Redes Móveis e Sem Fios

Instituto Superior Técnico

Sistema de controlo de música numa casa

Professor:
António Grilo

Elementos do Grupo:
Diana Fonseca Nº 86967
Gonçalo Mão-Cheia Nº 87007

Conteúdo

1 Introdução

Com o desenvolvimento de novas tecnologias nos últimos anos, o mundo encontra-se cada vez mais recetivo a novas ideias e projetos tecnológicos que permitem melhorar a qualidade de vida. Um dos temas mais falados dentro das áreas tecnológicas é a domótica. Domótica baseia-se na ideia da automatização das tarefas comuns numa casa, deste modo surge assim a ideia de Casas Inteligentes.

Com base nesta ideia de Casas Inteligentes e porque nos foi dada a possibilidade de criar um projeto na área de IoT (Internet of Things) decidimos assim criar um sistema de controlo de música entre divisões de uma casa. Este sistema é desenvolvido com base num protótipo que permite o controlo de música entre duas divisões através de sensores de movimento, em que quem usufrui do espaço, tem controlo sobre o mesmo através de uma aplicação.

O objetivo deste projeto é realizar um protótipo capaz de demonstrar esta ideia. Assim, pretende-se criar um protótipo capaz de reproduzir música em duas divisões diferentes, sendo possível alterar a música, volume e divisão na qual a mesma é reproduzida através de uma *mobile app*. O controlo da música nas colunas é possível através do uso de um Raspberry Pi conectado a estas através de um *sound card*. Pretende-se também que através de um NodeMCU em cada divisão seja possível fazer tocar música através de sensores de movimento e que seja possível variar o volume através de botões.

Neste relatório apresentamos a descrição do trabalho desenvolvido na fase final. Começamos assim por descrever o projeto, a arquitetura planeada, a *mobile app* e a conclusão, falando do que foi possível fazer e do que se poderia fazer caso se quisesse melhorar a implementação.

2 Apresentação do projeto

O projeto apresentado consiste num sistema inteligente de controlo de música numa casa. Através de sensores de movimento, podemos perceber em que divisão da casa se encontra uma pessoa e, consequentemente ligar automaticamente música nessa divisão. O utilizador tem acesso ao controlo do sistema de música através de uma aplicação descrita de seguida neste relatório que lhe permite continuar/parar a música, mudar de música e/ou aumentar e diminuir o som. Tem acesso também a botões de controlo de volume da música nas divisões onde se encontra.

Para o desenvolvimento do sistema de música planeámos usar o seguinte material:

- 1 Raspberry Pi;
- 2 NodeMCU EXP8266;
- 2 sensores de movimento;
- 4 botões;
- 2 colunas de som;

Com base no movimento detetado pelos sensores de movimento, o sinal é recebido pelo NodeMCU, que comunica com o *web server*, e este, por sua vez, com um Raspberry Pi que controla a música na coluna de som presente nos espaço onde o movimento é detetado. Existem 2 botões ligados a cada NodeMCU de modo a que quem está na divisão a ouvir a música possa controlar o volume premindo os mesmos. Ao ser premido um botão, há uma instrução que é enviada para o *web server* através do NodeMCU e este faz atuar no Raspberry Pi o seu efeito. Temos assim 2 divisões, cada uma com um EXP8266 onde se encontram ligados um botão de aumento de volume, um botão de diminuição de volume e um sensor de movimento.

O utilizador tem acesso à aplicação que foi desenvolvida em Android Studio e mediante as suas ações na mesma, há a comunicação das instruções ao *web server*, instruções essas passadas ao Raspberry Pi como descrito no caso do uso dos botões físicos.

Os atuadores do projeto, neste caso as colunas de som, encontram-se assim ligadas a um Raspberry Pi.

A transmissão de informação entre o Raspberry Pi, os NodeMCU e a aplicação móvel será realizada pelo *web server* Firebase.

Desta forma, o Raspberry Pi recebe instruções do Firebase e atua de acordo com as mesma, enquanto que a *mobile app* e os NodeMCU enviam instruções para o Firebase.

Na Figura 1 podemos ver um esquema representativo da descrição da arquitetura do projeto em desenvolvimento. De notar que na imagem apresentada, as ligações físicas entre os elementos encontram-se representadas por linhas pretas. As linhas laranja tracejadas correspondem a uma comunicação wireless entre os elementos.

