2007), (DOMÍNGUEZ; VACAS, 2006) e (CALLONI, 2011).

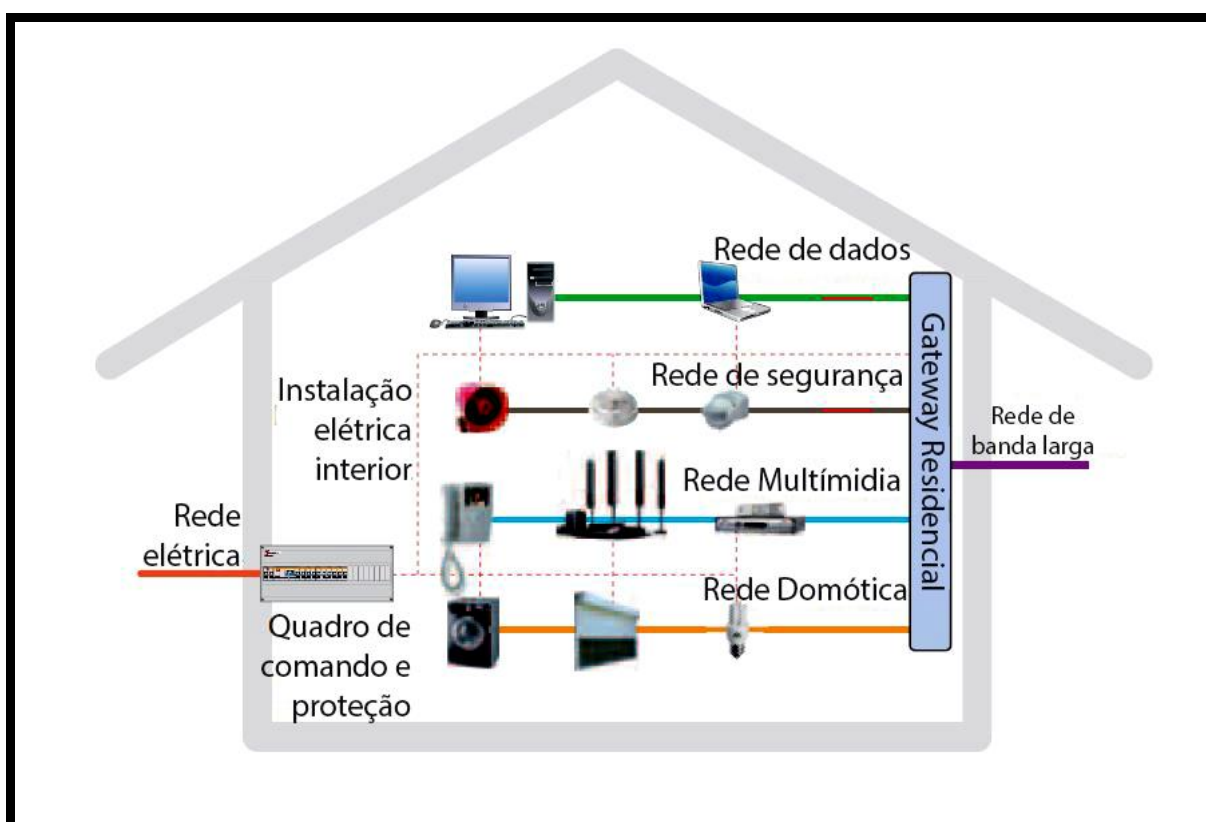


Figura 7 – Rede Doméstica (CASTILLO, 2009)

- Rede Domótica: É o barramento onde ocorre o transporte das informações permitindo a comunicação entre diferentes dispositivos existentes no sistema domótico, podendo ser um barramento próprio do sistema ou utilizar a rede de outros sistemas como a rede elétrica, rede telefônica, rede de dados ou rede sem fio.
- Rede de dados: É um tipo de rede que foi feita inicialmente para fins empresariais, e atualmente está sendo aplicada para o uso doméstico. Permite utilizar uma mesma rede de área local (LAN) para compartilhar arquivos, dispositivos, aplicações e conexões com a internet simultaneamente com outros aparelhos dentro da residência.
- Rede Multimídia: Esta rede gerencia os serviços de entretenimento, distribuindo áudio e vídeo por toda a residência mediante equipamentos distribuidores, gerencia as interfaces para o usuário e dispositivos de recepção.
- Rede de Segurança: Se encarrega de integrar os elementos que interveem na segurança da residência e de seus habitantes.

O gateway residencial faz a interligação da rede doméstica com a internet e possibilita o monitoramento da rede domótica através de qualquer meio com internet e em qualquer lugar do mundo.

A rede domótica, rede dados, rede multimídia e rede segurança interligadas formam a rede doméstica que prove ao usuário controle e acesso a todas as funções que a residência pode oferecer tornando seu gerenciamento mais fácil e acessível a todos os moradores da residência.

## 2.7 – MEIOS DE TRANSMISSÃO

Em todo o sistema domótico diferentes meios de transmissão podem ser utilizados para trocar informações entre os dispositivos. Cada protocolo usa um meio de

transmissão específico. Na Tabela 1 são mostrados alguns dos meios de transmissão existentes.

<b>Tipo</b>	<b>Custo</b>	<b>Usabilidade</b>	<b>Características e requerimentos</b>
<b>Par trançado</b>	Baixo	Usada para rede de computadores.	Grande segurança de transmissão
<b>Cabo Coaxial</b>	Muito Elevado	Utilizado no envio de sinais de vídeo. Bastante implantado.	Imune a interferências, mas muito rígida.
<b>Rede elétrica</b>	Nenhum	Não a necessidade de instalação adicional.	Pouca segurança e velocidade.
<b>Fibra Ótica</b>	Elevado	Grande capacidade	Utiliza-se para transmitir grande quantidade de informações
<b>Bluetooth</b>	Baixo	Bastante difundido	Não é um padrão, admite velocidade de transmissão média
<b>WiFi</b>	Alto	Bastante difundido	É um padrão, admite velocidade de transmissão alta.

**Tabela 1 - Meios de Transmissão**

Os meios de transmissão estão divididos em duas categorias a transmissão por cabos e a transmissão sem fios, como exemplificado a seguir.

Transmissão por cabos: A transmissão por cabos é utilizada por vários sistemas domóticos por ser mais viável e rápida, os dispositivos são ligados por meio de cabos. Os principais tipos são: Cabeamento dedicado, Par trançado, Coaxial, Fibra ótica, Rede elétrica (HUIDOBRO et al., 2007) e (CASTILLO, 2009)

Transmissão sem fios: A transmissão sem fio está sendo atualmente muito difundida para aplicações domóticas, sua principal vantagem é não depender de um meio físico para a transmissão dos dados. Os principais tipos são: *WiFi*, *Bluetooth*, Radiofrequência, Infravermelho, *ZigBee* (HUIDOBRO et al., 2007).

## 2.8 – ACESSIBILIDADE

Um dos conceitos fundamentais da domótica é a acessibilidade, pois facilita o acesso a todas as funcionalidades de uma residência para seus moradores, sejam eles crianças, adultos, ou portadores de alguma deficiência seja ela física ou mental.

É possível melhorar a qualidade de vida de pessoas que se encontram enfermas ou que possuam alguma deficiência, tornando essas pessoas mais independentes podendo controlar suas residências através de interfaces como tablets e smartphones.

O objetivo da acessibilidade é fazer com que as pessoas com deficiência ou enfermas possam ter acesso às tecnologias, porque as próprias tecnologias não são um objetivo, mas um meio para promover a autonomia pessoal.

Segundo Farias e Buchalla (2005), “a funcionalidade e a incapacidade dos indivíduos são determinadas pelo contexto ambiental onde as pessoas vivem”. Em um ambiente adaptado para pessoas com deficiência, essas deficiências são superadas e é isso que a domótica proporciona para seus usuários, maior conforto, segurança e bem estar.

## 2.9 – PROTOCOLOS E PADRÕES EXISTENTES

Um sistema domótico é caracterizado pelo padrão e protocolo de comunicação que se utiliza, que é nada mais do que a língua ou formato das mensagens que controlam os diferentes dispositivos e que cada um deve usar para se comunicar um com os outros de uma forma coerente. Dentre os protocolos existentes, pode-se realizar uma primeira classificação de acordo com sua padronização, a Tabela 2 demonstra alguns dos padrões existentes na domótica com algumas de suas vantagens e desvantagens (BOLZANI, 2004) e (MACEDO, 2009).

Tecnologia	Protocolo	Suporte Físico	Velocidade	Alcance	Vantagens	Desvantagens
<b>X10</b>	Aberto	Rede Elétrica	50bps	185 m	- Não necessita de novos cabos para a instalação doméstica - Maior confiabilidade	- Baixa velocidade de transmissão
<b>Lonworks</b>	Aberto	Todos	78 Kbps a 1.28 Mbps	1500m a 2700m	- Alta velocidade de transmissão - Padrão global e simples programação	- Tecnologia custosa
<b>KNX</b>	Aberto	UTP,RF, Par transado	9600 bps 1200/2400 bps	600m 1000m	- Fácil instalação e configuração - Maior distância de transferência - Compatibilidade entre equipamentos	- Baixa velocidade de transmissão
<b>CEBus</b>	Aberto	Todos	10000 bits	Depende do meio	- Não requer controladores centrais - Expansibilidade da rede - Tecnologia econômica	- Baixa velocidade de transmissão - Não cumpre as normas
<b>Zigbee</b>	Aberto	RF	300 Kbps	10m a 100 m	- Baixo custo - Baixo consumo de energia	- Baixa velocidade de transmissão
<b>HomeRF</b>	Aberto	RF	10 Mbits	50 m	- Permite sinais de telefonia e sinais de dados ao mesmo tempo	- Caiu em obsolescência

**Tabela 2 - Padrões da Domótica (BOLZANI, 2004), (ARAUJO; PEREIRA, 2004) e (MACEDO, 2009)**

Alguns dos padrões mais utilizados na domótica são descritos a seguir:

- *CEBus (Consumer Electronics Bus)*: O padrão *CEBus* é um protocolo criado pela Associação de Indústrias Eletrônicas (EIA) para ser possível a interligação e comunicação entre dispositivos eletrônicos de uma residência (ARAUJO; PEREIRA, 2004) e (BOLZANI, 2004).
- *HomeRF (Home Radio Frequency Working Group)*: A missão do grupo de trabalho *HomeRF* é tornar possível uma vasta gama de produtos eletrônicos de consumo que operem entre si, estabelecendo uma especificação aberta

para comunicações digitais de RF (Radiofrequência), entre computadores e produtos eletrônicos de consumo residencial (BOLZANI, 2004).

- *Zigbee*: Pensa-se que este pode ser um dos padrões que será bastante utilizado no mundo da domótica. É uma versão melhorada do *HomeRF* e com um baixo consumo de energia, mas também baixa velocidade de transmissão (BOLZANI, 2004).
- *Lonworks*: É basicamente uma plataforma de controle que descreve efetivamente uma solução completa para os problemas de sistemas domóticos, existem diversos dispositivos *Lonworks* cada dispositivo do nó da rede tem uma certa autonomia e sua comunicação só é realizada com outros dispositivos *Lonworks* (ARAUJO; PEREIRA, 2004) e (BOLZANI, 2004).
- *X-10*: Protocolo mais utilizado na domótica, utiliza a rede elétrica e facilita o controle de dispositivos domóticos sem instalação de qualquer fio adicional em casa, por utilizar a instalação de rede elétrica já existente. Tem pouca confiabilidade contra ruídos elétricos (ARAUJO; PEREIRA, 2004) e (BOLZANI, 2004).
- *KNX*: É uma tecnologia que resulta da convergência dos sistemas de controle europeu *Batibus*, *EIB* e *EHS*, com o objetivo de criar uma norma europeia única para a automação de residências e escritórios (BOLZANI, 2004).

Existem dois tipos de protocolos, os protocolos padrão que são aqueles que de algum modo são amplamente utilizados por diferentes empresas e estes fazem produtos que são compatíveis com os diferentes protocolos padrão existentes, tais como: *X-10* e *KNX*. E os protocolos proprietários que são aqueles desenvolvidos por uma só empresa, e são capazes de se comunicar apenas com dispositivos com um mesmo protocolo proprietário.

## 2.10 – DOMÓTICA NO BRASIL

Atualmente o Brasil está crescendo muito em tecnologia, o setor domótico nos últimos quatro anos cresceu 300% de acordo com dados da Aureside (Associação



brasileira de automação residencial) e cerca de 250 empresas no Brasil atualmente estão no ramo da domótica.

Em 2012, o seguimento faturou mais ou menos R\$ 4 bilhões e a previsão para o ano de 2013 é de um crescimento de mais 30% no setor. O mercado domótico está em expansão, pois se estima que cerca de 1,8 milhões de residências estejam aptas para a chegada da domótica (PEQUENAS EMPRESAS GRANDES NEGÓCIOS, 2013).

Existem pelo mundo diversas entidades e instituições reguladoras da domótica que buscam elaborar um padrão para fomentar a domótica, na Europa e Estados Unidos como o conceito de domótica já está bem difundido existem diversas entidades e associações como o *European Installation Bus Association* (EIBA), da Europa e o *National Association of Home Builders* (NAHB) do Estados Unidos, entre outros. No Brasil em fevereiro de 2000 foi registrada a AURESIDE (Associação Brasileira de Automação Residencial), que tem como missão divulgar conceitos a todos os envolvidos no setor, difundir tecnologias, homologar produtos e serviços, treinar e formar profissionais na área da domótica (AURRSIDE,2013).

### 3 - COMPUTAÇÃO FÍSICA COM ARDUINO

A computação física engloba todas as disciplinas que permitem desenvolver objetos iterativos que podem se comunicar com as pessoas através de sensores e atuadores controlados através de um software embarcado em um microcontrolador (BANZI, 2012).

A alma de um sistema embarcado é o seu microcontrolador, que é basicamente seu processador, sua memória e suas interfaces de entrada e saída todas dentro de um mesmo chip. Quase todos os equipamentos eletrônicos atualmente tem em seu interior um sistema embarcado.

Embora existam muitas outras plataformas de microcontroladores disponíveis, para a computação física, o mais simples para se trabalhar é provavelmente o *Arduino*. Isto é em parte devido à grande comunidade envolvente da plataforma, mas também porque o Arduino foi projetado desde o início para ser simples para os iniciantes (ALLAN, 2011).

O *Arduino* é uma plataforma de computação física de fonte aberta para criação de protótipos de eletrônica baseada no conceito de software e hardware livres, e de fácil utilização, projetada com um microcontrolador *Atmel AVR* de placa única, com suporte de entrada e saída embutido e uma linguagem de programação padrão (EVANS, 2011), (BANZI, 2012) e (ARDUINO, 2013).

