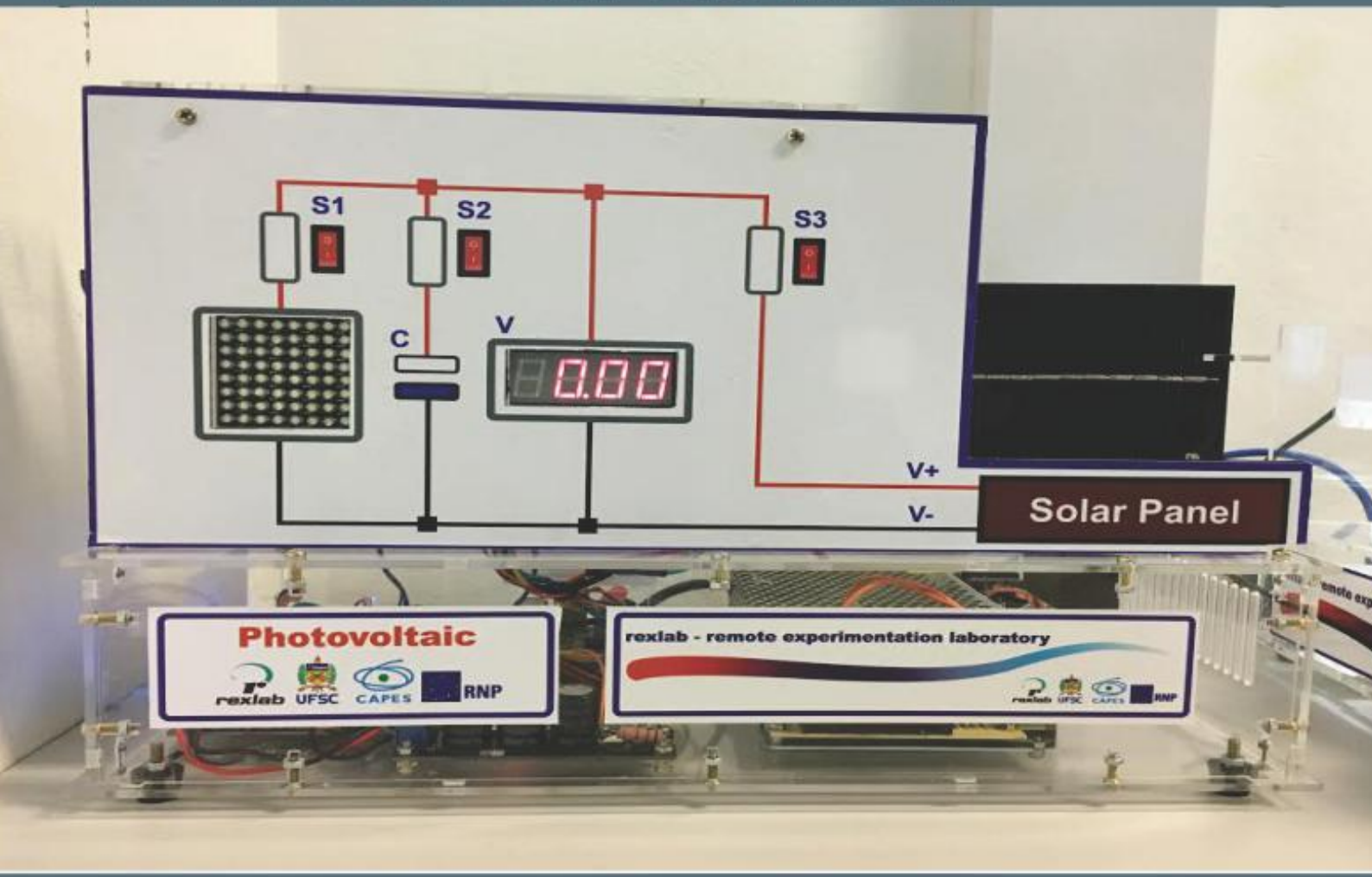
Manual Técnico

Conversão de Energia Luminosa em Elétrica



Experimentação Remota Móvel para o Ensino Básico e Superior

rexlab
LABORATÓRIO DE EXPERIMENTAÇÃO REMOTA

UNIVERSIDADE FEDERAL DE SANTA CATARINA

RNP

CAPES

Manual Técnico do Experimento Remoto Conversão de Energia Luminosa em Elétrica:
Experimentação Remota Móvel para a Educação Básica e Superior
Este guia, cada capítulo e suas imagens estão licenciados sob a licença Creative Commons
Rua Pedro João Pereira, 150, Mato Alto – CEP 88900-000
http://rexlab.ufsc.br/
rexlabufsc@gmail.com

Elaboração
Juarez Bento da Silva
João Paulo Cardoso de Lima
José Pedro Schardosim Simão
Josiel Pereira
Lucas Mellos Carlos
Editoria de arte, projeto gráfico e capa
Isabela Nardi da Silva
Ilustrações
Alex Moretti

Este guia, cada capítulo e suas imagens estão licenciados sob a licença Creative Commons - Atribuição-NãoComercial-Sem Derivados 4.0 Internacional. Uma cópia desta licença pode ser visualizada em http://creativecommons.org.nz/licences/licences-explained/.
Ela define que este manual é livre para reprodução e distribuição, porém sempre deve ser citado o autor. Não deve ser usado para fins comerciais ou financeiros e não é permitido qualquer trabalho derivado.
Se você quiser fazer algum dos itens citados como não permitidos, favor entrar em contato com os organizadores do manual.
O download em edição eletrônica desta obra pode ser encontrado em http://www.rexlab.ufsc.br.

