LAR INTELIGENTE ATRAVÉS DE MICROCONTROLADORES E DISPOSITIVOS ANDROID

Gustavo Paulo Medeiros da Silva[1], Marcelo Maia Sobral[2], Emerson Ribeiro de Mello[3]
[1]Instituto Federal de Santa Catarina/gustavomedeirosdasilva@gmail.com
[2]Instituto Federal de Santa Catarina/msobral@ifsc.edu.br
[3]Instituto Federal de Santa Catarina/mello@ifsc.edu.br

Resumo: *Desde a década de 70 a automação residencial vem ganhando espaço na área de pesquisa e desenvolvimento, tendo se intensificado recentemente devido a avanços dos padrões de redes sem fio, bem como a popularização de telefones inteligentes (smartphones) e tablets. Entre 2010 e 2011 no Instituto Federal de Santa Catarina campus São José foi desenvolvido um protótipo de um sistema de automação residencial chamado DroidLar. Esse sistema possibilitou que dispositivos com acesso a Internet e que usam o sistema operacional Android, tais como smartphones e tablets, monitorassem e controlassem dispositivos eletroeletrônicos residenciais através de uma rede Zigbee. Um componente fundamental do DroidLar é o SAR (Servidor de Automação Residencial), que controla os dispositivos residenciais e faz a interface entre a rede residencial e os dispositivos Android. Neste trabalho o SAR foi redesenhado para poder ser embarcado em um sistema de baixo custo, eliminando assim a necessidade de um computador para essa finalidade.*

Palavras-Chave: *Automação Residencial, Android, ZigBee, Arduino, Sistema Embarcado*

1 INTRODUÇÃO

O projeto de pesquisa *DroidLar - Automação residencial através de um celular Android* (EUZEBIO e MELLO, 2011), desenvolvido entre 2010 e 2011 no IFSC campus São José, apresentou um protótipo para automação residencial combinando microcontroladores (kits Arduino), redes sem fio *ZigBee*, um software para telefone celular *Android*, responsável por controlar tais dispositivos, e um Servidor de Automação Residencial (SAR), representado por um PC x86, que além de agregar a lógica de controle dos dispositivos, serviu de ponte entre a rede sem fio *ZigBee* e a rede TCP/IP, usada por dispositivos *Android*.

O projeto *Lar inteligente através de microcontroladores e dispositivos Android* é um projeto de continuação do sistema DroidLar. O objetivo é melhorar o sistema tornando-o mais econômico e viabilizar a sua implementação em uma residência. O projeto foi dividido em duas fases: (1) tratar os defeitos do cliente *Android* e traduzi-lo para o inglês; (2) desenvolver um novo Servidor de Automação Residencial (SAR) para colocá-lo em um sistema embarcado.

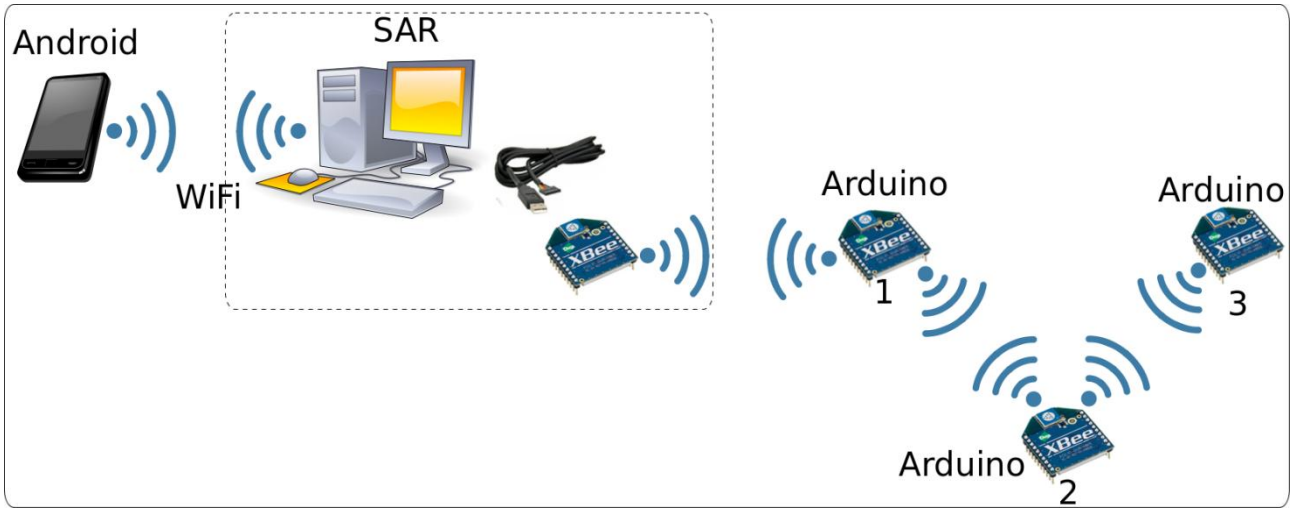
Esse artigo apresenta as melhorias realizadas no sistema DroidLar. Ele está dividido da seguinte forma: a seção 2 apresenta o sistema de automação residencial DroidLar; a seção 3 aborda as melhorias realizadas no cliente *Android*; a seção 4 descreve o desenvolvimento de um novo Servidor de Automação residencial (SAR).