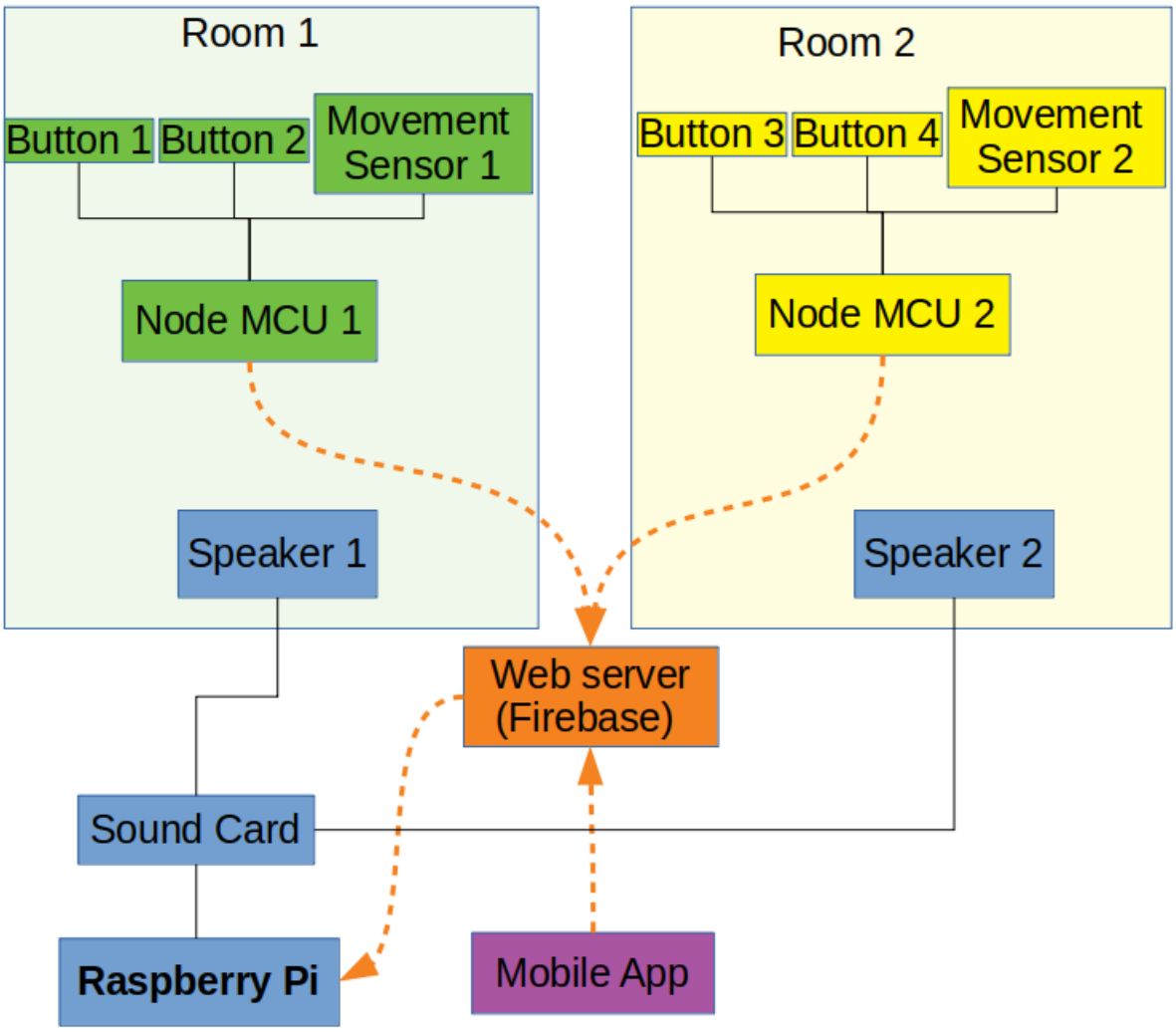

Figura 1: Diagrama com a arquitetura geral do projeto

3 Arquitetura do projeto

Em termos de arquitetura, este projeto pode ser dividido em 3 partes principais: o Raspberry Pi que está diretamente ligado aos atuadores deste projeto (as colunas de som), o *web server* para onde os comandos são enviados e consequentemente lidos pelo Raspberry Pi e os NodeMCU que têm a si conectados um conjunto de sensores (botões e sensores de movimento).