Manual Técnico do Experimento Remoto Conversão de Energia Luminosa em Elétrica :
Experimentação Remota Móvel para a Educação Básica e Superior / obra coletiva concebida, desenvolvida e produzida pelo Laboratório de Experimentação Remota (RExLab)

Araranguá - SC, Brasil, 2016

Experimento Conversão de Energia Luminosa em Elétrica

Apresentação

O experimento Conversão de Energia Luminosa em Elétrica tem como objetivo prover uma ferramenta para auxiliar estudante do Ensino Fundaental e Médio no estudo do efeito fotovoltaico ,ao observar a conversão da energia luminosa em elétrica.

Arquitetura

O dispositivo está implementado a partir da estrutura padrão de hardware e software Básico. Na Figura 1, pode ser observado o diagrama da arquitetura do experimento.

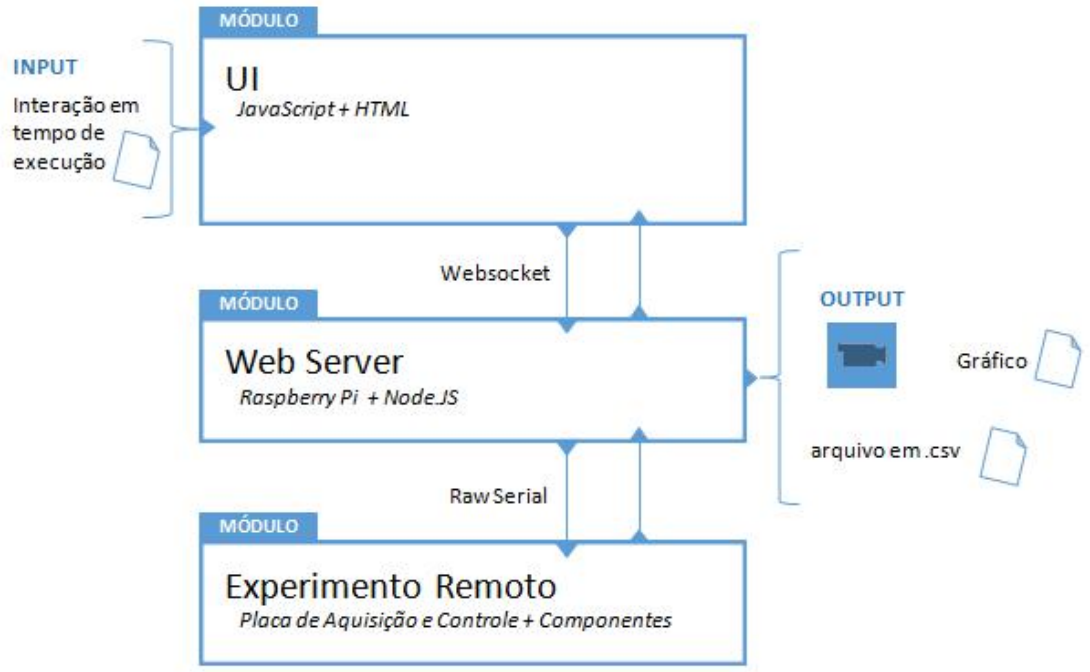


Figura 1 - Arquitetura do experimento: Conversão de Energia Luminosa em Elétrica.

Interface de Usuário (UI)

O experimento está disponível no sistema de gerenciamento RELLE(Remote Labs Learning Envirommente), que provê uma série de funcionalidades necessárias para o gerenciamento de experimentos remotos.

A interface de acesso ao experimento foi desenvolvida utilizando HTML juntamente com o framework front-end Bootstrap, o mesmo traz uma série de componentes prontos para o desenvolvimento além de prover tratamento para diferente tipos de resoluções de telas. Além de HTML e Bootstrap, é utilizado a biblioteca jQuery que traz uma série de funções JavaScript que simplificam o desenvolvimento.

A Figura 2 mostra como está disposto o experimento Conversão de Energia Luminosa em Elétrica no RELLE, e a Figura 3 o gráfico que também é exibido junto ao experimento.