2 SISTEMA DROIDLAR

O DroidLar (EUZEBIO e MELLO, 2011) é um sistema de automação residencial desenvolvido no Instituto Federal de Educação, Ciência e Tecnologia de Santa Catarina campus São José. Este projeto utiliza tecnologias de código e plataformas abertas. O DroidLar possibilita que um aplicativo rodando em dispositivos com *Android* controle dispositivos eletroeletrônicos de uma residência.

O sistema DroidLar é formado por três partes, mostradas na figura 1: i) cliente Android, com que um usuário pode monitorar e controlar os dispositivos residenciais, ii) o Servidor de Automação Residencial (SAR), que se comunica diretamente com os dispositivos residenciais e faz a interface entre eles e os clientes Android, e iii) os controladores e dispositivos baseados em Arduino.

Figura 01 – Elementos do DroidLar



Fonte: EUZEBIO e MELLO, 2011

2.1 Cliente *Android*

Android (OPEN HANDSET ALLIANCE, 2008) é um sistema operacional desenvolvido para dispositivos móveis inteligentes que roda sobre o núcleo Linux. Ele foi inicialmente desenvolvido pela Google e depois pela *Open Handset Alliance*, mas a Google é a responsável pela gerência do produto e engenharia de processos. O motivo da sua criação é devido às dificuldades de desenvolver uma aplicação para diferentes dispositivos pois os fabricantes possuíam seus próprios sistemas operacionais.

A Aplicação DroidLar foi desenvolvida para dispositivos com o sistema *Android*. Essa aplicação é responsável pela comunicação do usuário com o SAR. A comunicação é feita utilizando o protocolo HTTP ou HTTPS em uma conexão de rede IP. O HTTP e o

HTTPS utilizam o modelo de pedido e resposta, ou seja, a comunicação é estabelecida quando um cliente quer enviar uma requisição e é finalizada quando houver uma resposta, por isso a aplicação DroidLar não precisa está conectado ao SAR o tempo todo.

Figura 2 – Interface principal e de configuração do SAR

(a) Tela inicial (b) Informações sobre o servidor

Fonte: EUZEBIO e MELLO, 2011

2.2 Controladores de dispositivos

Os controladores de dispositivos têm a função de executar os comandos enviados pelas interfaces de controle nos eletroeletrônicos da residência, tais como, acender e apagar lâmpadas, fechar cortinas e etc. Normalmente, esses dispositivos são constituídos de microcontroladores e realizam ações de acordo com o comando recebido.

Para o papel de controladores de dispositivos no sistema DroidLar, foram utilizados kits Arduino[12]. Arduino é uma plataforma aberta desenvolvida em 2005 pela *Interaction Design Institute Ivrea*, é muito utilizada para desenvolvimento de protótipos por ser uma ferramenta simples e fácil de usar.