3.1 Raspberry Pi

Em termos de conexões físicas, o Raspberry Pi está conectado a um *sound card* através de USB, o qual está consequentemente ligado às colunas de som através de um cabo de som de 3.5 mm, como representado na Figura 2.

Figura 2: Ligação do Raspberry Pi com as colunas de som

Relativamente à implementação de código no Raspberry Pi, corre um *script* em Python.

Neste *script* é utilizada a biblioteca Firebase, de forma a permitir fazer a leitura e de seguida apagar mensagens do *web server*.

Primeiramente é feita a conexão ao *web server*. Depois é necessário abrir uma sessão do programa Tizonia. Este é um programa *open-source* que permite aceder a vários reprodutores de música através do terminal de comandos do linux. Neste caso, o programa Tizonia vai permitir o acesso ao Spotify (um serviço de streaming de música).

Existe, no entanto, um problema: a sessão do programa Tizonia necessita de permanecer aberta, para mais tarde ser possível enviar comandos através do *script* de Python, de forma

a alterar a música e o volume. Devido a isto o programa Tizonia é aberto com o comando tmux.

O comando tmux é um multiplexer de terminais, que permite fazer *detach* de um processo (deixando-o a correr no *background*) e fazer *reattach* do mesmo mais tarde. É através deste comando e da sua funcionalidade *send-keys* que é possível comunicar com o processo do programa Tizonia enquanto o *script* de Python lê e processa continuamente as mensagens contidas no *web server*.

Na Figura 3 encontra-se o fluxograma do *script* que é executado no Raspberry Pi.

As instruções que serão interpretadas pelo Raspberry Pi são as seguintes:

- PLAY: pára ou continua a música
- UP: aumenta o volume
- DOWN: diminui o volume
- NEXT: muda para a música seguinte
- PREV: muda para a música anterior
- ONOFF: liga ou desliga o programa Tizonia
- RIGHT: redireciona o som para a divisão à direita
- LEFT: redireciona o som para a divisão à esquerda