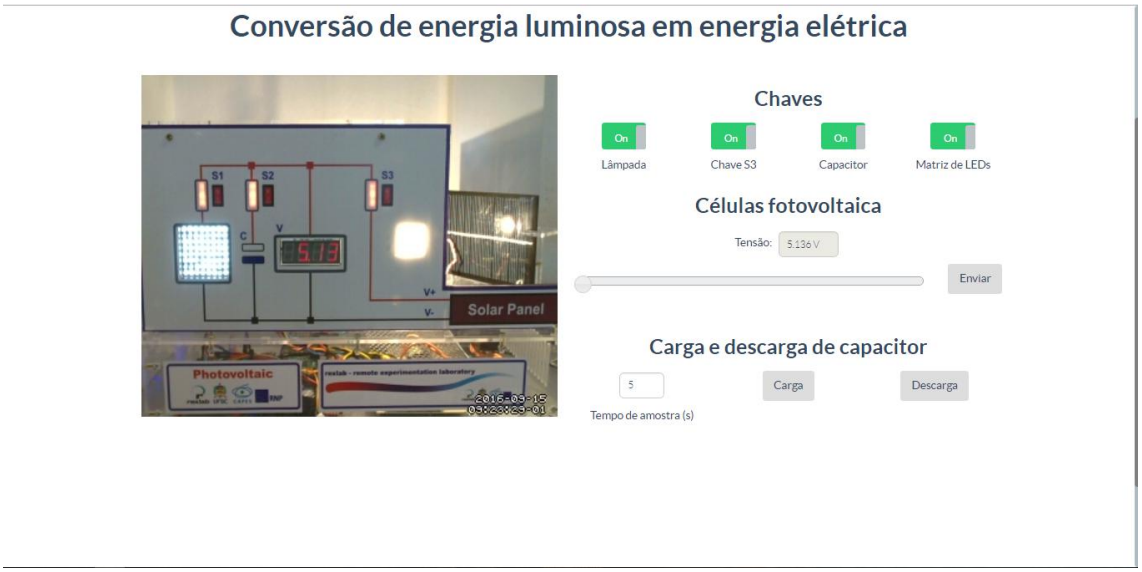


Figura 2 - Interface do usuário no RELLE

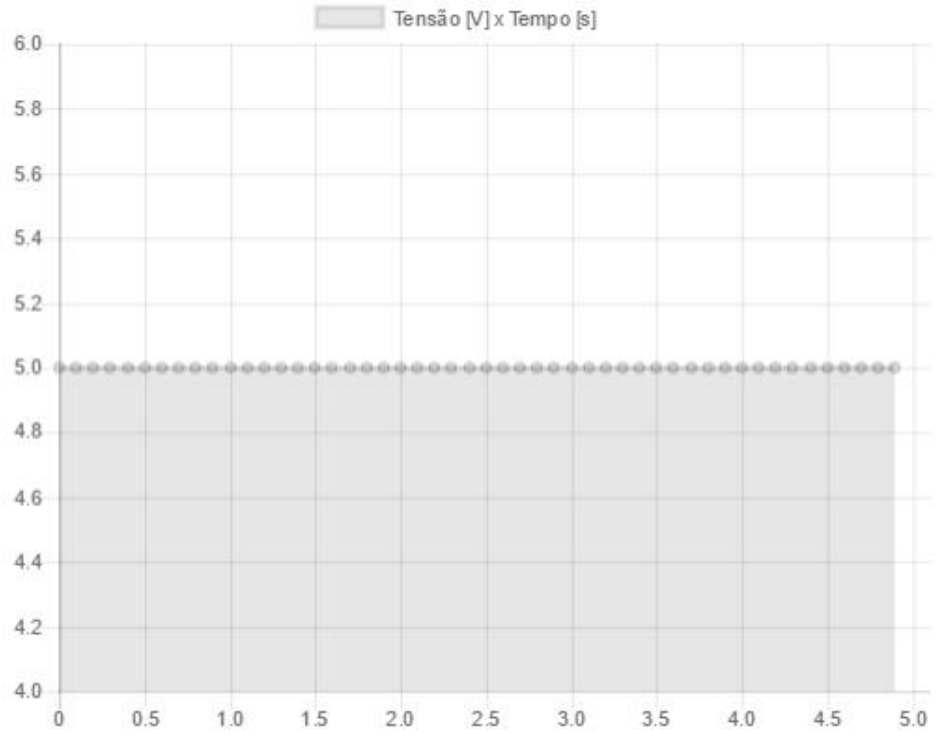


Figura 3 - Gráfico exibido na interface do usuário

Web Server

Atualmente, há uma ampla gama de bibliotecas e frameworks para construção de serviços web. Apesar de serviços baseados em HTTP predominarem a Internet, o uso do protocolo WebSocket é uma tendência em aplicações corporativas de grande porte. Uma das plataformas para desenvolvimento web para construção de serviços baseados em WebSocket é o framework NodeJS.

O NodeJS permite construir aplicações de servidor e de rede facilmente escaláveis. Ele é composto por um ambiente de execução multiplataforma e de código fonte aberto que interpreta códigos de aplicações escritas em Javascript. O NodeJS usa um modelo orientado a evento, com operações de entrada e saída não bloqueantes. Por este motivo, ele é ideal para aplicações em tempo real com troca intensa de dados entre dispositivos distribuídos.

A API para acesso às funcionalidades do SmartDevice[1] contém funções vinculadas à *listeners*, comuns ao paradigma de orientação a eventos. Este

[1] DOI: 10.1109/REV.2015.7087292

módulo usa a biblioteca Socket.io e é o ponto de partida da aplicação, onde o servidor é iniciado e eventos são vinculados. O Socket.io é composto por dois componentes: servidor e cliente, ao qual usa principalmente o protocolo WebSocket, e polling HTTP como compatibilidade reversa.