Os kits Arduino possuem duas funções: (1) executar os comandos recebidos no eletroeletrônico conectado a uma de suas portas; (2) enviar ao SAR uma mensagem contendo o estado dos dispositivos que controla, por exemplo, a quantidade de dispositivos, o tipo de dispositivo e se estão ligados ou não. A primeira é executada toda

[12] http://www.arduino.cc

vez que é recebida uma mensagem do SAR com a requisição do comando, já a segunda, é executada quando é recebida uma mensagem de varredura do SAR.

Os kits se comunicam com o SAR usando uma rede ZigBee através de um módulo XBee. O ZigBee foi criado por um grupo de empresas chamada de ZigBee Alliance, tendo sido construído sobre o padrão para LR-WPAN (Low Rate Wireless Personal Area Network) IEEE 802.15.4.

2.3 Servidor de Automação Residencial (SAR)

O servidor de automação residencial (SAR) é o servidor central do sistema DroidLar. Sua função é gerenciar os controladores de dispositivos e fazer a ponte entre o cliente *Android*, que está na rede IP, e os controladores de dispositivos, que estão na rede ZigBee. O SAR deve estar sempre ativo, pois, além de gerenciar as mensagens trocadas entre os membros da rede, também é responsável por executar tarefas agendadas pelo usuário, tais como acender ou apagar lâmpadas em instantes determinados.

O SAR foi originalmente escrito em Java, sendo executado sobre uma plataforma composta por um Java *Servlet Container* em um computador pessoal. Desta forma, os clientes Android acessam o SAR através de uma interface web, usando requisições HTTP. Apesar de funcional, essa versão do SAR não se mostrou adequada para ser implantada em um dispositivo de baixo custo e com recursos limitados, como por exemplo um ponto de acesso sem-fios IEEE 802.11.

3 Melhorias no cliente *Android*

Como explicado na sessão 2.1 o cliente *Android* é responsável por fazer a interface do usuário com o sistema. Porém esse aplicativo apresentou alguns problemas quando utilizado. O cliente *Android* travava, ou seja, não respondia as interações do usuário, como na tentativa de conexão com o SAR, caso este não estivesse operando ou se possuísse um endereço IP diferente do cadastrado no cliente Android.

A solução para esse problema foi tornar concorrentes algumas tarefas internas do cliente Android. Para isso, foram usadas *threads*[13] para possibilitar a intervenção do usuário enquanto uma comunicação bloqueante estava em andamento. Assim, enquanto

[13] http://developer.android.com/reference/java/lang/Thread.html

uma *thread* tenta se conectar com o servidor, uma segunda *thread* mostra uma mensagem de tentativa de conexão para o usuário (ver figura 3.a). Caso o servidor não esteja em operação, o cliente *Android* não trava, mas mostra uma mensagem de erro de conexão como mostrado na figura 3.b.

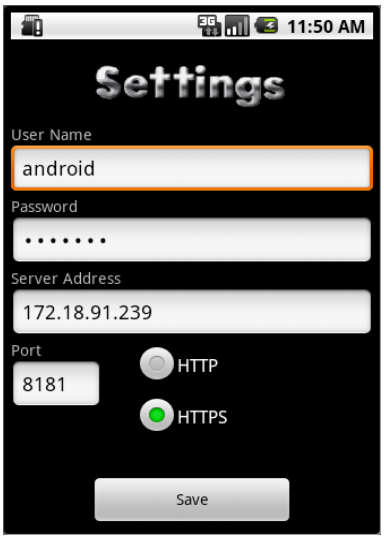
Figura 3 – Mensagens do cliente *Android*.

(a) Tentativa de conexão (b) Erro de conexão

Outra melhoria realizada no cliente *Android* é deixá-lo mais automatizado. Por exemplo, antes o usuário para poder controlar uma lâmpada, primeiro tinha que iniciar o cliente *Android*, depois fazer a tentativa de conexão com o SAR, fazer um pedido de varredura para descobrir quais dispositivos estavam operando, acionar o botão de dispositivos (em formato de lâmpada como mostra do na figura 2.a), fazer um pedido de atualização da lista de dispositivos para visualizar as lâmpadas no aplicativo, e por último controlar a lâmpada. Agora basta o usuário iniciar o aplicativo e acionar o botão de dispositivos que todo o processo de conexão até a atualização de lista é automático.