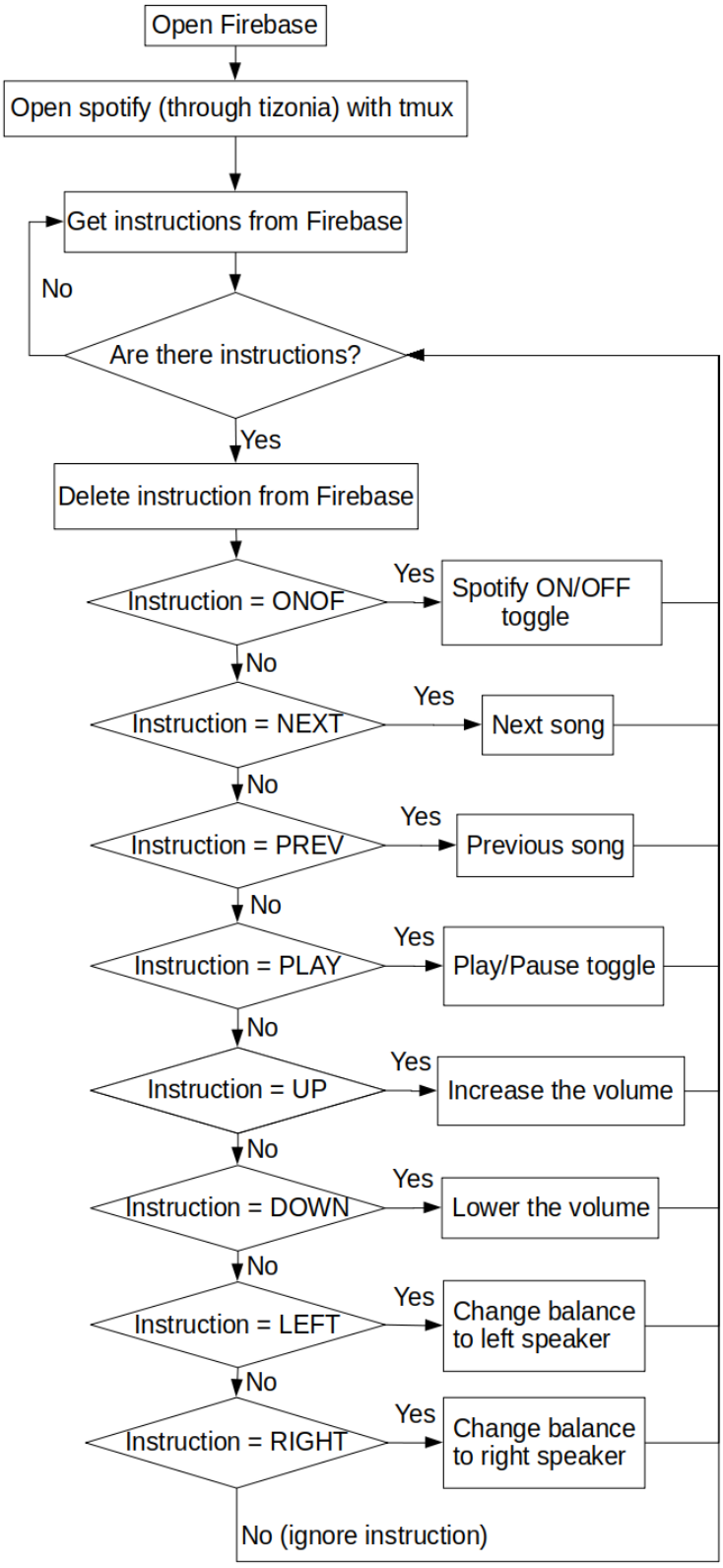

Figura 3: Fluxograma do *script* em Python executada no Raspberry Pi

3.2 *Web Server*

O *web server* foi implementado através da plataforma Firebase da Google. É assim utilizada um Firebase Realtime Database, que permite estabelecer a conexão entre a *mobile app*, os NodeMCUs e o Raspberry Pi. Através deste *web server* conseguimos assim transmitir mensagens em tempo real de forma a oferecer ao utilizador uma experiência natural e fluida.

3.3 NodeMCU ESP8266

Este dispositivo é um micro-controlador que permite comunicação com *wi-fi* fazendo conexões TCP/IP. É esta placa que permite enviar para o *web server* as instruções correspondentes à deteção de movimento e do sinal de botões de volume. Isto é conseguido através da sua configuração para se ligar à rede *wi-fi* local e para comunicar com o *web server*. Para a sua configuração neste projeto foi usado o Arduino IDE.

3.3.1 Implementação do ESP8266

Na Figura 4 podemos ver o esquema de ligação dos botões de controlo de volume e o sensor de movimento. Os dois botões usados são digitais e ao serem pressionados, o NodeMCU deteta e envia para a base de dados a instrução UP ou DOWN, consoante o botão pressionado. O sensor ao detetar movimento, envia informação de movimento para o NodeMCU que envia para o Firebase a instrução DIV1 ou DIV2 (mediante a divisão).

A comunicação do NodeMCU com o Firebase é possível após a configuração do mesmo para se conectar à rede rede *wi-fi* da casa em que está e que se ligue a esta após lhe ser atribuído um IP. Fica assim apto a comunicar com o *web server* que também ele é configurado no código do mesmo.