A autorização de sessão no SmartDevice garante a integridade do acesso exclusivo, já que o dispositivo exposto como um serviço pode ser utilizado concorrentemente por outro cliente. Apesar de algumas funcionalidades poderem ser utilizadas no modo observador, como consultar o estado das chaves e metadados, as funcionalidades de controle necessitam de consulta ao sistema de fila.

O sistema de fila, ou mesmo agendamento, pode ser externo ou interno ao SmartDevice. O primeiro é baseado em um token de autenticação provido pelo usuário e validado pelo SmartDevice. As implementações dos experimentos de física exemplificam o uso do sistema de reserva externo (próprio do RELLE). Já o controle de acesso no próprio SmartDevice é exemplificado pela implementação do Laboratório de desenvolvimento em Arduíno, pois neste encontra-se um modelo de acesso diferente dos anteriores.

O código fonte desenvolvido para comunicação serial e gerência dos sensores e atuadores são complementos para o NodeJS escritos em C++. Estes complementos são objetos compartilhados de vínculo dinâmico que pretendem dar suporte a códigos nativos, rapidez e portabilidade. Esses objetos compõem a abstração de cada experimento físico, que é representado por métodos e atributos intrínsecos a cada um. Por exemplo, são definidos os métodos de “get” e “set” para saídas digitais, “get” para valores de sensores, “get” e “set” para calibragem e configuração dos sensores.

O dispositivo central do experimento é o servidor de laboratório, que na plataforma desenvolvida pelo GT-MRE a escolha recaiu sobre o Raspberry Pi[2,] (Figura 4) modelo B+, que tem como principal função intermediar os acessos aos demais dispositivos de hardware dos experimentos via rede.

O servidor de laboratório (SL) tem função prover interfaceamento e gerenciamento para a conexão entre a rede (web) e a “placa de aquisição e

[2] O Raspberry Pi é um computador é baseado em um system on a chip (SoC) Broadcom BCM2835, que inclui um processador ARM1176JZFS rodando a 700 MHz, GPU VideoCore IV, e 512 MB de memória RAM em sua última revisão. O Raspeberry PI foi desenvolvido no Reino Unido pela Fundação Raspberry Pi.

controle” (PAC). O SL acessa a PAC para a coletar os dados dos sensores ou para enviar comandos para os atuadores, essa comunicação é feita via porta UART(Universal asynchronous Receiver/Transmitter) que se comunica via protocolo MODBUS[3].

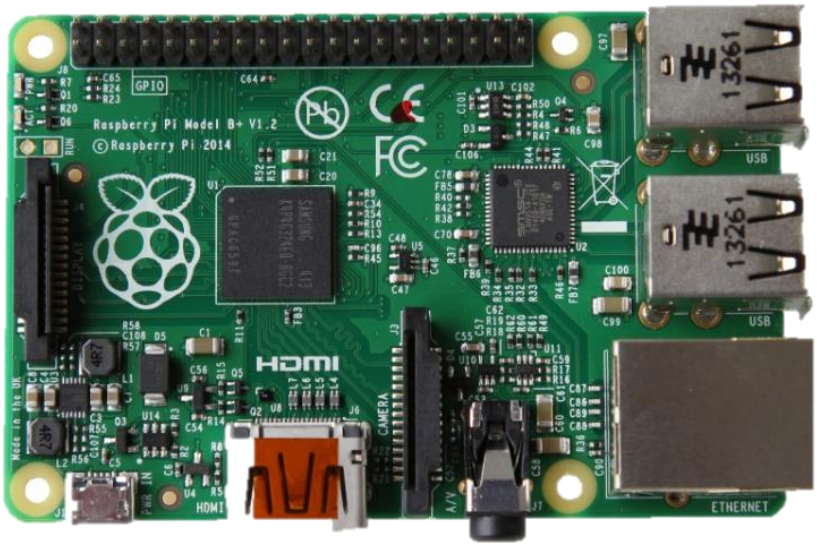


Figura 4 – Raspberry Pi, Model B+

API WebSocket

Os componentes da aplicação são suficientemente leves para serem executados por uma placa Raspberry Pi ou outro computador Linux de baixo custo. Um dos componentes, a API WebSocket, oferece uma interface aos sensores e atuadores na estrutura de um serviço web. A aplicação não requer alto uso da memória e pode ser utilizada em qualquer sistema Linux.

O resultado é uma arquitetura fracamente acoplada, adotada pelo GT-MRE, que habilita o compartilhamento dos experimentos em outras plataformas. Esse paradigma, chamado de SmartDevice já é utilizado no projeto Go-Lab[4], no qual estão bem destacadas aplicações clientes e servidor, e fornecem interfaces bem definidas entre o usuário e o sistema.

---

[3]Modbus é um protocolo de comunicação de dados utilizado em sistemas de automação industrial. É um dos protocolos mais utilizados em redes de Controladores lógicos programáveis (PLC) para aquisição de sinais (0 ou 1) de instrumentos e comandar atuadores. É de utilização livre e sem taxas de licenciamento.

[4]http://www.go-lab-project.eu/

Os tópicos seguintes apresentam com mais detalhes aspectos do serviço web utilizado no servidor de experimento, bem como as funcionalidades internas e as motivações para o uso de certos protocolos, padrões e ferramentas de desenvolvimento, conforme a Figura 5.