Segundo Banzi, 2012 o diferencial do *Arduino* em relação a outras plataformas presentes no mercado se deve por:

- Trata-se de um ambiente multiplataforma; pode ser executado em Windows, Macintosh e Linux.
- Pode ser programado utilizando-se um cabo USB, não necessita de porta serial.
- O hardware é barato.
- É um hardware e software de fonte aberta (*open-source*).

- Há uma comunidade ativa e colaborativa de usuários
- É ideal para iniciantes que desejam resultados rápidos

Pode-se utilizar o *Arduino* para interagir com o ambiente onde ele se encontra através de sensores acoplados em suas entradas, e com esses dados coletados realizar as tomadas de decisões para que o mesmo interaja com o ambiente. Um exemplo de seu uso: com a utilização de um sensor de presença desenvolver um programa no Arduino para que quando alguém entre em um quarto que tenha o sensor de presença, ele acenda a luz desse quarto automaticamente.

Assim à possibilidade de criar diversos dispositivos para interagir com o meio e facilitar o dia a dia das pessoas e com isso automatizar muitas ações do cotidiano.

### 3.1 – HISTÓRIA DO ARDUINO

O projeto *Arduino* foi criado na Itália, em 2005 como resultado da colaboração de muitas pessoas, mas seu início veio da necessidade de uma nova plataforma de prototipagem fácil e barata pensada por David Cuartielles e Massimo Banzi, ambos professores de computação física que se conheceram no *Interaction Design Institute Ivrea*, idealizaram o Arduino para ensinar seus alunos a criar protótipos interativos e para complementar o aprendizado de programação, computação física e gráfica.

Atualmente a equipe *Arduino* mostrada na Figura 8 é composta por Massimo Banzi, David Cuartielles, Tom Igoe, Gianluca Martino e David Mellis.



**Figura 8 – Equipe Arduino (Cuartielles, 2008)**

Seus predecessores incluem os microprocessadores *BASIC Stamp*, que teve o seu início em 1992, bem como o *OOPic*, *ATOM Basic*, *BASICX24*, e o *PICAXE* (EVANS, 2011).

Seu sucesso foi sinalizado com o recebimento de uma menção honrosa na categoria *Comunidades Digitais* em 2006, pela *Prix Ars Electronica*, além da marca de mais de 50.000 placas vendidas até outubro de 2008 (ARDUINO, 2013).

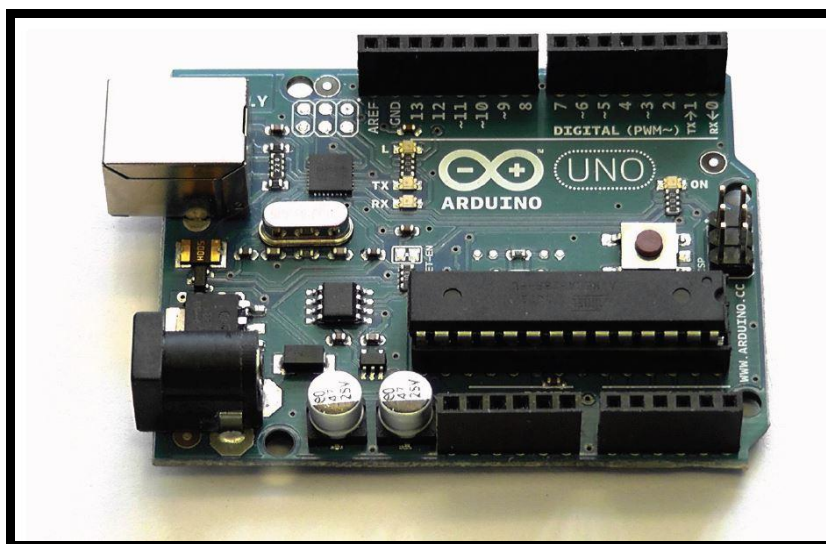
### 3.2 – HARDWARE ARDUINO

A atual revisão da placa *Arduino* é conhecida como *Arduino Uno*. Esta placa baseia-se no microcontrolador ATmega328. Tem quatorze pinos digitais de entrada e saída, seis dos quais podem ser usados como saídas de modulação por largura de pulso (PWM), juntamente com mais seis pinos de entrada analógica conforme mostrado na Tabela 3 (SCHMIDT, 2011) e (ARDUINO, 2013).

Microcontrolador	ATmega328
Voltagem de operação	5V
Voltagem de entrada (recomendado)	7-12V
Voltagem de entrada (limites)	6-20V
Pinos Digitais I/O	14 (dos quais 6 podem ser Saídas PWM)
Pinos Entrada Analógica	6
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Memória Flash	32 KB (ATmega328) dos quais 0,5 KB utilizados pelo bootloader
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Frequência de Clock	16 MHz

**Tabela 3 - Resumo da placa Arduino (ARDUINO, 2013)**

A plataforma *Arduino* tem tudo o que é necessário para suportar um microcontrolador embarcado. Ele pode ser ligado diretamente ao seu computador para programar usando uma conexão USB, e pode ser alimentada através da mesma conexão USB ou com uma fonte de alimentação externa, uma vez feita a programação e você quiser retirar a placa do seu computador, a alimentação externa manterá o funcionamento da mesma. O *Arduino* pode operar com uma fonte externa de 6 a 20 volts. A opção mais fácil é usar uma bateria ou um adaptador de energia 9V, na Figura 9 é mostrado o hardware *Arduino UNO* (ALLAN, 2011), (KELLY; TIMMIS, 2013) e (ARDUINO, 2013).



**Figura 9 – Hardware Arduino**

O *Arduino* utiliza a porta USB para passar os *sketches* para sua memória interna e também pode interagir em tempo real de forma muito fácil através da comunicação serial com programas de várias linguagens de programação diferentes.

### 3.2.1 - Entrada e Saída

Cada um dos 14 pinos do *Uno* pode ser usado como uma entrada ou uma saída. Operam a 5 V com uma corrente limite de 40 mA. Cada pino tem também um resistor pull-up interno de 20-50 kOhms, embora este seja desligado por padrão.

Alguns dos pinos têm funções especializadas, como os pinos 0 e 1. Estes pinos podem ser utilizados para receber (RX) e transmitir (TX) dados seriais TTL. Estes pinos são ligados aos pinos correspondentes do ATmega8U2 e, conseqüentemente, ligado à conexão USB serial, para conexão com o computador (ALLAN, 2011) e (ARDUINO, 2013).

### 3.2.2 – Comunicação

O ATmega328 fornece comunicação serial UART TTL a 5 V, e está disponível nos pinos digitais 0 (RX) e 1 (TX). O *Arduino Uno* tem um chip ATmega8U2 acoplado que redireciona esta comunicação serial através do USB, permitindo que o *Arduino* crie uma porta serial virtual para seu computador. Se você estiver usando uma placa mais antiga, como o *Duemilanove* ou *Diecimila*, estas placas fazem uso do driver FTDI USB-para-serial, com um chip para realizar a mesma tarefa, mas ao contrário destas, com a placa *Uno*, você é obrigado a instalar um driver para que o seu computador identifique a placa corretamente (ALLAN, 2011) e (ARDUINO, 2013).

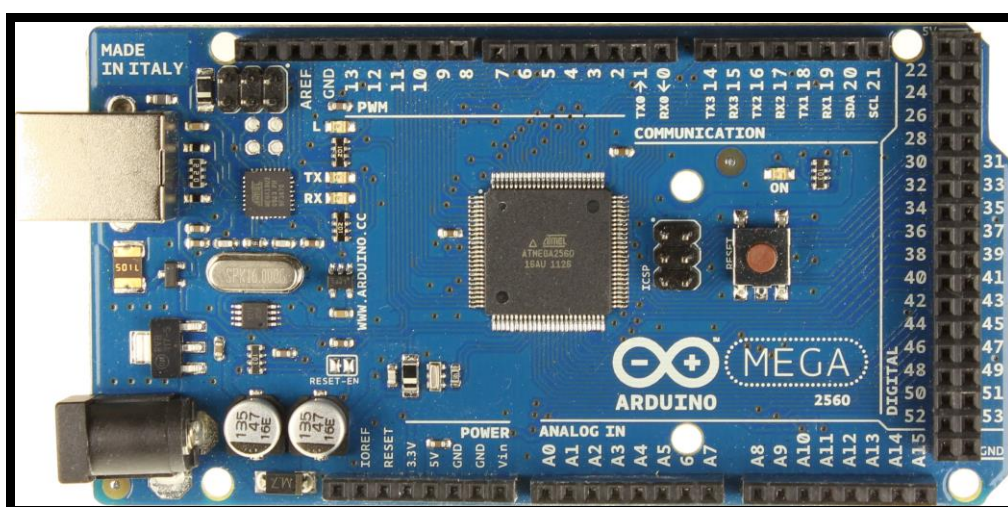
### 3.2.3 – Modelos De Arduino

Existem diversos modelos de *Arduino*, cada um com suas particularidades e aplicações, para empregar dependendo do que se deseja fazer, com diferentes formatos e configurações de hardware. O *Arduino Uno* mostrado no item 3.2 é um dos mais utilizados atualmente. Temos também o *Arduino Mega*, o *Arduino Nano*, uma versão reduzida de um *Arduino* comum. O *Arduino BT*, com uma conexão *Bluetooth* embutida na placa. O *Arduino LilyPad* utilizado para criação de iterações com roupas. Além dos *Arduino* da loja oficial temos vários fabricantes independentes

de *Arduino* como o *Severino* e o *Tatuino*, que são clones desenvolvidos no Brasil, o *Program-ME* também brasileiro desenvolvido pela *Global Code*, entre tantos outros modelos existentes no mundo inteiro (SCHMIDT, 2011) e (ARDUINO, 2013).

### 3.2.3.1 – Arduino Mega

O *Arduino Mega* mostrado na Figura 10, possui muito mais portas, permitindo a criação de dispositivos maiores e mais complexos, com eficiência e baixo custo.

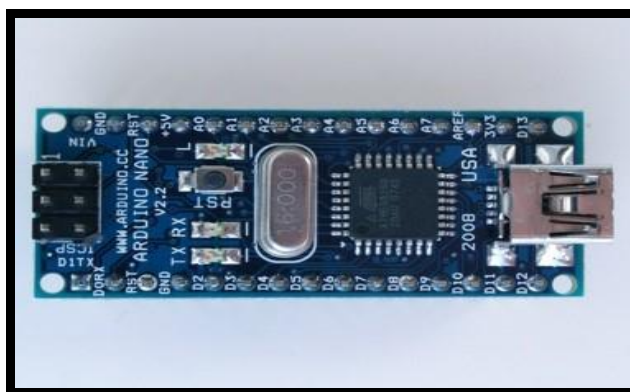


**Figura 10 – Arduino Mega (Arduino, 2013)**

O *Arduino Mega* 2560 é uma placa de microcontrolador baseada no ATmega2560. Ele possui 54 pinos de entradas/saídas digitais (dos quais 15 podem ser usados como saídas PWM), 16 entradas analógicas, 4 UARTs (portas seriais de hardware), um oscilador de cristal de 16 MHz, uma conexão USB, uma entrada de alimentação, uma conexão ICSP e um botão de reset. Ele contém tudo o que é necessário para dar suporte ao microcontrolador; basta conectar a um computador com um cabo USB ou a uma fonte de alimentação e já está pronto para começar. O *Mega* é compatível com a maioria dos Shields desenhados para os *Arduino Uno*. (Arduino, 2013).

### 3.2.3.1 – Arduino Nano

O *Arduino Nano* mostrado na Figura 11 é uma pequena versão do *Arduino* parecida com o *Arduino UNO*, pois também possui um chip ATmega328, na versão SMD. Possui um conector para cabos Mini-USB para gravação. Uma das diferenças entre esta placa e a *Arduino UNO*, é que esta placa possui 2 entradas analógicas a mais e um jumper de +5V AREF. Esta placa não possui um conector para fonte externa, mas é possível alimentá-la pelo pino Vin. O *Arduino Nano* automaticamente seleciona a maior alimentação fornecida (ARDUINO, 2013).



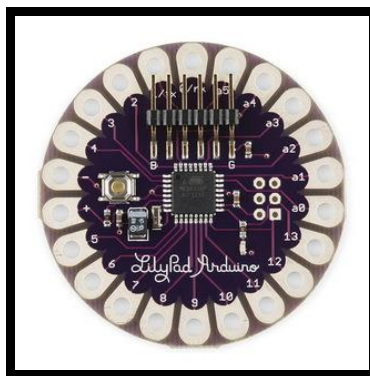
**Figura 11 – Arduino Nano (ARDUINO, 2013)**

O *Arduino Nano* é ideal para projetos que requerem pouco espaço, com seu tamanho reduzido podemos coloca-los em lugares que os outros *Arduinos* não caberiam.

### 3.2.3.1 – Arduino LilyPad

O *Arduino LilyPad* mostrado na Figura 12 é uma placa de microcontrolador baseado no ATMEGA168V (versão de baixa potência do ATmega168) ou o ATmega328V. Ele tem 14 pinos digitais (dos quais 6 fornece uma saída de PWM) e 6 pinos de entradas analógicas, um oscilador de cristal de 8 MHz, uma conexão micro USB, um conector JST para a 3.7V *LiPo* bateria e um botão de reset (ARDUINO, 2013).



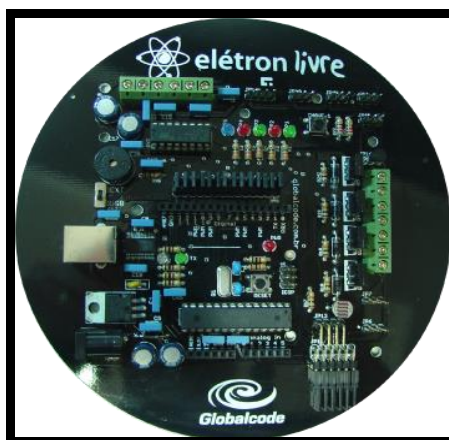


**Figura 12 – Arduino LilyPad (ARDUINO, 2013)**

Ele contém tudo o necessário para suportar o microcontrolador, basta conectá-lo a um computador com um cabo USB ou ligá-lo com uma bateria para começar. O *LilyPad* foi criado para poder ser utilizado em vestimentas, podendo ser costurado diretamente sobre tecidos com fios condutores de energia.

### 3.2.3.1 – Program-ME

O Program-ME é uma placa controladora compatível com Arduino desenvolvida pela Global Code, que facilita o desenvolvimento de projetos, pois conta com recursos extras que um *Arduino* convencional não tem e já embarca diversos componentes diretamente na placa (GLOBAL CODE, 2013).