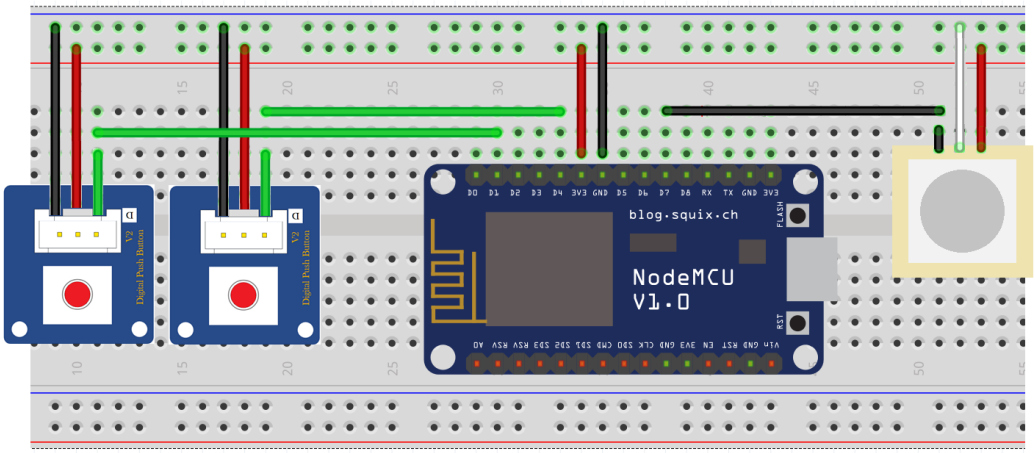

Figura 4: Ligação do Raspberry Pi com as colunas de som

3.3.2 Configuração do ESP8266

Para a configuração do NodeMCU definimos o Firebase a que o nosso equipamento deve comunicar as instruções e as informações da rede *wi-fi* a que é para se ligar. De seguida

definimos os locais das conexões dos botões e do sensor através dos números associados a cada pin do Raspberry Pi e definimos os dois botões e o sensor de movimento como *input* e o D0 como *output* de modo a fazer-se a conexão com a Internet e de seguida a sessão com o Firebase.

Seguidamente foram configurados cada botão para, quando pressionados, enviarem a instrução UP ou DOWN, consoante o botão pressionado, de modo a aumentar ou diminuir o volume a música. Quando o sensor de movimento deteta algo, envia uma instrução para o Firebase de modo a começar a música, sendo que temos um sensor na divisão 1 que envia LEFT para poder começar a tocar musica na divisão 1 e caso seja o sensor da divisão 2 a ser acionado, o seu respetivo NodeMCU envia a instrução LEFT para o *web server* poder comunicar com o Raspberry Pi e este ligar a música na coluna da divisão 2.

4 *Mobile App*

4.1 *Front End*

A aplicação de controlo do sistema pelo utilizador foi desenvolvida no programa Android Studio. A sua interface pode ser observada na Figura 5.

Figura 5: Interface da *Mobile App*

A interface é simples, com 8 botões que permitem realizar algumas funções básicas relativamente à reprodução de música. As mensagens enviadas por cada botão são:

- PAUSE/PLAY: PLAY
- VOL +: UP
- VOL -: DOWN
- NEXT: NEXT
- PREVIOUS: PREV
- ON/OFF: ONOFF
- LEFT: LEFT
- RIGHT : RIGHT

4.2 *Back End*

Primeiramente, quando é ligada, é estabelecida a conexão da *mobile app* com o *web server* Firebase.

Seguidamente entra num loop, no qual aguarda que um botão seja premido e envia uma instrução para o Firebase de acordo com o botão premido.

5 Conclusão

De uma forma geral, conseguimos provar o conceito a que tínhamos pretendido chegar desde o início. Existiram, no entanto, problemas na implementação de *hardware* (nomeadamente com os botões e sensores de movimento) nos NodeMCU, mas é possível simular os *inputs* e demonstrar que a comunicação entre os NodeMCU é, não só existente, como também funcional.

O *web server* usado neste projeto foi o Firebase da Google. Este recebe todas as instruções enviadas pelos NodeMCU e pela *mobile app*. Todas as instruções que permitem o funcionamento deste projeto são lidas do Firebase.

Relativamente à *mobile app*, esta está 100% funcional e, apesar de simples, faz exatamente o que é preciso, enviando mensagens para o *web server* com sucesso.

O Raspberry Pi cumpre todos os requisitos propostos, conseguindo descodificar todas as instruções provenientes do *web server* e descartar instruções não válidas, de forma a não ocorrerem falhas durante o seu funcionamento.

Caso se pretendesse ampliar esta ideia a uma casa com mais do que duas divisões, a implementação de um Raspberry Pi por cada 2 divisões não é ideal. Deveria assim ser trocada em prol da utilização de um Raspberry Pi por divisão, que permitiria o uso de *stereo* (estéreo em português) em todas as divisões. Esta implementação melhoraria a qualidade de som e também facilitaria a organização e divisão do produto, tendo em conta que cada Raspberry Pi ficaria associado a uma divisão. Isto implicaria também alterações na *mobile app* de forma a fazer com que a seleção das divisões fosse mais intuitiva.