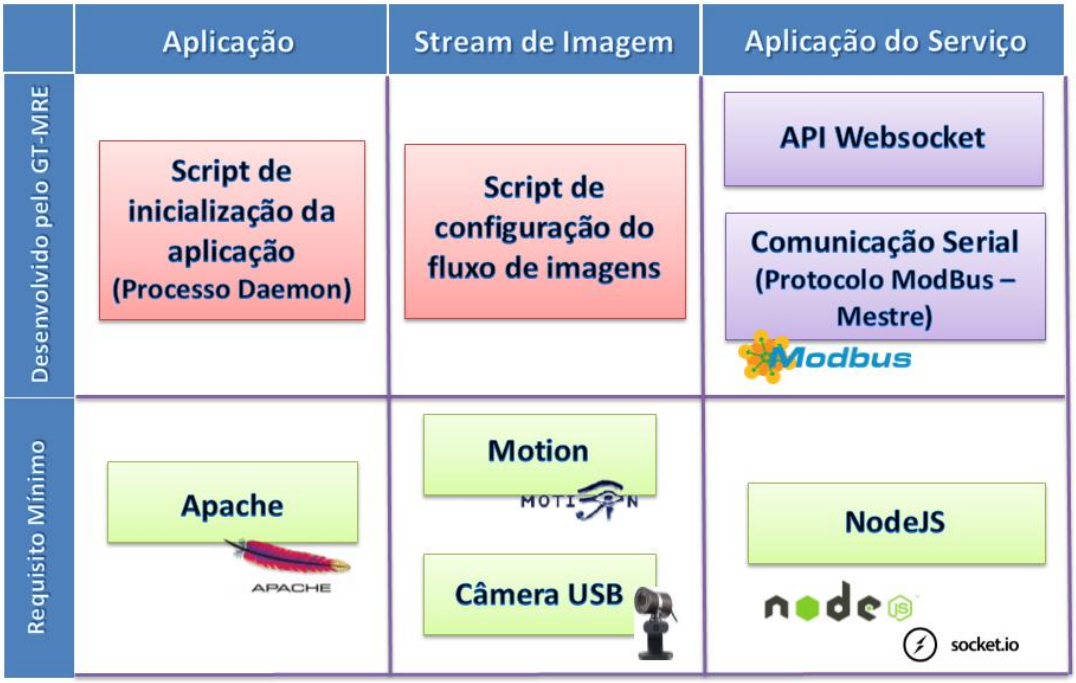


Figura 5 - Esquema de aplicação embarcada. Fonte: GT-MRE.

Controle e monitoramento do experimento

O SmartDevice é capaz de comunicar-se com sensores através do barramento serial (Porta UART). Ao invés de usar o protocolo serial em sua forma bruta, optamos por incluir o protocolo Modbus na camada de aplicação para identificação de erros, endereçamento e controle de colisão. Conectados ao mesmo barramento (rede), cada sistema embarcado, responsável por um ou mais sensores ou atuadores, é um dispositivo escravo que responde às requisições da aplicação que é executada no Raspberry Pi.

Um dos módulos desenvolvidos para aplicação é responsável pelo serviço de fila externo ou interno, sendo possível acoplar o serviço de fila provido pelo RELLE ou habilitar serviços internos. No primeiro caso, a aplicação usa a lógica necessária para validação de token de sessão enviado

pelo cliente. Na segunda, todo processo realizado pela web API de fila é realizado pelo SmartDevice.

Acesso à web API pelo cliente

A Figura 6 apresenta o esquema de comunicação no uso da API desenvolvido para o serviço/protótipo.