**Figura 13 – Program-ME (GLOBAL CODE, 2013)**

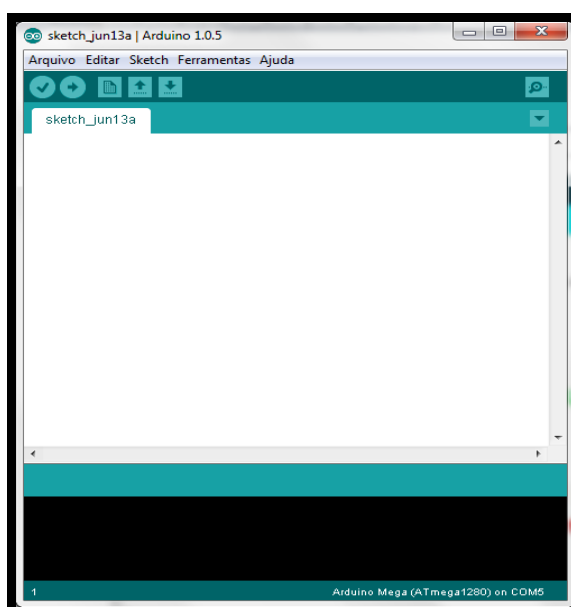
Ele vem com mais de 12 funções como: sensor de luz, sensor de temperatura, entradas para servo-motor, *speaker*, leds entre outras. O tem facilita muito, pois não precisa de equipamentos adicionais para iniciar os projetos com ele.

### 3.3 – IDE ARDUINO

A *IDE Arduino* foi escrita em Java e é extremamente intuitiva derivada do projeto *Processing* desenvolvido por Ben Fry e Casey Reas. Pode se comunicar com várias linguagens de programação, através de uma conexão serial, como exemplo: *Processing*, *Pura Data*, *Python*, *Java* entre outras (ARDUINO, 2013).

A *IDE Arduino* mostrada na Figura 14 contém um editor de texto para a escrita de código que posteriormente será carregado na placa *Arduino*, os programas no *Arduino* são chamados de *sketches*, ele possui também uma área de mensagens (console), uma barra de ferramentas com alguns botões principais e menus. Ela se conecta ao *hardware* do *Arduino*, permitindo o carregamento de programas e a comunicação com a placa.

Os *sketches* escritos no editor de texto são salvos com a extensão “.ino”. Na área de mensagem, tem-se o retorno das ações executadas pela *IDE*, e a exibição de erros.



**Figura 14 – IDE Arduino**

Na Figura 15 são mostrados seus principais botões, seguida de suas descrições.



**Figura 15 – Botões IDE Arduino**

- Verificar: Para Verificar se o programa está sem erros.
- Carregar: Carrega o programa no Microcontrolador.
- Novo: Carrega um novo *Sketch*
- Abrir: Abre um *Sketch* salvo no computador
- Salvar: Salva o *Sketch* no computador
- Monitor Serial: Ativa a leitura da porta serial. Mostra valores.
- *Sketch*: é o nome que o Arduino usa para um programa. É a unidade de código que é carregado e executado em uma placa Arduino.
- Área de notificação: Área de status da programação e da ligação com o *Arduino*.
- Barra de status: informa o status do *Arduino*

A Figura 16 mostra a área de desenvolvimento do *sketch*, a barra de status e a área de notificação do *Arduino*.



**Figura 16 – Sketch, Barra de status e Área de notificação do Arduino**

Atualmente a *IDE Arduino* está na versão 1.0.5, e foi projetada para ser uma interface amigável á pessoas que nunca apresentaram contato com o desenvolvimento de software, portanto é bastante intuitiva e de fácil utilização, se tornando ideal para pessoas que encontram-se começando no campo da programação (KELLY; TIMMIS, 2013), (SCHMIDT, 2011).

### 3.4 - LINGUAGEM ARDUINO

A sintaxe da linguagem *Arduino* é derivada do *Wiring* desenvolvido por Hernando Barragan, e é basicamente C/C++, possui funções simples e específicas para trabalhar com as portas do *Arduino*, e precisa de duas funções elementares para seu funcionamento: *setup ()* e *loop ()* (ARDUINO, 2013).

#### 3.4.1 – Funções no Arduino

Há inúmeras funções que podem ser utilizadas com o *Arduino*, destacando-se:

- **setup()**: No *Arduino* a função *setup ()* é chamada no momento em que o programa começa. É usada para inicializar variáveis, definir os modos de entrada ou saída dos pinos, indicar bibliotecas e outras tarefas. Essa função é executada somente uma vez, quando o *Arduino* é iniciado ou quando é resetado.
- **loop()**: No *Arduino* após a função *setup ()*, que inicializa e declaram os valores iniciais, a função *loop ()* faz precisamente o que seu nome indica: ela repete-se continuamente permitindo que seu programa funcione dinamicamente. É utilizada para controlar de forma ativa a placa *Arduino*.
- **pinMode()**: Configura o pino do *Arduino* apontado para que se comporte como entrada ou saída. Deve-se informar o número do pino que se deseja configurar e em seguida, se o pino será determinado como entrada (*INPUT*) ou saída (*OUTPUT*).
- **digitalWrite()**: Gera um valor *HIGH* (5 v ou 1) ou *LOW* (0 v ou 0) em um pino digital.
- **digitalRead()**: Lê o valor de um pino digital nomeado e retorna um valor *HIGH* ou *LOW*.
- **AnalogRead()**: Faz leitura de um pino analógico. O *Arduino* contém um conversor analógico-digital de 10 bits, com isto ele pode mapear voltagens de entrada entre 0 e 5 Volts para valores inteiros entre 0 e 1023
- **AnalogWrite()**: Gera um valor analógico entre 0 e 255. Onda PWM *Pulse Width Modulation* ou Modulação por Largura de Pulso (MLP) é um método para obter resultados analógicos com meios digitais, esta função no *Arduino UNO* está disponível nos pinos 3, 5, 6, 9,10 e 11. Para usar esta função deve-se informar o pino ao qual deseja escrever e em seguida informar um valor entre 0 e 255.
- **Serial.begin()** Ajusta o taxa de transferência em bits por segundo para uma transmissão de dados pelo padrão serial. Para comunicação pode se usar uma destas taxas: 300, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 57600, 115200.

- **delay()** Suspende a execução do programa pelo tempo (em milissegundos) mencionado. 1000 milissegundos são iguais a 1 segundo.

Estas funções fazem parte da linguagem de programação *Arduino* e torna a escrita do *sketch* simples e de fácil compreensão. Possibilitando que iniciantes comecem a programar para o *Arduino* de forma muito rápida e sem se intimidar com a linguagem.

### 3.4.2 – Bibliotecas

As bibliotecas no *Arduino* nos proporcionam opções de programação mais ampla e diversa, proporcionando várias novas funções para trabalhar com *Shields* e outros tipos de programação para ações específicas com o *Arduino*. Na *IDE* do *Arduino* já existem algumas bibliotecas padrões, mas além dessas bibliotecas, qualquer pessoa pode escrever uma biblioteca para o *Arduino* e distribuí-la para a comunidade.

## 3.5 - SHIELDS

*Shields* são placas que aprimoram as capacidades do *Arduino* e que podem ser plugadas na parte superior da placa *Arduino*, como mostrado na Figura 17 e se fundamentam no mesmo objetivo do *Arduino*, montagem simplificada, produção barata e de distribuição *open-source* (ARDUINO, 2013).

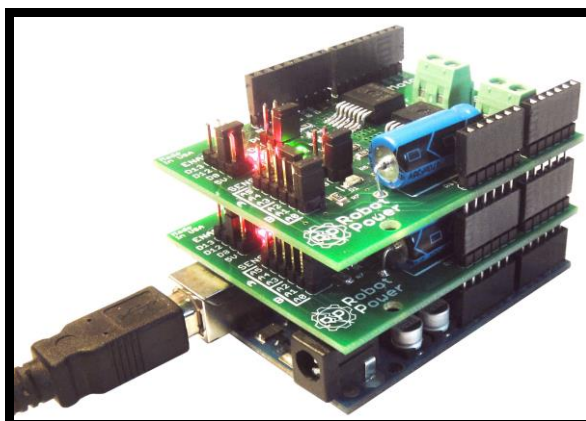
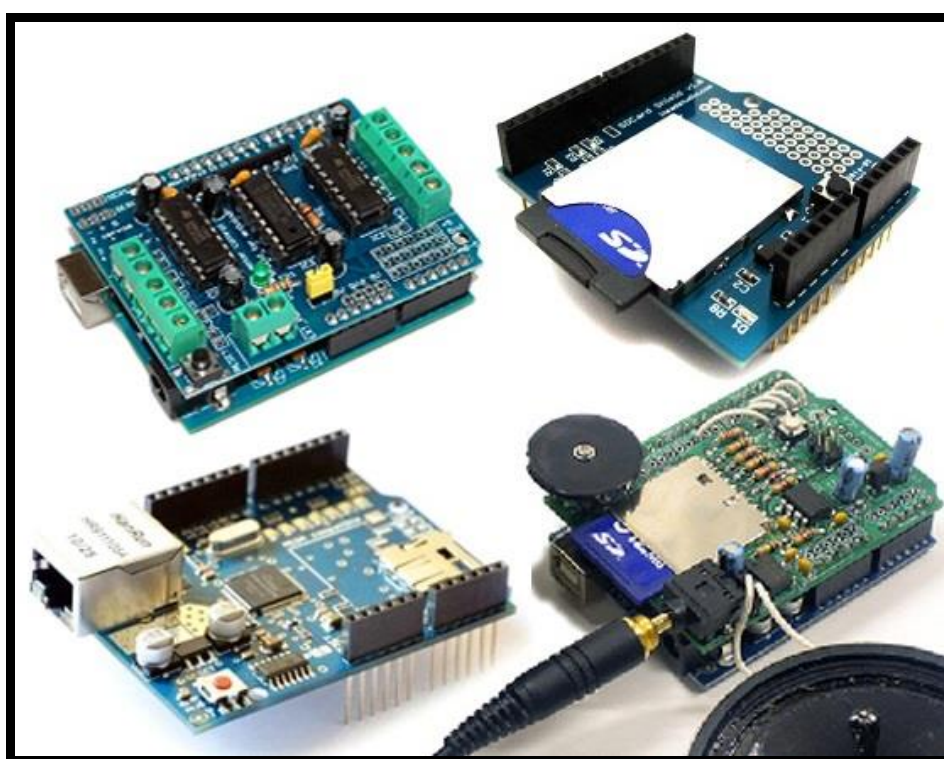


Figura 17 – Empilhamento de Shields

Atualmente você pode obter *Shields* de diferentes modos, para distintos usos e de diversos fabricantes diferentes. No mercado a uma extensa lista de *Shields* para *Arduino*, podemos deparar com *Shields* que permitem realizar diversas funções como: MP3 Player, *Shields* WiFi ou ethernet para conexão com a internet, *Shields* que simulam funções de celulares, *Shields* de controle de motores, telas de LCD, *Shields* para cartões SD entre outros. A Figura 18 mostra alguns *Shields* existentes no mercado.



**Figura 18 – Motor Shield, SD Shield, Ethernet Shield e Wave Shield**

Normalmente os fabricantes de *Shields* proporcionam bibliotecas com conteúdo para facilitar a programação, deste modo, em muitas ocasiões não é necessário fazer um estudo muito detalhado para aprender a executar funções particulares de cada *Shield*.

### 3.6 - HARDWARE E SOFTWARE OPEN-SOURCE

A plataforma *Arduino* em si é bastante útil para projetos de microcontroladores, mas só isso não é suficiente para impulsionar a popularidade e a ampla adoção da plataforma. Em vez de fechar o desenho da placa de interface e ambiente de desenvolvimento, todo o projeto *Arduino* está profundamente enraizado na prática emergente de *hardware open-source*, onde os objetos físicos são o resultado e a prática de *software open-source*, onde o Linux é geralmente o exemplo frequentemente citado (EVANS, 2011).

O *Arduino* é composto por *hardware* e *software open-source*. Os *designs* (esquemas e arquivos CAD) de referência do *hardware* são distribuídos sob a licença *Creative Commons Attribution Share-Alike* e estão disponíveis no site do *Arduino*, os arquivos de layout e produção para algumas versões do *hardware* também estão disponíveis. O código fonte da *IDE Arduino* também está disponível e liberado sob a GNU GPL (*General Public License*) e disponíveis no site do *Arduino*.

Ele envolve um modelo distribuído de desenvolvimento de *hardware* com os contribuintes em geral, residentes em diferentes partes do mundo. Em vez de sistemas fechados, projetos de código aberto permitem uma liberdade individual para acessar os arquivos de origem de um projeto, fazer melhorias, e redistribuir essas melhorias para uma maior comunidade.

O ecossistema *Arduino* fundamentalmente encarna essa aspiração de abertura em design, arquitetura, colaboração e filosofia. Pois todos os arquivos do projeto, esquemas e softwares estão disponíveis gratuitamente para download, usar, modificar, refazer, e até mesmo revender. A prática de contribuintes que têm a liberdade de usar projetos livremente e sem obrigação de comprar qualquer coisa ajuda a tornar o *Arduino* uma plataforma muito agradável e de fácil aceitação (EVANS, 2011).

Apenas o nome “*Arduino*” está guardado sobre uma licença proprietária para que seja exclusivo para o produto oficial, e não seja usado para trabalhos de terceiros sem autorização.



### 3.7 – COMUNIDADE ARDUINO

Talvez embora mais importante do que a própria plataforma de *hardware* e *software*, a comunidade *Arduino* é provavelmente a única grande engrenagem na máquina *Arduino*. Esta comunidade de fabricantes tem colaborado para o *Arduino* com códigos de desenvolvimento e bibliotecas, projetando novos *hardwares*, disponibilizando oficinas pedagógicas e classes, e compartilhando o que eles inventaram. O *Arduino* já está sendo ensinado em escolas, faculdades e universidades de todos os lugares, a “Noite Arduino” é um ritual regular em qualquer uma das séries de *hackerspaces* em todo o mundo. As pessoas que se unem para desenvolver alguma ideia em volta do *Arduino* contribuem para esta comunidade global, que fez o *Arduino* se tornar o sucesso que é hoje (EVANS, 2011).

O envolvimento da comunidade para um projeto como o *Arduino* é essencial para o sucesso ou fracasso do projeto. O *Arduino* também ajudou a difundir o movimento *DIY* uma abreviação de *Do It Yourself* (do inglês faça você mesmo) que é você adquirir os esquemas de um projeto e você mesmo modifica-los para suas necessidades e construir projetos adaptados para sua necessidade e também distribuir na comunidade esses projetos. Cada vez mais novos criadores entram para a comunidade, tornando-a muito mais ativa e rica em conhecimento.

## 4 - TECNOLOGIA GOOGLE ANDROID

*Android* é a primeira plataforma de *software* para dispositivos móveis de código aberto, que inclui um sistema operacional baseado no *kernel* do Linux, *middleware*, aplicações básicas para o usuário e ferramentas para a criação de aplicativos como o SDK e o ADT. Desenvolvida pela Google e posteriormente mantida pela *Open Handset Alliance*. Ele permite aos desenvolvedores escrever código utilizando a linguagem de programação Java e executam sobre a máquina virtual customizada para dispositivos moveis chamada *Dalvik* (OHA, 2013), (ABLESON; COLLINS; SEN, 2009) e (LECHETA, 2009).