Também foi realizado a internacionalização do cliente *Android*, ou seja, a tradução do aplicativo para o inglês. Caso o aplicativo detectar o português como idioma padrão no celular, o cliente *Android* é executado em português, se o aplicativo detectar outro idioma o cliente *Android* é executado em inglês.

Figura 4 – Internacionalização do aplicativo



4 Desenvolvimento do novo SAR

Inicialmente o SAR foi escrito em Java para rodar em um computador pessoal. Diferente de outras linguagens que geram um código executável, o Java é compilado para um *bytecode* que é executado por uma máquina virtual. Máquina virtual é um programa que carrega e executa os aplicativos Java, convertendo os *bytecodes* em código executável de máquina.

Para deixar o sistema mais econômico, o SAR foi reprojetado para ser executado em um dispositivo embarcado que use o sistema operacional Linux, como por exemplo *BeagleBone* (BEAGLEBONE, 2012) ou pontos de acesso IEEE 802.11 com firmware Openwrt (*OPENWRT, 2011*). Muitos sistemas embarcados possuem pouca memória e pouco poder de processamento e por esse motivo é necessário escrever programas que utilizem pouco esses recursos. Como o *bytecode* do Java precisa de uma máquina virtual para ser executada, ele utiliza bastante memória e processador em relação aos dispositivos embarcados.

Com a desvantagem do Java para sistemas embarcados foi necessário escolher outra linguagem de programação que dispensa máquinas virtuais ou interpretadores, portanto, uma linguagem que é compilada diretamente para um executável. Por ser popular para programas de sistemas embarcados, a linguagem C foi escolhida para desenvolver o novo SAR.

A primeira parte a ser desenvolvida foi o servidor *WEB*. O novo SAR utiliza a

biblioteca GNU *libmicrohttpd (*GNU LIBMICROHTTPD, 2011) para a implementação de um servidor *WEB*. GNU libmicrohttpd é uma biblioteca C de pequeno porte criado por Christian Grothoff, foi construído para tornar mais fácil a criação de um servidor HTTP como parte de outro aplicativo. O GNU libmicrohttpd é um software livre e parte do projeto GNU. A vantagem de construir um servidor *WEB* integrado ao SAR é a economia de recursos de memória e processador pois ele será exclusivo para o cenário do DroidLar.

Para o SAR se comunicar com os *kits* Arduino, é necessário que haja um módulo *XBee*, assim como nos Arduinos, conectado à porta USB do *BeagleBone*. Esse módulo tem que estar configurado como coordenador da rede *ZigBe*e, sendo o responsável pelas configurações dessa rede.