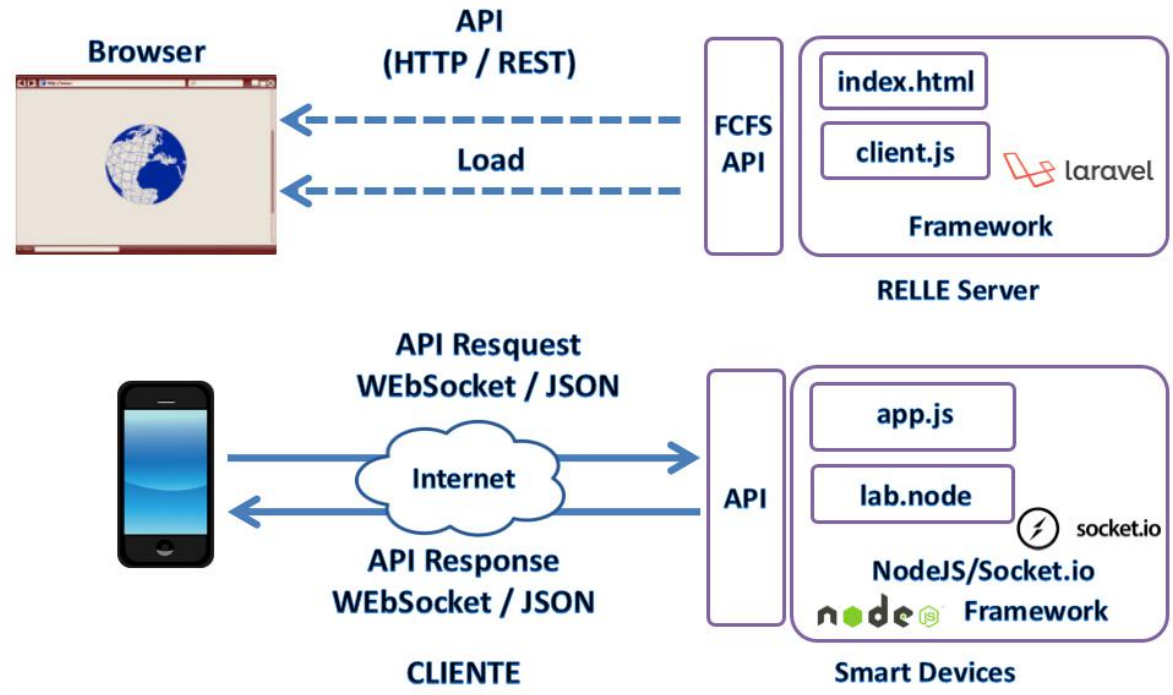


Figura 6 - Esquema de comunicação crossdomain no uso da API desenvolvida pelo GT-MRE. Fonte: Autores.

O cliente web disponibilizado pelo sistema RELLE é composto por um arquivo html, css e javascript diferentes para cada experimento. O RELLE provê uma página comum para cada experimento onde carrega os dados que foram inseridos no momento da publicação do experimento (armazenados numa base de dados). Por exemplo, o experimento de ID 1 é acessível pela URL “relle.ufsc.br/labs/1” pelo método GET e contém suas informações dentro do layout padrão do sistema. A partir do botão “Acessar” é possível disparar um evento para comunicação com a Web API FCFS (first-come first-served).

Ao obter a permissão no navegador, o cliente navegador poderá carregar os arquivos (html, css e js), pois a API já tem o seu token de sessão como usuário sendo servido. Após carregar o cliente para o SmartDevice (client.js), uma conexão WebSocket com este dispositivo é estabelecida.

Streaming de imagens

No GT-MRE foi optado pelo uso de câmeras web com conexão USB devido ao baixo custo e a facilidade de aquisição. O mesmo computador embarcado utilizado para controle do experimento também é o responsável pelo gerenciamento e disponibilização do streaming no formato MJPEG (Motion JPEG). O MJPEG é um formato de compressão de vídeo na qual cada frame de vídeo é comprimido separadamente como uma imagem JPEG.

Visto que existem muitos servidores de streaming de código aberto, optou-se pelo Motion[5] para explorar aspectos de leveza (utilização de poucos recursos) e configuração flexível. O Motion é um software escrito em C para sistemas Linux que usa a API de vídeo linux, e é capaz de detectar se uma parte significante da imagem tem mudado. Algumas variáveis são ajustadas através de seu arquivo de configuração principal para adequar-se aos requisitos de nossa aplicação.

Atualmente, os principais navegadores do mercado como Firefox, Google Chrome e Safari já possuem o suporte nativo para o streaming MJPEG. Para clientes Android existem bibliotecas de código fonte aberto para incluir um visualizador MJPEG em aplicações de código nativo.

Experimento Remoto

O objetivo do experimento é mostrar a transformação da energia luminosa em energia elétrica, utilizando uma lâmpada de filamento automotivo e uma célula fotovoltaica. Junto a estrutura foram adicionados um capacitor e um resistor permitindo testes dos tempos de carga e descarga do capacitor mediante a quantidade de energia produzida.

O experimento contém um fotovoltaico fixo encima de um servo motor, que permite ao usuário movimentar a placa para perto ou longe da luz, e com isto, respectivamente gerando mais ou menos energia. Esta geração de energia pode ser verificada através de um multímetro e uma matriz de LEDs, onde a intensidade de luz gerada por este painel é diretamente proporcional

[5]http://www.lavrsen.dk/foswiki/bin/view/Motion/WebHome

com quantidade de energia produzida. Na Figura 7 é exibida uma ilustração do experimento e suas partes.