A inauguração da plataforma *Android* em 05 de novembro de 2007 foi anunciado com a fundação da *Open Handset Alliance*, um grupo de empresas de TI e de telecomunicações dedicados ao avanço de padrões abertos para dispositivos móveis. A ideia *Android* está resumida em: “*Android* - A imaginação é o limite”.

Uma das características mais interessantes e atraentes do *Android* é que, devido à sua arquitetura, os aplicativos de terceiros, incluindo aqueles que são “desenvolvidos em casa”, são executado com a mesma prioridade como aqueles que são distribuídos junto com o núcleo do sistema, todos os aplicativos são tratados pelo sistema sem distinção de prioridade. (OHA, 2013)

Além do generoso SDK e as bibliotecas bem formadas que estão disponíveis para os desenvolvedores *Android*, a característica mais interessante é que agora se tem acesso a tudo o que o sistema operacional tem acesso. Em outras palavras, se você quiser criar um aplicativo que disca o telefone, você tem acesso ao discador do telefone, se você quiser criar um aplicativo que utiliza o GPS interno do telefone (se equipado), você tem acesso a ele. O potencial para que os desenvolvedores criem aplicações dinâmicas e intrigantes, está aberta com a plataforma *Android*. (ANDROID DEVELOPERS, 2013).

A notícia de seu lançamento causou um grande impacto e superou as expectativas de vendas da empresa, e desde então a plataforma se tornou um sucesso, em aparelhos de diversos fabricantes, chegando a corresponder 68,4% do mercado de *smartphones* em 2012 (MARTINS, 2013).

Os dispositivos móveis estão cada vez mais se tornando uma realidade na vida das pessoas, pois em um mesmo aparelho, temos funções que vão desde a um simples telefonema e recebimento de mensagens até processamento de grandes quantidades de informação, funções de GPS, acesso à Internet e demais funções, existindo milhares de aplicativos para todas essas funcionalidades.

Com constantes atualizações do *Android*, para torna-lo melhor, surgiram diversas versões, cada uma com um nome de uma sobremesa ou bolo (em inglês) como mostrado na Tabela 4 fornecendo dados sobre o número relativo de dispositivos que executam uma determinada versão da plataforma *Android* coletados pelo *Android Developer* em junho de 2013. (ANDROID DEVELOPERS, 2013).

Versão	Codiname	API	Distribuição
1.5	Cupcake	3	-
1.6	Donut	4	0.1%
2.1	Eclair	7	1.5%
2.2	Froyo	8	3.2%
2.3 - 2.3.2	Gingerbread	9	0.1%
2.3.3 - 2.3.7		10	36.4%
3.2	Honeycomb	13	0.1%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	25.6%
4.1.x	Jelly Bean	16	29.0%
4.2.x		17	4.0%

**Tabela 4 - Versões do Android (Android Developers, 2013)**

Até o momento, as versões de distribuição são: 1.5 *Cupcake* lançado em Abril de 2009, 1.6 *Donut* lançado em Setembro de 2009, 2.0 e 2.1 *Eclair* lançado em Outubro de 2009, 2.2 *FroYo* lançado em Maio de 2010, 2.3 *Gingerbread* lançado em Dezembro de 2010, 3.0/3.2 *Honeycomb* lançado em Fevereiro de 2011 (Apenas

para tablets), 4.0 *Ice Cream Sandwich* lançado em outubro de 2011 e 4.1 e 4.2 *Jelly Bean* lançado em junho de 2012.

#### 4.1 - OPEN HANDSET ALLIANCE

A *Open Handset Alliance* é um grupo de 84 empresas de TI e tecnologia móveis, liderados pelo *Google* que se uniram para acelerar a inovação em consumidores móveis e oferecer uma experiência móvel mais rica, mais barata e melhor. O grupo desenvolveu e aprimorou a primeira completa, aberta e livre plataforma móvel o *Android*, e está empenhada em implantar comercialmente aparelhos e serviços que utilizam a plataforma *Android*. Entre alguns integrantes do grupo, estão nomes consagrados como *HTC*, *LG*, *Motorola*, *Samsung*, *Sony Ericsson*, *Toshiba*, *Sprint Nextel*, *China Mobile*, *T-Mobile*, *ASUS*, *Intel*, *Garmin* e muitos mais (FARIA, 2008) e (OHA, 2013).

A visão que a *Open Handset Alliance* compartilha é de mudar a experiência móvel para os consumidores, assim desenvolveram uma plataforma aberta e se empenham em melhorar a transparência do setor móvel para que permitam a todos os envolvidos inovar mais rapidamente e responder as demandas dos consumidores em relação dispositivos móveis com a plataforma *Android* (OHA, 2013).

Assim, o escopo do grupo é determinar uma plataforma única e aberta para celulares, permitindo assim mais satisfação dos consumidores com o produto final. Outro objetivo dessa aliança foi desenvolver uma plataforma moderna e flexível para o desenvolvimento de aplicações corporativas. O resultado dessa união de empresas foi à criação do *Android* (LECHETA, 2009) e (OHA, 2013).

O mundo da tecnologia está sempre em evolução, e a OHA foi empregada precisamente para manter uma plataforma-padrão em que todas as novas tendências do mercado permaneçam englobadas em uma única solução (LECHETA, 2009).

O fato de o *Android* ser de código aberto colabora muito para seu aprimoramento, uma vez que desenvolvedores de todos os lugares do mundo podem cooperar para

melhorar seu código-fonte, acrescentando novas funcionalidades ou simplesmente corrigindo falhas (ABLESON; COLLINS; SEN, 2009).

#### 4.2 - DESENVOLVIMENTO DE APLICATIVOS ANDROID

É possível desenvolver uma aplicação para o *Android*, utilizando a linguagem de programação *Java* em um dos ambientes de desenvolvimento disponíveis, como o *Eclipse*, *Netbeans* ou *Android Studio* nova plataforma de desenvolvimento para *Android* baseado no *IntelliJ IDEA*. (LECHETA, 2009) e (ANDROID DEVELOPERS, 2013).

O *Eclipse* é o ambiente de desenvolvimento preferido pelo *Google*, e há um *plug-in* chamado *Android Development Tools* (ADT) para facilitar o desenvolvimento, os testes e a compilação do projeto, e o *Android SDK software* adotado para desenvolver aplicações no *Android* e um emulador mostrado na Figura 19 que simula um celular (LECHETA, 2009).



**Figura 19 – Emulador Android**

O SDK possui um emulador para simular o celular, ferramentas utilitárias e uma API completa para a linguagem *Java*, com todas as classes necessárias para desenvolver as aplicações. É possível executar o emulador do *Android* diretamente no *Eclipse* usando o *plug-in* ADT que visa integrar o ambiente de desenvolvimento *Java* com o emulador e assim possibilitando desfrutar de todos os seus recursos. (DIMARZIO, 2008) e (LECHETA, 2009).

Deste modo é muito fácil e pratico desenvolver aplicações para *Android*, basta ter um conhecimento básico em *Java* para iniciar o desenvolvimento e criação de aplicações com diversas funcionalidades.

#### 4.3 - Máquina Virtual Dalvik

*Dalvik* é o nome da máquina virtual usada pelo *Android*, baseada em registradores, desenhada e escrita por “Dan Bornstein” e alguns outros engenheiros do *Google*. Nela encontramos uma grande diferença em relação à máquina virtual *Java* (JVM), uma vez que a máquina virtual do *Google* não está baseado em uma pilha.

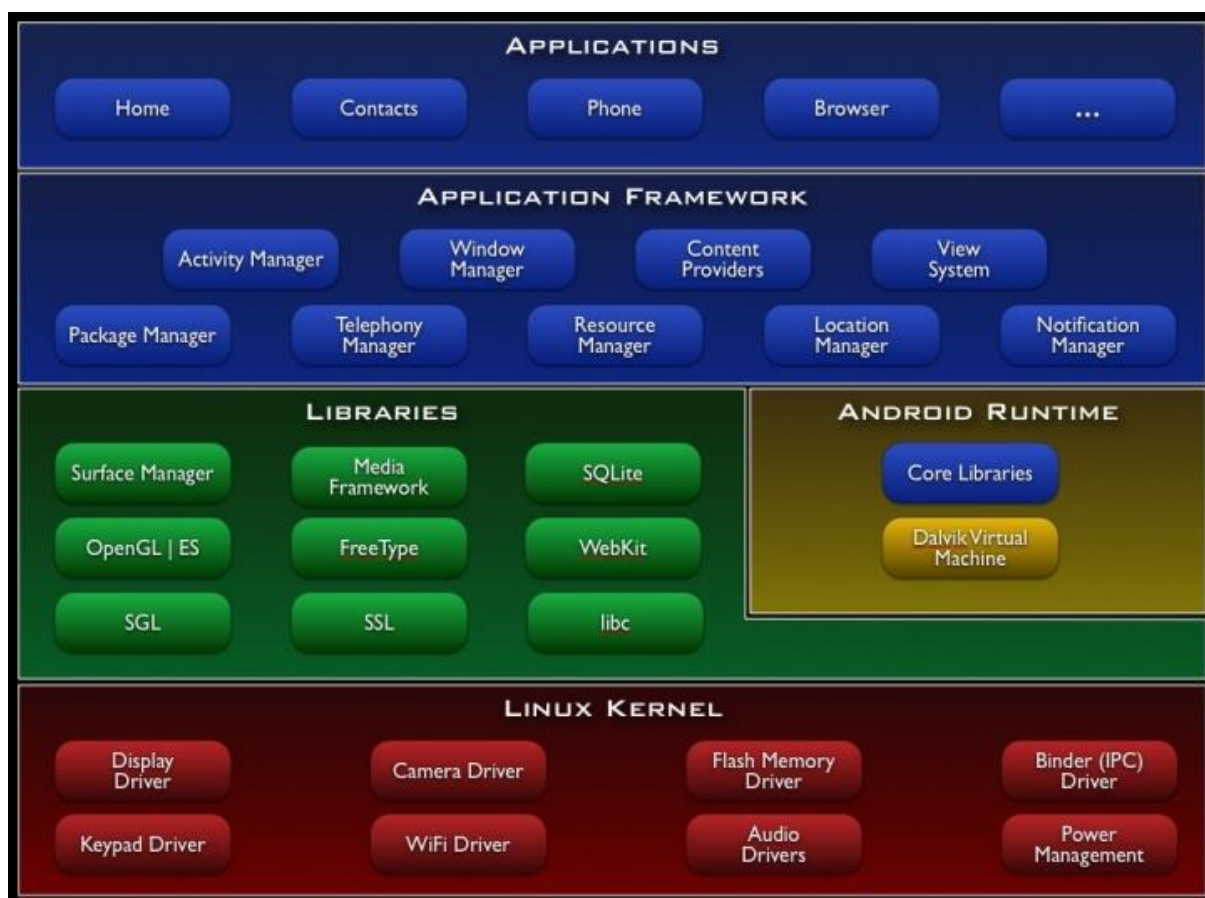
A máquina virtual *Dalvik* é um interpretador que roda apenas os arquivos executáveis de formato *.dex* (*Dalvik Executable*). Este formato é otimizado para rodar múltiplas instâncias com utilização eficiente e baixa de memória, a qual consegue delegando ao *kernel* a gestão dos threads (*multithreading*), de memória e de processos.

Ao desenvolver as aplicações para o *Android*, é possível empregar a linguagem *Java* e todos os seus recursos normalmente, porém, o *bytecode* (*.class*) é compilado e convertido para o formato *.dex* (*Dalvik Executable*), representando a aplicação do *Android* compilada (ABLESON; COLLINS; SEN, 2009), (DIMARZIO, 2008) e (LECHETA, 2009).

Após esse processo, os arquivos *.dex* e outros recursos, como imagens, são compactados em um único arquivo com a extensão *.apk* (*Android Package File*), representando a aplicação final, pronta para ser distribuída e instalada (ABLESON; COLLINS; SEN, 2009) e (LECHETA, 2009).

#### 4.4 - ARQUITETURA ANDROID

Como já mencionado, o *Android* é uma plataforma móvel que contém um sistema operacional, middleware e aplicações básicas para o usuário. Será feita uma descrição geral da arquitetura em camadas, mostrada na Figura 20, desta plataforma usada no *Android*.



**Figura 20 – Camadas da Arquitetura Android (ANDROID DEVELOPERS, 2013)**

Cada uma dessas camadas usa serviços oferecidos pela camada acima, que por sua vez, oferecem suas próprias camadas para níveis mais elevados, contribuindo para o desenvolvimento de aplicativos mais eficientes.

#### 4.4.1 – Linux Kernel

Na base temos o *kernel* 2.6 do *Linux*, utilizado pela sua robustez comprovada e implementação das funções básicas para qualquer sistema operacional, como exemplo: segurança, gerenciamento de memória e processos, implementação de conectividade de rede e vários intérpretes (*drivers*) para comunicação com dispositivos físicos (*hardware*). O *kernel* também atua como uma camada de abstração entre o *hardware* e o resto do *software*, exercendo a funcionalidade de *middleware* entre as camadas da plataforma *Google Android*

O *Android* utiliza como base o *kernel* do *Linux*, mas os dois sistemas não são os mesmos, além das funcionalidades já implementadas no *kernel* do *Linux*, o *Android* adiciona algumas coisas específicas para plataformas móveis, como a comunicação entre processos, uma forma de lidar com a memória compartilhada e gerenciamento de energia.

Sempre que um fabricante inclui um novo componente de hardware, a primeira coisa a ser feita para que ele possa ser usado a partir do *Android* é criar bibliotecas de controle ou os drivers necessários dentro do núcleo *Linux* embarcado no *Android* (MURPHY, 2009) e (CATALÁN, 2011).

#### 4.4.2 – Libraries e Android Runtime

Sobre o *kernel*, temos um conjunto de bibliotecas C e C++ usado pelo sistema para diversos fins, como controle de display (*Surface Manager*), bitmaps e fontes (*Free Type*), gráficos 2D e 3D (SGL e *OpenGL*), gerenciamento de mídia (*Media Framework*), armazenamento de dados (*SQLite*) e motor de renderização para navegar na internet (*WebKit*).

Junto com essas bibliotecas, podemos encontrar tudo o que é necessário para executar os aplicativos através da máquina virtual *Dalvik*. Cada aplicativo usa uma instância da máquina virtual executando um arquivo DEX (*Dalvik Executable*) e o sistema está otimizado para rodar múltiplas instâncias da máquina virtual. É desenvolvido em Java, mas não utiliza uma máquina virtual Java (JVM) para



execução, nem arquivos do tipo *.class*, os arquivos *.class* são convertidos em arquivos *.dex* e executados pela Máquina Virtual Dalvik (MURPHY, 2009) e (CATALÁN, 2011).