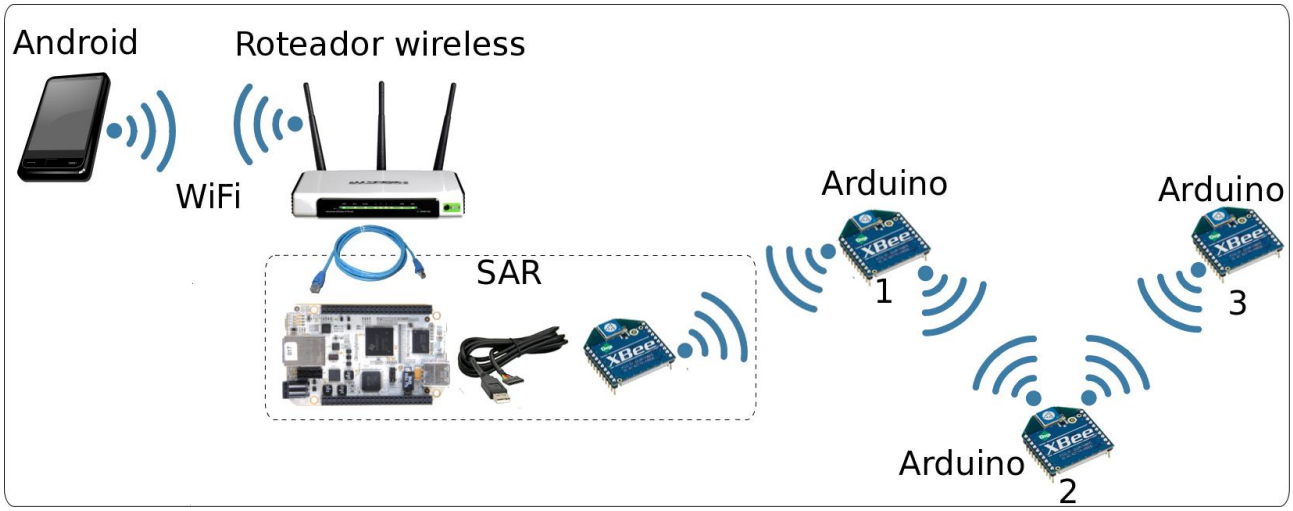
Para o gerenciamento do módulo *XBee* no SAR, foi utilizada a biblioteca *libxbee* (LIBXBEE, 2012). Criada por Attie Grande, o *lixbee* foi desenvolvida para auxiliar os programadores a utilizar os módulos *XBee* com maior facilidade. Segundo o autor é a única biblioteca C desenvolvida para Linux, porém possui documentação escassa.

Para guardar informações, tais como o nome de um dispositivo, foi necessário criar um banco de dados. Existem muitos bancos de dados como o *SQLite* e o famoso *MySQL*, mas optou-se utilizar um banco de dados simples e pequeno, pois o SAR não guardará muitas informações no disco. A biblioteca utilizada para a implementação do banco de dados foi o *GNU dbm* (GDBM, 2011), mais conhecido como GDBM. As informações guardadas serão os nomes dos dispositivos, por exemplo *lâmpada da sala*, listas de agendamentos de dispositivos e os nomes de usuários e suas respectivas senhas.

5 CONSIDERAÇÕES FINAIS

Com os trabalhos realizados no cliente *Android* a usabilidade do aplicativo ser tornou mais prático e fácil, até mesmo para usuários mais leigos. Com o desenvolvimento de um novo servidor de automação residencial o sistema DroidLar chegou ao cenário da figura 5, onde SAR estar embarcado em um *BeagleBone*. O objetivo de embarcar o SAR é para qualquer dispositivo que possui um sistema baseado em Linux e uma porta USB, o *BeagleBone* foi utilizado pois atende essas características.

Figura 5 – Elementos do DroidLar com o SAR embarcado.



AGRADECIMENTOS

Ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) pelo apoio financeiro sem qual não seria possível desenvolver o presente trabalho.

REFERÊNCIAS

BEAGLEBONE. BeagleBoard.org – bone, 2011. Disponível em <http://beagleboard.org/bone>. Acesso em 09 de maio de 2012.

EUZÉBIO, M. V. M.; MELLO, E. R. DroidLar – Automação residencial através de um celular Android, 2011. Disponível em <http://www.sj.ifsc.edu.br/~mello/artigos/droidlar-2011.pdf>. Acesso em 06 de julho de 2012.

GDBM. GNU dbm, 2011. Disponível em <http://www.gnu.org.ua/software/gdbm/>. Acesso em 12 de junho de 2012.

GNU LIBMICROHTTPD. GNU libmicrohttpd: a library for creating an embedded HTTP server, 2011. Disponível em <http://www.gnu.org/software/libmicrohttpd/>. Acesso em 21 de março de 2012.

LIBXBEE. attie.co.uk – libxbee, 2012. Disponível em <http://attie.co.uk/p/projects/software/libxbee>. Acesso em 23 de abril de 2012.

OPEN HANDSET ALLIANCE. Android operating system. 2008. Disponível em: <http://www.android.com>**.**

OPENWRT. OpenWrt Wireless Freedom, 2011. Disponível em <https://openwrt.org/>. Acesso em 17 de janeiro de 2012.

ZIGBEE ALLIANCE. Zigbee specification. ZigBee Document 053474r06, Version, v. 1, 2005.