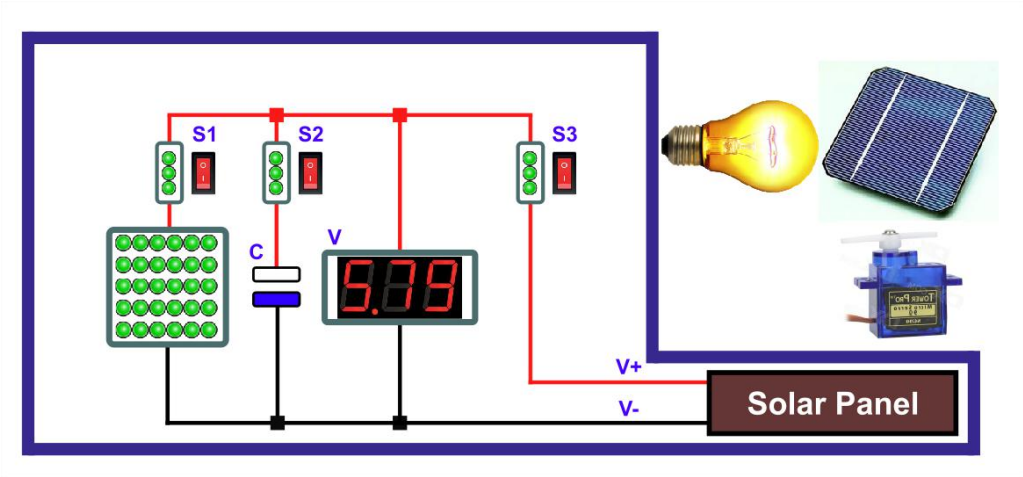


Figura 7 – Ilustração do Experimento Conversão de Energia Elétrica em Luminosa

Quantidade	Componente
1	Computador embarcado;
1	Placa de Aquisição e Controle;
1	Placa fotovoltaica;
1	Matriz de LEDS;
1	Lâmpada;
1	Capacitor;
1	Chave;
1	Servo motor;
1	Visor;
4	Relés;
1	Webcam.

Tabela 1 – Lista de quantidade e componentes.

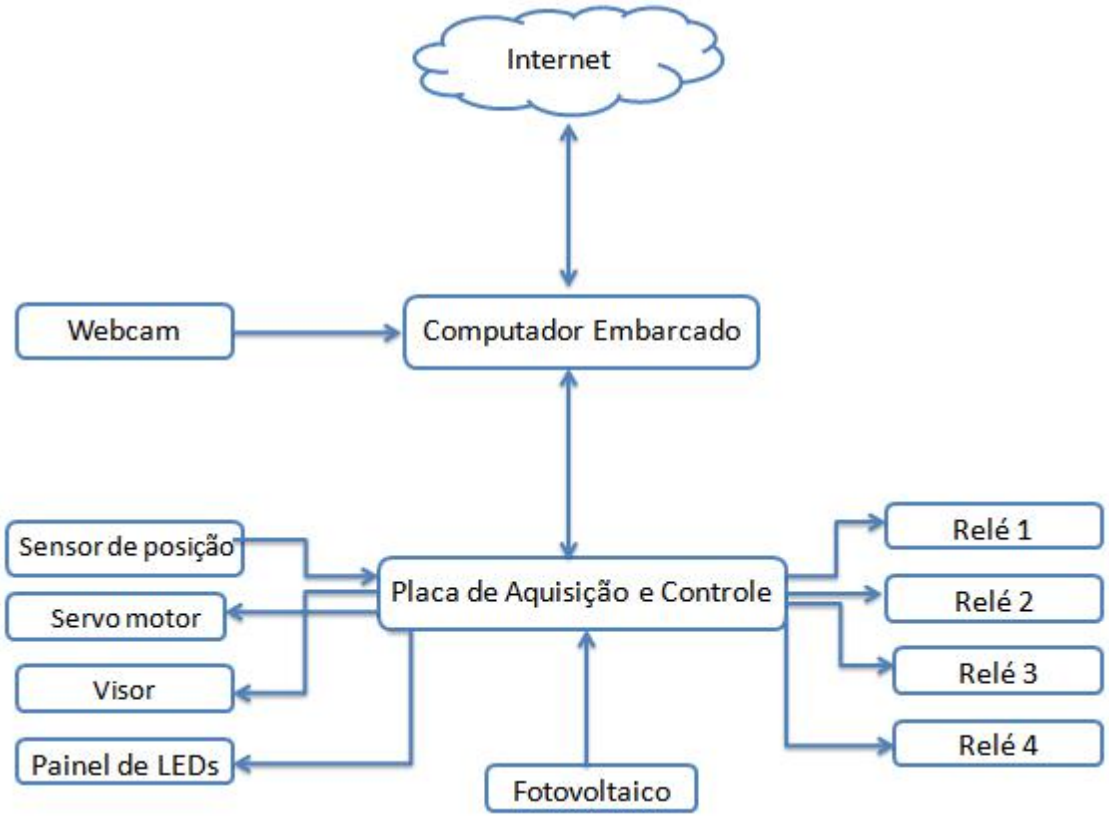


Figura 8 – Diagrama de blocos do experimento

Servo Motor: Tem como função girar a fotovoltaico para longe ou perto da fonte de luz.

Rele 1: Este relé conecta ou não o painel de LEDs ao circuito.

Rele 2: Este relé conecta ou não o capacitor ao circuito.

Rele 3: Este relé conecta ou não o fotovoltaico ao circuito.

Rele 4: Este relé liga ou desliga a lâmpada.

Painel de LEDs: Conjunto de LEDs que serão alimentados pelo fotovoltaico. A intensidade da luz emitido pelos LEDs é diretamente proporcional com a quantidade de energia gerada.

SBC (Single Board Computer): Um computador embarcado rodando Linux que contêm um conjunto de aplicativos oferecendo recursos para que os usuários possam se conectar ao experimento via WEB, visualizá-lo em tempo real e enviar comandos de experimentação.

RH (Recurso de Hardware): Um sistema embarcado dedicado a um controle mais sistemático dos elementos dos experimentos, tais como leituras de sensores e drivers dos atuadores.