#### 4.4.3 – Application Framework

Sobre as bibliotecas encontra-se uma estrutura que fornece um conjunto de ferramentas para o desenvolvimento de qualquer aplicação, disponibilizando todas as APIs e recursos necessários para os pacotes e aplicativos, este *Framework* permite aos desenvolvedores tirar vantagem de um sistema de classes visuais já construídos, administrar notificações e acessar dados através de provedores de conteúdo, entre outras coisas. (FARIA, 2008), (MURPHY, 2009) e (CATALÁN, 2011).

#### 4.4.4 – Applications

Este é o último nível das camadas da arquitetura *Android* em que, como o próprio nome revela, encontram-se todos os aplicativos do *Android*, como cliente de *e-mail*, navegador *Web*, contatos, entre outros. Todos os projetos desenvolvidos e instalados estarão presentes nessa camada da plataforma. Todas estas aplicações utilizam os serviços, APIs e bibliotecas de níveis anteriores (FARIA, 2008), (MURPHY, 2009) e (CATALÁN, 2011).

### 4.5 - COMPONENTES BÁSICOS DE UM APLICATIVO ANDROID

Uma vez vista a arquitetura, deve-se apresentar os componentes fundamentais para desenvolver um aplicativo *Android*. Os componentes básicos de um aplicativo são: *Activities*, *Intents*, *views*, *services*, *content providers* e *broadcast receivers*.

Nem todos os aplicativos necessitam ter todos os componentes, mas qualquer aplicativo será uma combinação destes.

#### 4.5.1 – Activities

É definitivamente o componente mais comum de aplicativos *Android*. Um componente *Activity* reflete uma determinada atividade exercida por um aplicativo, e que normalmente está associado a uma janela ou uma interface de usuário, é importante notar que não só inclui as ilustrações, mas este componente faz parte da *Activity* por meio de exibições representados por classes, tais como *View* e seus derivados. Esse componente é implementado pela classe com o mesmo nome da *Activity*. A maioria das aplicações permite a execução de várias ações através da existência de uma ou mais telas. Por exemplo, pense em uma aplicação de mensagens de texto. Nela, a lista de contatos é exibida em uma janela. Ao implantar uma segunda janela, o usuário pode gravar a mensagem para o contato selecionado, e em outra terceira pode rever o seu histórico de mensagens enviadas ou recebidas. Cada uma destas janelas deve ser representada por um componente *Activity*, de modo que navegar a partir de uma janela para outra envolve no lançamento de uma atividade para dormir ou outra (MURPHY, 2009) e (CATALÁN, 2011).

*Android* permite controlar todo o ciclo de vida de componentes *Activity*, pois ele se preocupa em iniciar e parar todos esses processos, gerenciar a execução e decidir o que fazer de acordo com os recursos disponíveis e os comandos dados pelo usuário.

Ao projetar um aplicativo *Android* deve-se levar em conta o seu ciclo de vida.

#### 4.5.2 – Intents

Consiste basicamente em uma intenção de realizar alguma ação, geralmente associada a alguns dados. Lançando um *Intent*, um aplicativo pode delegar o trabalho para outro, de modo que o sistema é responsável por encontrar o único aplicativo dentre os instalados no sistema, que pode executar a ação solicitada, Por exemplo, abrir uma URL em um navegador web, ou escrever um e-mail a partir de

um cliente de e-mail. Os *Intents* estão incluídos no *AndroidManifest* porque descrevem de onde e quando podem começar um *Activity*. Quando um *Activity* cria uma *Intent*, ele pode ter a propriedade do que você quer fazer (MURPHY, 2009) e (CATALÁN, 2011).

Uma vez que o aplicativo é executado, o *Android* compara essas informações *Intent* por *Intent* de cada aplicativo, escolhendo o mais adequado para a operação especificada pelo chamador.

#### 4.5.3 - Broadcast Receivers

São componentes usados para lançar qualquer execução dentro do aplicativo atual, quando um determinado evento ocorre (geralmente abrir um componente de *Activity*). Por exemplo, uma chamada ou SMS recebido. Este componente não tem nenhuma interface de usuário associado, mas você pode usar a API *Notification Manager* para notificar o usuário do evento produzido através da barra de status do dispositivo móvel (MURPHY, 2009) e (CATALÁN, 2011).

Esse componente é implementado através de uma classe de nome *Broadcast Receiver*. Para que o *Broadcast Receiver* funcione, não é necessário que o aplicativo em questão seja o aplicativo ativo no momento do evento. O sistema inicia a aplicação, se necessário, quando ocorre o evento monitorado.

#### 4.5.4 – Service

Um componente *Service* representa um aplicativo executado sem interface com o usuário, e que normalmente ocorre em segundo plano enquanto outros aplicativos estão ativos na tela do dispositivo. Um exemplo típico deste componente é um leitor de música. A interface de reprodução mostra ao usuário as diferentes faixas disponíveis, bem como os botões usuais de play, pause e volume. No momento em que o usuário toca uma música, ela é ouvida enquanto ainda mostra todas as ações acima, e pode até mesmo executar um aplicativo diferente, sem que a música pare. A interface de usuário do reprodutor seria um componente *Activity*, mas a música

tocando seria um componente *Service*, porque ele é executado em segundo plano (MURPHY, 2009) e (CATALÁN, 2011).

#### 4.5.5 - Content Provider

Com o componente *Content Provider*, qualquer aplicativo *Android* pode armazenar dados em um arquivo em um banco de dados *SQLite*, ou em qualquer outro formato que considere. Além disso, esses dados podem ser compartilhados entre diferentes aplicativos. Uma classe que implementa o componente *Content Provider* irá conter uma série de métodos para armazenar, recuperar, atualizar e compartilhar dados em um aplicativo (MURPHY, 2009) e (CATALÁN, 2011).

Há uma coleção de classes para diferentes tipos de gerenciamento de dados no pacote *android.provider*. Além disso, qualquer formatação adicional que você deseja implantar deverá pertencer a este pacote e seguir seus padrões de funcionamento.

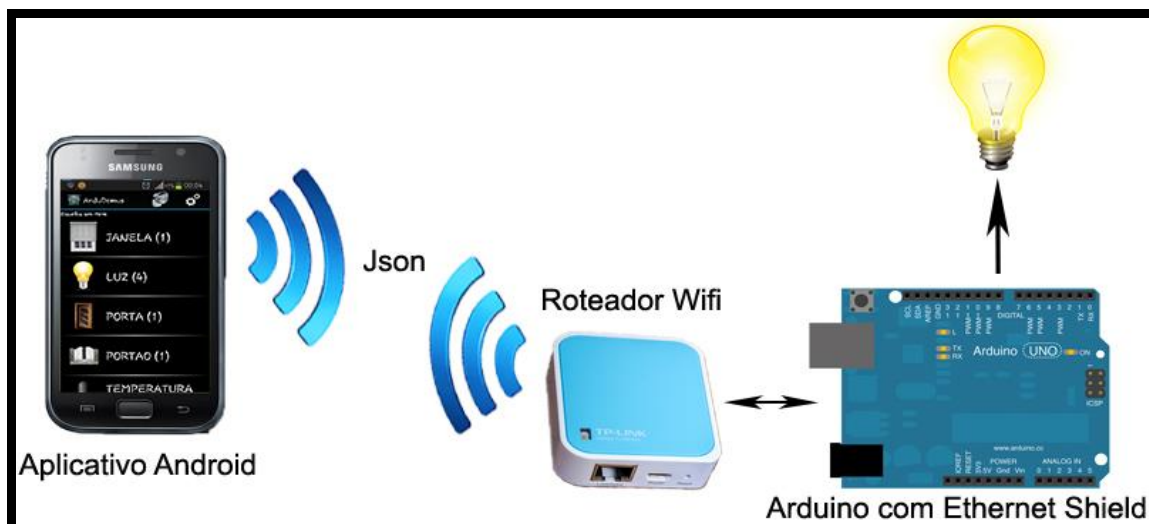
## 5 – PROPOSTA DE TRABALHO

Neste capítulo será apresentada a especificação do modelo proposto neste trabalho. A solução desenvolvida é composta em duas partes: a parte física, que consiste na maquete com o circuito elétrico ligado aos sensores e atuadores e gerenciados pelo programa carregado para o *Arduino*, e a parte móvel (interface), que consiste no aplicativo para *smartphones* e *tablets* com *Android*. Ambas as partes serão detalhadas nos capítulos seguintes.

### 5.1 – DEFINIÇÃO DA PROPOSTA DE TRABALHO

A proposta deste trabalho é de modelar e construir um protótipo miniaturizado de um projeto domótico de baixo custo, alto poder computacional, fácil interatividade e gerenciado por dispositivos móveis como *smartphones* e *tablets*. Para isso, é necessária uma arquitetura condizente com esses objetivos.

Assim sendo, será utilizada na parte física da solução proposta a plataforma de prototipagem *Arduino* como sendo o controlador do sistema domótico, uma vez que, estando muito empregada em soluções domóticas por ser fundamentada no conceito de *software* e *hardware* livres, possuir baixo custo de implementação, ser de fácil utilização, e possuir total compatibilidade com as demais tecnologias domóticas existentes no mercado, como por exemplo, X10 e *ZigBee*. Também serão utilizados alguns sensores como: presença, luminosidade, temperatura, umidade relativa do ar, umidade do solo (para os jardins), e alguns atuadores como: luzes, motores, alarmes sonoros. Para a comunicação com a interface de usuário nos dispositivos móveis será utilizado um *Shield* de *ethernet* no *Arduino* ligado a um roteador *WiFi* para fazer a ponte entre o *Arduino* e o dispositivo móvel. E na parte móvel (interface) será desenvolvido um aplicativo para *Android* que se comunicará com o *Arduino* através de requisições HTTP e esperará como resposta o formato *JSON* por meio do sinal *WiFi* do roteador ligado ao *Arduino*, conforme ilustrado na Figura 21:



**Figura 21 – Arquitetura simplificada das tecnologias utilizadas**

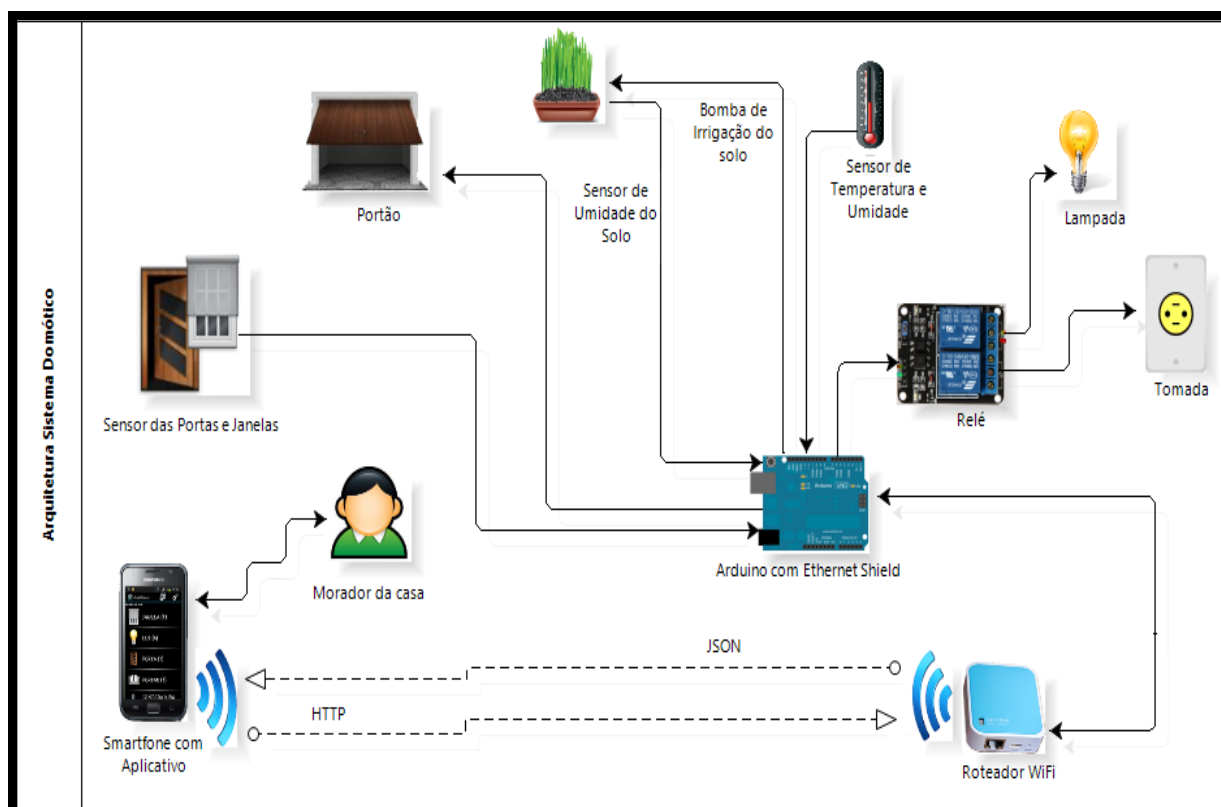
A partir do desenvolvimento e junção da parte física e da parte móvel foi possível a visualização do funcionamento do sistema domótico, nas sessões a seguir será mostrado os componentes utilizados no desenvolvimento do trabalho.

## 5.2 – ARQUITETURA DO PROJETO DOMÓTICO

A modelagem do problema a ser abordado é ilustrada na Figura 22 e representa a arquitetura do projeto domótico com suas entidades e relacionamentos. Esta arquitetura expõe claramente a complexidade do problema abordado neste trabalho.

O modelo arquitetado possui o morador da casa, como entidade usuária do projeto domótico. O morador da casa irá monitorar e interagir com o sistema domótico através de um aplicativo *Mobile*, desenvolvido na plataforma *Google Android*.

O aplicativo desenvolvido fará requisições HTTP para o servidor *Arduino* que vai interpretar cada requisição recebida e executará uma das ações pré-definidas e retornará como resposta a essa requisição um *JSON* com o status da ação requisitada.

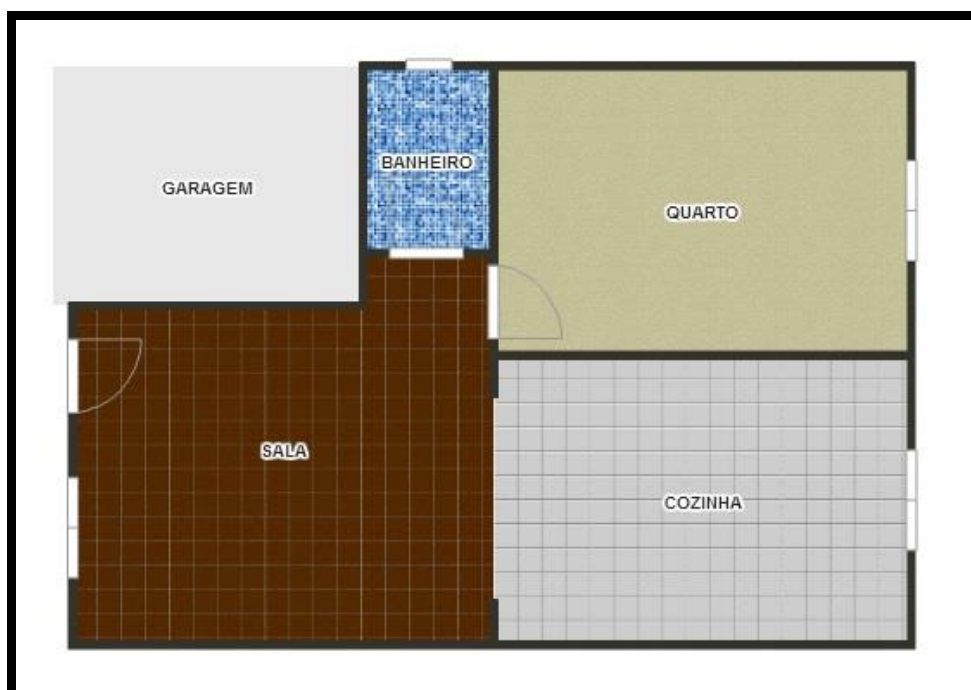


**Figura 22 – Arquitetura Sistema Domótico**

Após o servidor *Arduino* interpretar a requisição e retornar o status da ação no formato *JSON* o aplicativo irá interpretar o *JSON* recebido e mostrar as informações retornadas na tela do *smartphone* para o morador da casa, que tomará as decisões de ligar uma lâmpada, um aparelho conectado a uma tomada ou acionar a bomba para irrigar o jardim.

### 5.3 – PLANTA DA MAQUETE

Para a realização do protótipo domótico foi necessária à construção de uma maquete de uma casa a fim de representar seus respectivos cômodos que serão automatizados pelo sistema domótico. Primeiramente foi desenvolvida para a representação uma planta baixa da residência a fim de auxiliar na criação da maquete a Figura 23 contém a planta baixa da casa e demonstra como seus cômodos foram distribuídos.



**Figura 23 – Planta baixa da maquete**

Os cômodos da maquete estão distribuídos da seguinte forma: na frente temos a sala e a garagem com um banheiro ao lado e no fundo temos a cozinha e o quarto. A partir da planta baixa da casa foi possível uma melhor visualização para a construção da maquete que é detalhada no capítulo seguinte.

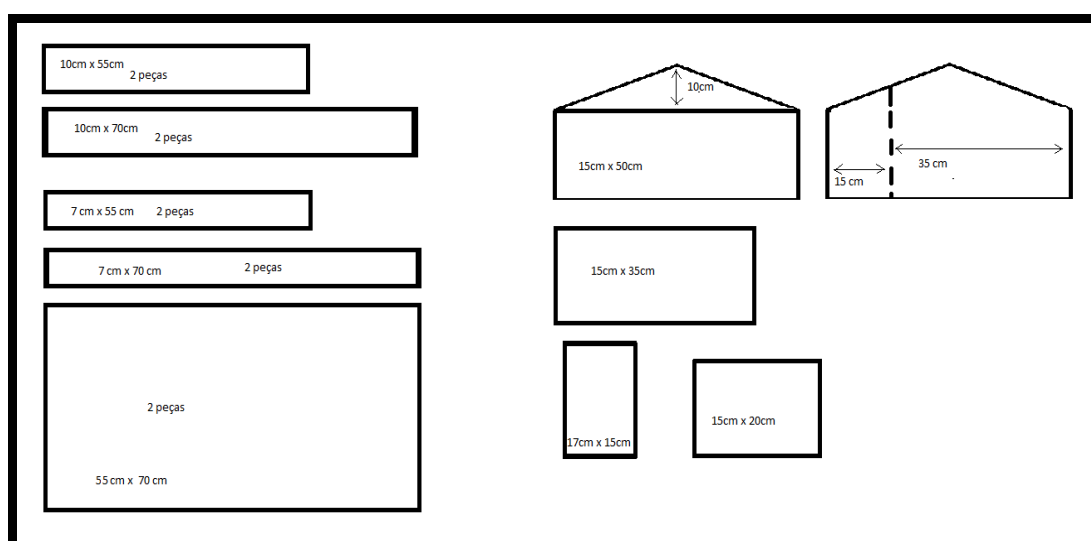


## 6 - ESTUDO DE CASO

Neste capítulo serão apresentadas as implementações do modelo proposto neste trabalho e detalhados os componentes utilizados na parte física do projeto domótico e o aplicativo desenvolvido com a plataforma *Google Android* para a parte móvel (interface) do projeto.

### 6.1 – CONSTRUÇÃO DA MAQUETE

Para o início da construção da maquete foi desenhado um esboço como o mostrado na Figura 24 com as medidas para o corte da chapa de MDF, para depois montar a base e as paredes e divisórias dos cômodos da maquete.



**Figura 24 – Esboço das medidas para o corte da madeira**

Para a construção da maquete foi utilizado uma chapa de madeira MDF, palitos de sorvete para o telhado e tinta para a pintura. Depois de desenhados os esboços e feitos os cortes na chapa de MDF, foi utilizado cola de artesanato para a montagem da maquete. A Figura 25 demonstra a construção da maquete até seu estágio final.



**Figura 25 – Maquete do Protótipo Domótico**

Em cada cômodo da maquete foram implementadas as funcionalidades do sistema domótico para demonstrar o funcionamento do projeto domótico.

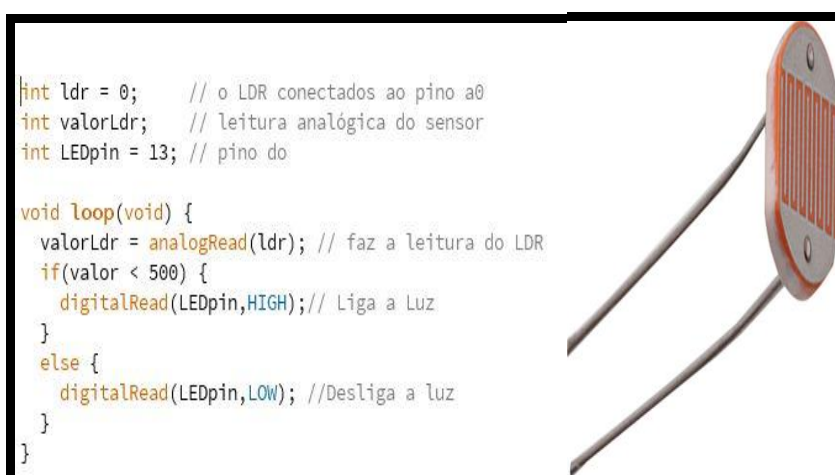
## 6.2 – PARTE FÍSICA DO PROJETO

O protótipo domótico desenvolvido consiste no monitoramento da temperatura ambiente, umidade relativa do ar e estado da janela e porta (aberta ou fechada), juntamente com o acionamento das luzes da residência e abertura e fechamento do portão de entrada.

No projeto domótico o LDR foi utilizado para verificar a luminosidade do exterior da casa e de acordo com a luminosidade recebida ele manda esse sinal para o *Arduino* que liga a luz da garagem se estiver escurecendo ou desliga quando estiver amanhecendo e a luz solar voltar a iluminar o sensor LDR.

O LDR, Resistor Dependente de Luz, mostrado na Figura 26 (às vezes chamado de fotoresistor), é um tipo de resistor que varia sua resistência à partir da luminosidade captada.

O LDR é constituído de cádmio, um material semicondutor, que é disposto na superfície do componente. Esse material tem a característica de diminuir sua resistência quando a luminosidade sobre ele aumenta. Já quando está escuro ou a luminosidade decai, a sua resistência é aumentada.

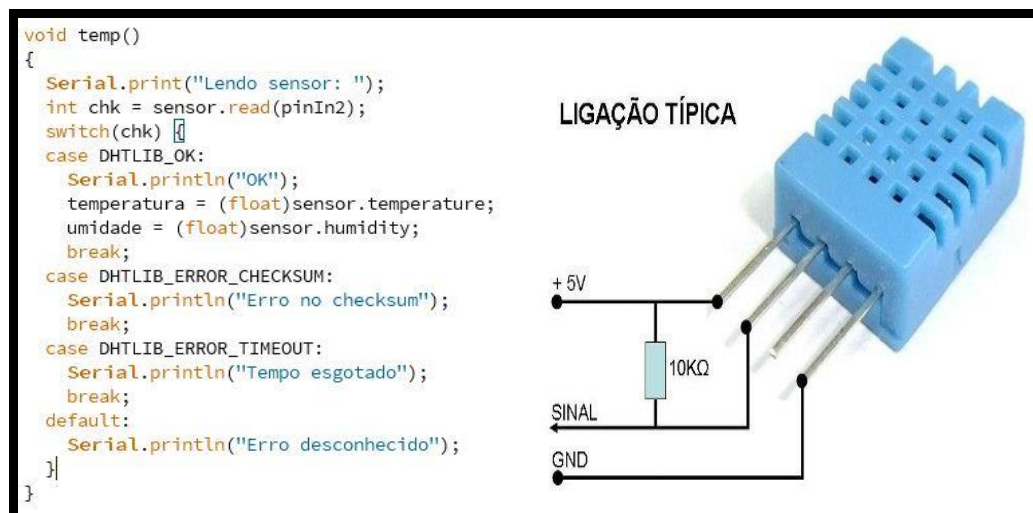


**Figura 26 – LDR**

O LDR é muito utilizado nas chamadas fotocélulas que controlam o acendimento de poste de iluminação e luzes em prédios. (MCROBERTS, 2009). Na Figura 26 é mostrado uma parte do código para o monitoramento do LDR.

Para o monitoramento da temperatura ambiente e umidade relativa do ar foi utilizado o sensor DHT11, mostrado na Figura 27, juntamente com trecho do código para a leitura do sensor, este sensor inclui um componente de medição de umidade do tipo

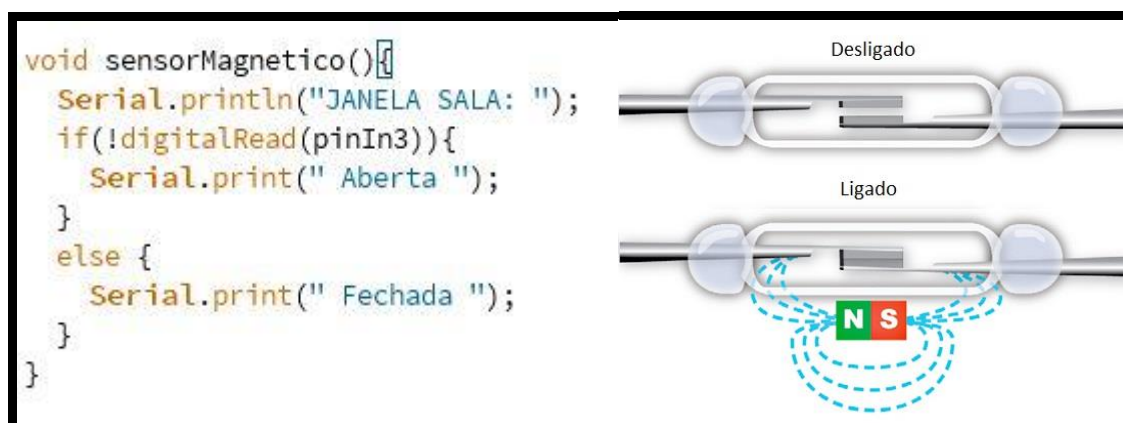
resistivo e um componente de medição de temperatura NTC, ele é básico e de baixo custo.



**Figura 27 – Sensor DHT11**

Este sensor é bastante simples de usar, mas requer uma espera para pegar os dados. A única desvantagem real deste sensor é que você só pode obter novos dados uma vez a cada 2 segundos (SUNROM TECHNOLOGIES, 2013).

No monitoramento das janelas foram utilizados sensores magnéticos (*reed-switch*), mostrado na Figura 28, juntamente com o código utilizado para monitorar o sensor.

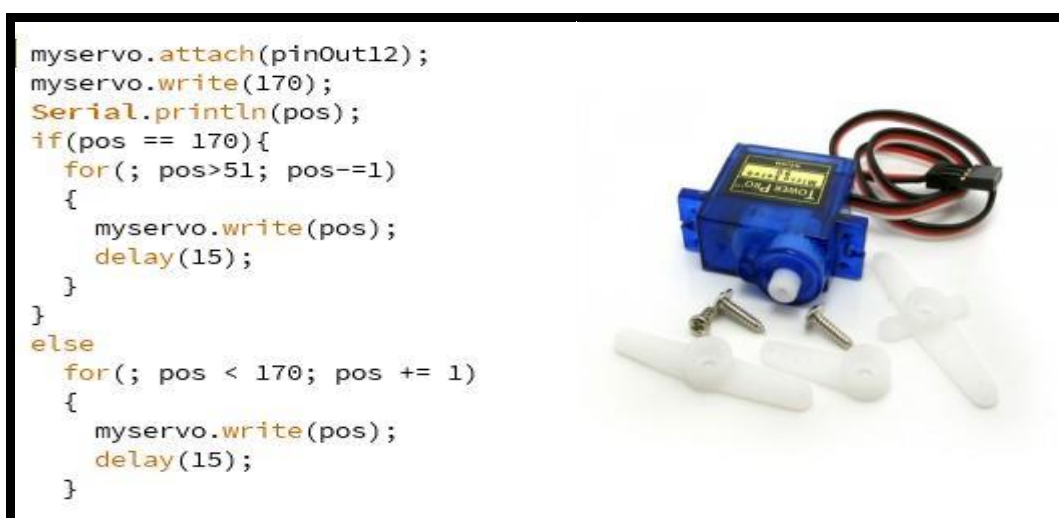


**Figura 28 – Sensor Reed Switch (sensor magnético)**

O *reed-switch* é composto de uma cápsula de vidro e de duas lâminas de um material ferromagnético (ligas de níquel e ferro). As duas lâminas são colocadas muito próximas, sem que haja contato entre elas, com uma extremidade afixada no vidro e mergulhadas num gás inerte, para não sofrerem oxidação ou deformação mecânica (para durarem mais).

Para acionar o *reed-switch*, isto é, para haver contato elétrico entre as lâminas, é necessário induzir a magnetização delas, fazendo com que elas se atraiam magneticamente. Basta aproximar um pequeno ímã do *reed-switch* (BRAGA, 2013).