Visor: Exibe tensão gerada pelo fotovoltaico. O RH coleta os dados de energia e envia para o visor via protocolo ModBus o valor da tensão a ser exibido.

Apêndices

Tutorial de reinicialização do experimento

Para reiniciar o experimento usa-se um terminal para conexão ssh, por exemplo o software PuTTY realiza este tipo de conexão. O software pode ser baixado no seguinte endereço: http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html. Utilizando o PuTTY, basta inserir o endereço IP do experimento que se deseja reinicializar.

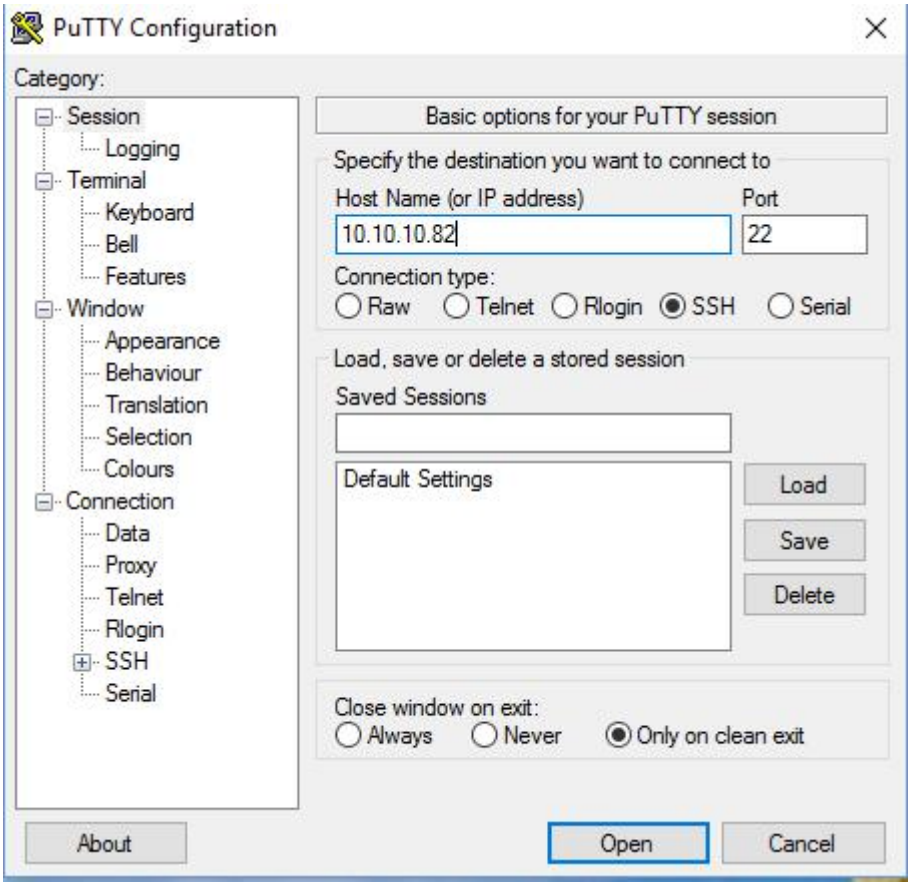


Figura 9 – PuTTY

Ao iniciar a conexão será aberto um terminal (Figura 10), e um usuário (user) para autenticação será solicitado. Recomenda-se autenticar com o usuário root, e logo em seguida, será requisitada a senha do computador embarcado. E por fim, para reiniciar o computador embarcado, digite o comando *reboot* no terminal.

Figura 10 - Terminal SSH com experimento

Verificação e reinicio do serviço

Para verificar os se os serviços do laboratório remoto estão rodando, basta usar o comando "*ps –aux | grep node"* que ira verificar os processos rodando referente ao servidor web Node.JS responsável por executar o serviço da aplicação. Caso o serviço esteja rodando, o resultado será algo similar a Figura 11 que exibe o usuário e número do processo em execução. Neste caso o processo PID 2434.

```
root@raspberrypi:~# ps -aux | grep node
warning: bad ps syntax, perhaps a bogus '-'?
See http://gitorious.org/procps/procps/blobs/master/Documentation/FAQ
root      2434  0.1  9.7 1182324 43412 ?       Sl   May12 102:36 /usr/local/bin/node /home/conducao_app/apps.js
root     21479  0.0  0.3   3520  1740 pts/0    S+   14:17   0:00 grep node
```

Figura 11 - Verificação do serviço

Ações de iniciar, pausar ou verificar status do serviço podem serem executadas usando os comandos *"service NOME DO SERVIÇO start|stop|status".*

Manutenção do streaming de vídeo

O vídeo é transmitido pelo software Motion. Para instalação do software pode-se fazer seu download via repositório através do comando *"apt-get install*

motion” e acessar os arquivos de configurações *motion* e *motion.conf* através de algum editor de código no diretório */etc/default/motion* definindo o parâmetro *start_motion_deamon* para o valor *yes.*

As configurações relacionadas a qualidade da imagem e a transmissão ficam disponíveis no arquivo motion.conf no diretório /etc/motion/. Ainda para inicio da transmissão os parâmetros *deamon* e *webcam_localhost* devem ser mudados para *on* e *off*, respectivamente.