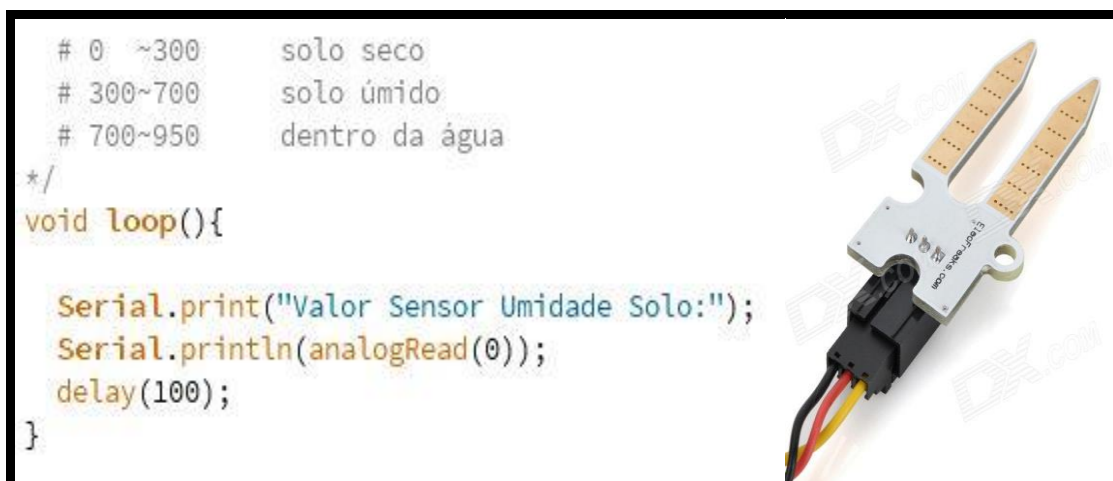
Para abrir e fechar o portão foi utilizado um servomotor, como mostrado na Figura 29, e um trecho do código para o seu funcionamento, o servomotor se posiciona de acordo com o sinal recebido pelo *Arduino* com ângulo de 0 a 180 graus.



**Figura 29 – Servomotor**

Servomotor é um atuador rotativo com alta precisão de controle para posicionamento angular, que apresenta movimento proporcional a um comando, em vez de girar ou se mover livremente sem um controle mais efetivo de posição como a maioria dos motores; servomotores são dispositivos de malha fechada, ou seja: recebem um sinal de controle; verificam a posição atual; atuam no sistema indo para a posição angular do motor de forma precisa (KOLLMORGEN, 2013).

Para monitorar a umidade do solo do jardim foi utilizado um sensor de umidade de solo, mostrado na Figura 30, juntamente com seu código para o monitoramento do sensor.



**Figura 30 – Sensor Umidade Solo**

O sensor de umidade do solo pode ler a quantidade de umidade presente no solo ao seu redor. É um sensor de baixa tecnologia, mas ideal para monitoramento de um jardim urbano, ou o nível de água de uma planta. Este sensor é necessário para manter seu jardim conectado ao sistema.

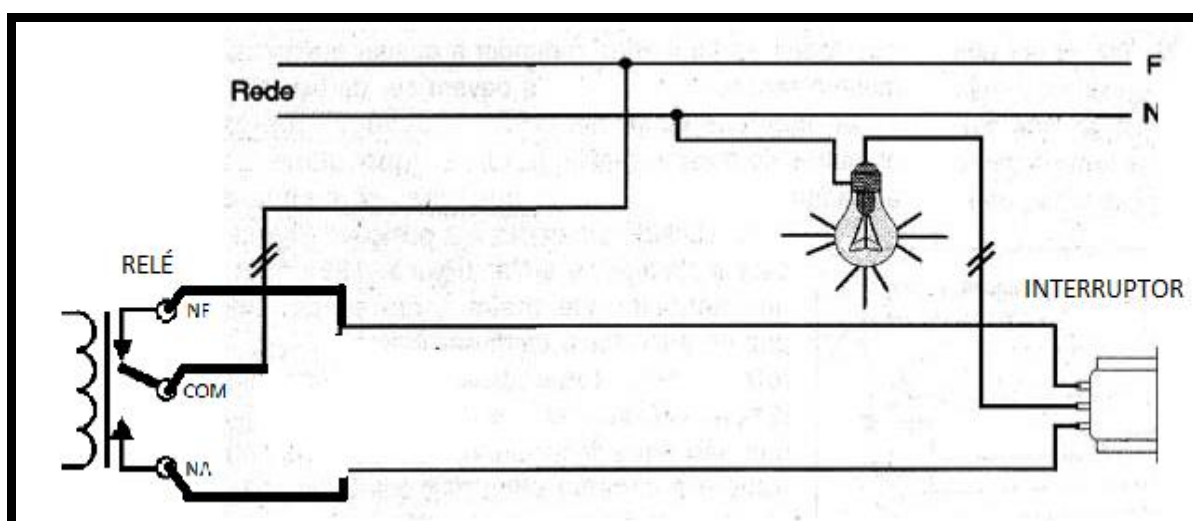
O sensor usa as duas sondas para passar a corrente através do solo, e, em seguida, lê-se essa resistência para obter o nível de umidade. Quanto mais água o solo torna-se mais fácil conduzir eletricidade (menos resistência), enquanto que o solo seco conduz menos eletricidade (mais resistência). Este sensor é útil para lembrá-lo de molhar suas plantas em casa ou para monitorar a umidade do solo em seu jardim (ELECTFREAKS, 2013).

Para molhar o jardim foi utilizado uma mini bomba de água, mostrada na Figura 31, com o monitoramento da umidade do solo o morador pode tomar a decisão de ligar a bomba de água para regar o jardim.



**Figura 31 – Mini bomba de água**

No acionamento das lâmpadas foi utilizado um módulo de relé ligado em paralelo a um interruptor, como na Figura 32, e para as tomadas foi utilizado apenas o módulo de relé para o seu acionamento.



**Figura 32 – Ligação paralelo relé e interruptor**

O módulo de relé pode controlar dispositivos elétricos de alta voltagem (máximo de 250V). Ele pode ser usado em projetos interativos e também pode ser utilizado para controlar a iluminação, equipamentos elétricos como válvulas solenoides, lâmpadas, motores e outros dispositivos de alta corrente ou alta tensão. O módulo pode ser



controlado diretamente por uma vasta gama de microcontroladores por uma porta digital.

Todos esses sensores e atuadores são gerenciados pelo *Arduino* com um *Shield Ethernet* ligado a um roteador *WiFi*, como mostrado na Figura 33, o *Arduino* se comporta como um servidor que interpreta os comandos recebidos, para controlar e gerenciar os sensores e atuadores ligados a ele.



**Figura 33 – Shield Ethernet com roteador WiFi**

O *Shield Ethernet* permite que uma placa *Arduino* se conecte à internet. Ele baseia-se no chip *Wiznet* de *ethernet*. O *Wiznet W5100* fornece uma rede (IP) com suporte a TCP e UDP. Ele suporta até quatro conexões de soquete simultâneos. Usa-se a biblioteca *Ethernet* para escrever os programas no *Arduino* para que ele possa se conectar à internet (ARDUINO, 2013).

Conectando ao *Shield Ethernet* um roteador *WiFi* pode-se estabelecer conexão com qualquer dispositivo ligado à rede local, e enviar todas as informações vindas do *Arduino* via *WiFi* para qualquer um desses dispositivos, como por exemplo o *smartphone* ou *tablet* se conecta à rede através do *WiFi* e assim comunica-se com o *Arduino* mandando e recebendo informações. Deve-se configurar o roteador ligado ao *Shield Ethernet* para que ele encaminhe os pacotes recebidos para o IP e porta



do *Arduino* configurada no *software* desenvolvido, habilitando a passagem dos pacotes recebidos pelo roteador para chegarem até o *Arduino*, no IP e porta especificada.

Para a comunicação e gerenciamento dos dispositivos ligados ao *Arduino* com o aplicativo desenvolvido, foi preciso mapear cada dispositivo ligado a ele no código do *software* desenvolvido para gerenciar os dispositivos e se comunicar com o aplicativo, para cada dispositivo foi relacionada uma porta do *Arduino* como entrada (sensores) ou saída (atuadores) de acordo com o tipo do dispositivo a ser gerenciado.

### 6.3 – PARTE MÓVEL (INTERFACE) DO PROJETO

Primeiramente foi desenvolvido uma página *HTML*, como mostrada na Figura 34, onde o usuário visualiza e controla de forma interativa a situação atual dos equipamentos para gerenciar o projeto domótico, servindo de interface entre o usuário e a residência utilizando o *Arduino* como servidor da página *HTML* desenvolvida.



**Figura 34 – Interface para o usuário (página html)**

A interface que gerencia os equipamentos da residência foi desenvolvida para testar todas as funcionalidades do projeto antes de partir para a implementação do aplicativo desenvolvido com a plataforma *Google Android*.

Clicando nos botões quarto, cozinha e sala é ligada ou apagada as luzes desses cômodos, clicando no botão portão é aberto ou fechado o portão de entrada e o monitoramento da temperatura ambiente, umidade do ar e estado da janela são mostrados abaixo dos botões.

Para possibilitar a comunicação entre o *Arduino* e o dispositivo móvel com *Android*, foi desenvolvido, um aplicativo desenvolvido com a plataforma *Google Android* chamado “ArduDomus”, comunicando via *HTTP* com o *Arduino* e recebendo os resultados no formato *JSON*, e um *software* para o *Arduino* cujo objetivo é interpretar os comandos e enviar as informações para o aplicativo ArduDomus.

### **6.3.1 - ArduDomus**

Para gerenciar e controlar os dispositivos mapeados no *Arduino* de forma fácil e flexível, possibilitando mobilidade e iteratividade na comunicação entre a parte física e a parte móvel, foi desenvolvido um aplicativo para dispositivos móveis com a plataforma *Google Android* chamado ArduDomus, mostrado na Figura 35.

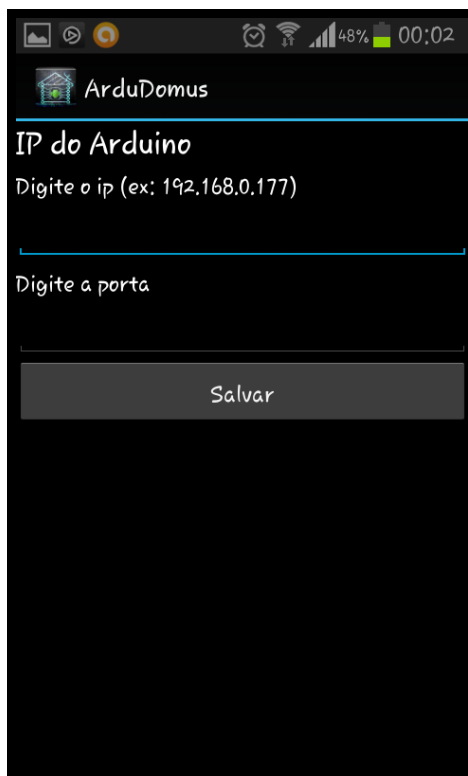


**Figura 35 – ArduDomus**

O aplicativo ArduDomus desenvolvido para este trabalho, tem como objetivo gerenciar o sistema domótico, enviando e recebendo informações para a plataforma *Arduino*, através de uma conexão *WiFi*, assim interpretando e mostrando essas informações para o usuário, o aplicativo ArduDomus envia seus comandos e recebe as respostas vindas do *Arduino*, configurando no aplicativo o IP e a porta para que o mesmo possa localizar na rede local o *Arduino*, esses comandos são interpretados pelo *software* desenvolvido e carregado para o *Arduino* e com isso permite o gerenciamento dos dispositivos mapeados previamente no mesmo e o retorno de suas respostas.

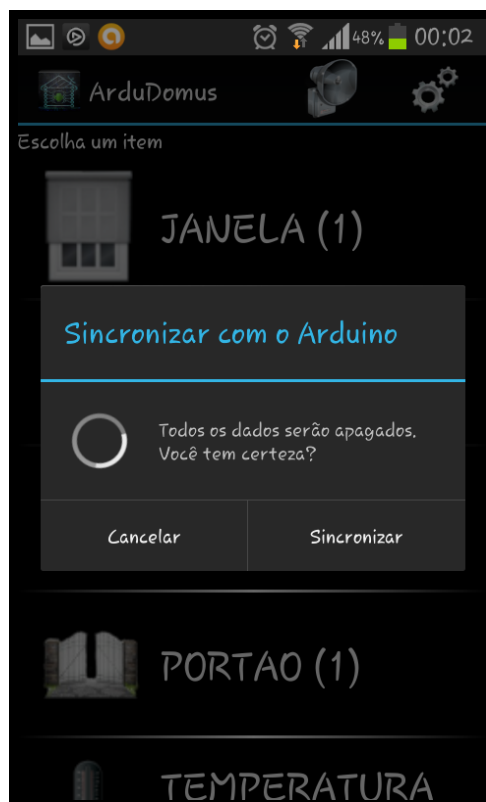
Após a primeira conexão do ArduDomus e sincronização com o *Arduino*, ele grava no banco de dados *SQLite* do *Android* todas as informações recebidas sobre todos dispositivos mapeados previamente no *Arduino*, como por exemplo: tipo do dispositivo, nome do dispositivo, pino em que o dispositivo está conectado, status do dispositivo e seu valor. Além disso, o ArduDomus grava com o *Shared Preferences* as informações de configuração do *Shield Ethernet* do *Arduino*, passadas pelo

morador e usuário do sistema domótico, como IP e porta de comunicação, mostrada na Figura 36, informações estas que são utilizadas no processo de sincronização do aplicativo com o *Arduino* e no processo de envio e recebimento de informações geradas pelo sistema domótico.



**Figura 36 – Configuração do ArduDomus**

Ao iniciar o aplicativo pela primeira vez, o usuário deve fazer a sincronização do ArduDomus com o *Arduino*, mostrada na Figura 37, que irá receber uma lista de todos os dispositivos mapeados previamente no *Arduino* no formato *JSON* e irá grava-los no banco de dados *SQLite* do *Android*, e de forma dinamicamente mostrá-los na interface do aplicativo. Desta forma, o ArduDomus pode gerenciar cada dispositivo mapeado facilmente sem a necessidade de inserir manualmente um por um. Caso sejam acrescentados mais dispositivos ao *Arduino*, será preciso fazer uma nova sincronização para que o ArduDomus reconheça esses novos dispositivos e os grave no banco de dados.



**Figura 37 – Sincronização do ArduDomus**

Depois de realizado o processo de sincronização, o ArduDomus já está apto para gerenciar o sistema domótico e pode começar a enviar e receber informações do *Arduino*. Uma lista contendo todos os tipos de dispositivos previamente mapeados no *software* desenvolvido e carregado para o *Arduino* é exibida na tela principal, desta forma, selecionando cada tipo de dispositivo aparecerá os dispositivos correspondentes a aquele tipo, tornando-se possível interagir com os mesmos através de um simples clique no dispositivo que o usuário quer controlar e selecionar uma das opções para ligar ou desligar os dispositivos atuadores ou apenas monitorar os dispositivos sensores, neste caso, o ArduDomus apresenta uma tela apropriada para cada tipo de dispositivo, possibilitando um controle fácil, iterativo e adequado para cada dispositivo utilizado que pode ser gerenciado.

O ArduDomus consegue enviar e receber informações de forma apropriada comunicando com o *Arduino* muito facilmente. Toda a troca de informações entre o aplicativo ArduDomus e o *Arduino* é feita através da classe *URLConnection* que cria e abre uma conexão *HTTP* entre o aplicativo e o *Arduino* aonde é possível

enviar requisições *GET* e receber uma resposta a partir do fluxo de entrada retornado pelo *InputStream*. Deste modo, o aplicativo ArduDomus se comunica com o *Arduino* enviando e recebendo informações, de uma forma fácil e simples.

No primeiro acesso do usuário ao fazer a sincronização com o *Arduino*, o ArduDomus envia um comando *GET* para o IP e porta do *Arduino* configurado no aplicativo com a mensagem *"/?out=all"*. Em seguida, o *Arduino* identifica essa requisição e retorna para o ArduDomus a resposta a essa requisição em formato *JSON* contendo todos os dispositivos e suas respectivas configurações como: pino de conexão, tipo de dispositivo, status do dispositivo e valor, assim o ArduDomus recebe esse *JSON* e o interpreta com a biblioteca *GSON*, que o divide em objetos do tipo dispositivo e os insere no banco de dados *SQLite*, trecho da implementação que interpreta o *JSON* recebido pelo *Arduino*:

```
public static ArrayList<Dispositivo> getTodosDispositivos(String ip) {
    ListaDispositivos arrayDispositivo = new ListaDispositivos();
    ArrayList<Dispositivo> disp = new ArrayList<Dispositivo>();
    try {
        String url_login = "http://" + ip + "/?out=all";
        Gson gson = new GsonBuilder().setVersion(1.0).create();
        String result = getHttp(url_login);
        if (result != null) {
            Type collectionType = new TypeToken<ListaDispositivos>()
            {}.getType();
            arrayDispositivo = gson.fromJson(result, collectionType);
            disp = arrayDispositivo.getDisp();
            for (int i = 0; i < disp.size(); i++) {
                disp.get(i).setCriado_em(new Date());
                disp.get(i).setIp(ip);
                disp.get(i).setStatus("0");
                disp.get(i).setValor("0");
            }
        }
    } catch (JsonIOException e) {
        Log.e(LOG_TAG, "Erro conversão do Json", e);
    }
    return disp;
}
```

Dessa forma, o ArduDomus poderá gerenciar e informar ao *Arduino* quais os dispositivos devem ser acionados. Ou seja, o ArduDomus manda uma requisição ao *Arduino* informando que ligue ou desligue o dispositivo Luz que está plugado ao pino "9", o *Arduino* liga esse dispositivo e retorna uma informação ao ArduDomus com o

status e valor do dispositivo, que mostra esse status ou valor na interface para o usuário.

O *Arduino*, por meio do software carregado e desenvolvido para este trabalho, recebe as informações vinda do ArduDomus interpreta e executa cada comando e responde cada requisição com informações no formato *JSON*. Desta forma, o ArduDomus persiste os dados vindos do *Arduino* e realiza de forma fácil e precisa sua comunicação, além de proporcionar um ambiente seguro, o ArduDomus evita erros humanos na criação dos dispositivos no aplicativo, pois cada dispositivo é criado dinamicamente na sincronização do ArduDomus com o *Arduino*, que prove uma forma simples e prática para a comunicação e envio de comandos para os dispositivos gerenciados em uma residência, permitindo mobilidade e flexibilidade para o projeto domótico.

Ao concluir a etapa de implementação do ArduDomus, foi possível testar e validar todas suas funcionalidades juntamente com a parte física do *Arduino* e os dispositivos ligados a ele.

Após os testes realizados foi possível analisar as principais vantagens e desvantagens do projeto domótico proposto. Começando pela forma de acesso, a solução proposta é de o acesso ser feito via *WiFi* através de um roteador em uma rede local, O sistema só funcionará dentro do alcance do roteador ligado ao *Arduino*, e na implementação do *software* para o gerenciamento dos dispositivos ligados ao *Arduino* e comunicação com o ArduDomus, o mapeamento dos dispositivos deve ser feito de forma correta para que o sistema domótico proposto execute suas funções da forma esperada.

No entanto, a solução apresentou resultados condizentes com os esperados, além de possibilitar uma grande flexibilidade em relação ao acesso e controle dos dispositivos, permitindo também mobilidade e um controle preciso dos recursos da residência.

## 7 – CONCLUSÃO

Em função do projeto desenvolvido, que teve como finalidade o objetivo de modelar e construir um protótipo miniaturizado de um projeto domótico de baixo custo, alto poder computacional, fácil interatividade e gerenciado por dispositivos móveis como *smartphones* e *tablets*, foi possível verificar que o objetivo principal do mesmo foi atingido em sua plenitude.

O projeto domótico se mostrou flexível, robusto e de fácil interatividade, proporcionando ao morador e usuário do sistema a opção de sincronizar o ArduDomus com o *Arduino* e assim criar os dispositivos dinamicamente no aplicativo, evitando erros por parte do morador e usuário. Assim sendo, qualquer pessoa pode utilizar o ArduDomus para gerenciar e controlar sua residência de forma fácil e iterativa de qualquer parte da residência, de uma maneira segura.

Toda a arquitetura e as tecnologias empregadas neste projeto domótico são *open source*, portanto, é possível que o morador e usuário do sistema domótico possa alterar e melhorar da forma que achar necessário qualquer componente de forma a deixar o projeto mais condizente com a sua realidade e aplicação própria.

Deste modo, a solução proposta para o projeto domótico foi condizente com os objetivos propostos, além de proporcionar uma solução de menor custo tornando-se mais acessível, permitindo que boa parte da população de baixa renda possa ter acesso a esse tipo de sistema.

### 7.1 – TRABALHOS FUTUROS

A partir deste trabalho pode-se desenvolver diversas implementações para o setor domótico, adicionando novas funcionalidades e aperfeiçoando o projeto. Pode-se adicionar comandos por voz, para que o usuário e morador da residência com um simples comandos de voz possa gerencia-la, adicionar novos tipos de sensores e



atuadores para uma melhor adequação com as diversas necessidades de cada tipo de usuário.

Em trabalho futuros, tal solução pode ser aperfeiçoada, modificando sua arquitetura para suportar novos dispositivos, implementando uma forma de segurança no acesso ao servidor *Arduino* como uma autenticação, habilitar acesso via internet ao servidor *Arduino*, assim permitindo ao aplicativo *ArduDomus* conectado à internet de qualquer lugar do mundo controlar e gerenciar o projeto domótico.

Por fim, é possível dar continuidade na pesquisa, com referência neste trabalho, podendo desenvolver vários outros tipos de projetos domóticos que irá auxiliar diversas pessoas em suas residências proporcionando aos seus moradores maior conforto, comodidade e segurança, que é o que todos buscam em suas residências.

## REFERÊNCIAS

ABLESON, W. Frank; COLLINS, Charlie; SEN, Robi. **Unlocking Android – A Developer's Guide**. 1 ed. Greenwich: Manning, 2009.

ALLAN, Alasdair. **iOS Sensor Apps with Arduino**. 1 ed. Sebastopol,: O'Reilly Media, 2011.

ANDROID DEVELOPERS. **Android**. Disponível em:  
<<http://www.developer.android.com.html>>. Acesso em: Junho de 2013.

ARDUINO. **Arduino Shields**. Disponível em:  
<<http://www.arduino.cc/en/Main/ArduinoShields>>. Acesso em: Março de 2013.

ARDUINO. **Arduino**. Disponível em: <<http://www.arduino.cc>>. Acesso em: Março de 2013.

ARAUJO, Jair J.; PEREIRA, Carlos E. Análise de Protocolos de Automação Predial/Residencial. In: **15th Congresso Brasileiro de Automática**, 2004, Gramado. 15th Congresso Brasileiro de Automática, 2004.

AURESIDE. **Brasileiros têm grande interesse em Automação Residencial, diz estudo**. Disponível em:  
<[http://www.aureside.org.br/noticias\\_recentes/default.asp?file=01.asp&id=332](http://www.aureside.org.br/noticias_recentes/default.asp?file=01.asp&id=332)>. Acesso em: Março de 2013.

AURESIDE (Brasil). **MISSÃO**. Disponível em: <<http://www.aureside.org.br/quemsomos/default.asp?file=missao.asp&menu=quemsomos>>. Acesso em: Maio 2013.

BANZI, Massimo. **Primeiros passos com o Arduino**. 1. Ed. São Paulo: Novatec, 2012.

BOLZANI, Caio Augustus Moraes. **Residências Inteligentes: Domótica; Redes Domésticas; Automação Residencial**. 1. ed. São Paulo: Livraria da Física, 2004.

BOLZANI, Caio Augustus Moraes. **Análise de Arquiteturas e Desenvolvimento de uma Plataforma para Residências Inteligentes**. 155 p. Tese (Doutorado) - Departamento de Engenharia de Sistemas Eletrônicos, Escola Politécnica da Universidade de São Paulo, São Paulo, 2010.

BRAGA, Newton C. **Como funciona o Reed-Switches (MEC089)**. Disponível em: <<http://www.newtoncbraga.com.br/index.php/como-funciona/3860-mec089>>. Acesso em: Julho 2013.

CALLONI, Juan Carlos, **Curso Básico de Domótica**. 1. ed. Buenos Aires: Libreria y Editorial Alsina, 2011.

CASTILLO, Juan Carlos Martin, **Instalaciones Domóticas**. Madri: Editex, 2009.

CATALÁN, Adrián. **Curso Android: Desarrollo de aplicaciones móviles**. 1 ed. Maestros del Web, 2011.

CEDOM, Asociacion Espanola de Domotica, **Instalaciones Domóticas - Cuaderno de buenas prácticas para promotores y constructores**. Espanha: Aenor, 2008.

CHAMUSCA, Alexandre. **Domótica e Segurança Electrónica: A inteligência que se instala**. Portugal: Ingenium Edições, 2006.

CIEC, Colegio de Ingenieros Especialistas de Córdoba, Guía de Contenidos Mínimos para La Elaboración de un Proyecto de Domótica. In: **Domótica Registrada**. Comisión de Domótica, Córdoba, 2011. 34 p.

CUARTIELLES, David. **Team Arduino**. Disponível em: <<http://www.flickr.com/photos/dcuartielles/2363657276/in/photostream/>> Acesso em: junho 2013.

DIMARZIO, Jerome F. **Android – A Programmer's Guide**. 1. ed. New York City: McGraw-Hill, 2008.

DOMÍNGUEZ, Hugo Martín; VACAS, Fernando Sáez. **Domótica: Un enfoque sociotécnico**. 1 ed. Madri: Fundación Rogelio Segovia Para El Desarrollo de Las Telecomunicaciones, 2006.

ELECFREAKS. **Octopus Soil Moisture Sensor Brick**. Disponível em: <<http://www.electfreaks.com/store/octopus-soil-moisture-sensor-brick-p-422.html>>. Acesso em: Outubro 2013.

EVANS, Brian. **Beginning Arduino Programming - Writing Code for the Most Popular Microcontroller Board in the World**. 1. Ed. New York City: Apress, 2011.

FARIA, Alessandro de Oliveira. **Programe seu Andróide**. Linux Magazine, Volume 1, Número 43, p. 73-77, 2008.

FARIAS, Norma; BUCHALLA, Cassia Maria. A classificação internacional de funcionalidade, incapacidade e saúde da organização mundial da saúde: conceitos, usos e perspectivas. **Revista Brasileira de Epidemiologia**, v.8, n.2, junho,2005,187–193.

GLOBAL CODE. **Program-ME**. Disponível em: <<http://www.eletronlivre.com.br/program-me/>>. Acesso em: Junho 2013.

KELLY, James Floyd; TIMMIS, Harold. **Arduino Adventures: Escape from Gemini Station**. 1 ed. New York City: Apress, 2013.

KOLLMORGEN. **Servomotores**. Disponível em: <<http://www.kollmorgen.com/pt-br/products/motors/servo/servomotores/>>. Acesso em: Outubro 2013.

HUIDOBRO, José Manuel; NOVEL, Beatriz; CALAFAT, Cristhian; SULLER, Eduardo; ESCUDERO, Adrian Nogales; TOLEDANO, José Carlos; SANTAMARÍA, Asunción; LASTRES, Carmen. **La Domótica Como Solución De Futuro**. Madri: Fenercom, 2007.

LECHETA, Ricardo R. **Google Android – Aprenda a criar aplicações para dispositivos móveis com o Android SDK**. 1. ed. São Paulo: Novatec, 2009.

MACEDO, Glayson Paraíso. **DOMÓTICA: UMA ANÁLISE DE PROTOCOLOS DE AUTOMAÇÃO PREDIAL/RESIDENCIAL**. 43 p. Dissertação (Graduação) - Curso de Engenharia De Controle E Automação, Universidade Federal De Ouro Preto Escola De Minas, Ouro Preto, 2009.

MARTINS, Felipe. **Android deteve 70% do mercado mundial de smartphones em 2012**. Disponível em: <<http://mobilexpert.com.br/mercado-telecom/materias/2198/android-deteve-70-do-mercado-mundial-de-smartphones-em-2012>>. Acesso em: Março de 2013.

MEYER, Gordon. **Smarth Home Hacks**. Tips & Tools for Automating Your House. Sebastopol: O'Reilly Média, 2008.

MCROBERTS, Michael. **Arduino básico**. 1. ed. São Paulo: Novatec, 2011.

MCROBERTS, Mike. **Arduino Starters Kit Manual - A Complete Beginners guide to the Arduino**. 1 ed. Earthshine Design, 2009.

MURPHY, Mark L. **The Busy Coder's Guide to Android Development**, 2 ed. Estados Unidos: CommonsWare, 2009.

OHA. **Open Handset Alliance**. Disponível em: <<http://www.openhandsetalliance.com/>>. Acesso em: Maio. 2013.

PEQUENAS EMPRESAS GRANDES NEGÓCIOS (Brasil). **Empresa fatura R\$ 1 milhão com serviços de automação residencial**. Disponível em: <<http://g1.globo.com/economia/pme/noticia/2013/05/empresa-fatura-r-1-milhao-com-servicos-de-automacao-residencial.html>>. Acesso em: Maio 2013.

RILEY, Mike; **Programming Your Home - Automate with Arduino, Android, and Your Computer**. Pragmatic Bookshelf, 2012.

SCHMIDT, Maik. **Arduino - A Quick-Start Guide**. 1 ed. Estados Unidos: Pragmatic Bookshelf, 2011.

SILVEIRA, Paulo Rogério da; SANTOS, Winderson E. dos. **Automação e Controle Discreto**. 9. ed. São Paulo: Érica, 2009.

SUNROM TECHNOLOGIES. **DHT11 - Humidity and Temperature Sensor**. Disponível em: <<http://robocraft.ru/files/datasheet/DHT11.pdf>> Acesso em: Julho 2013.