



Universidade do Minho
Escola de Engenharia

Laboratórios de Telecomunicações e Informática II

ENGENHARIA DE TELECOMUNICAÇÕES E INFORMÁTICA
2020/2021

(Docentes: Bruno Daniel Mestre Viana Ribeiro, Vadym Serhiyovych Hapanchak,
José Augusto Afonso, Sérgio Adriano Fernandes Lopes)

16 de maio de 2021

Relatório

FASE B

Rui Filipe Ribeiro Freitas – a84121@alunos.uminho.pt

Sandro Teixeira Ribeiro – a85316@alunos.uminho.pt

Tiago João Pereira Ferreira – a85392@alunos.uminho.pt

Índice

Índice de figuras	3
Índice de tabelas	4
Lista de abreviaturas	5
Introdução	6
1. Planeamento do projeto	7
1.1. Planeamento temporal	7
1.2. Tecnologias/Ferramentas necessárias	8
1.2.1. Ao nível do <i>hardware</i>	8
1.2.2. Ao nível do <i>software</i>	9
2. Fundamentos	10
2.1. Sistema central	10
2.2. Concentrador	10
2.3. Sistema sensor simulado	11
3. Desenvolvimento	12
3.1. Protocolo de comunicação entre concentradores e sistema central	12
3.2. Base de dados	14
3.3. Interface Web	15
4. Implementação	17
4.1. Sistema sensor simulado	17
4.1.1. Código	17
4.2. Concentrador	18
4.2.1. Código	18
4.2.2. Fluxograma	19
4.3. Gestor de Serviços	20
4.3.1. Código	20
4.3.2. Fluxograma	21
5. Testes e discussão de resultados	22
Conclusão	24
Referências	25
Autoavaliação	26

Índice de figuras

Figura 1 - Planeamento temporal tabela.....	7
Figura 2 - Planeamento temporal gráfico.....	7
Figura 3 - Protocolo TCP vs. Protocolo UDP.	10
Figura 4 - Esquema da Fase B.	11
Figura 5 - Mensagem START.	12
Figura 6 - Mensagem STOP.....	12
Figura 7 - Mensagem ERROR.	12
Figura 8 - Mensagem DATA.	13
Figura 9 - Mensagem BEGIN.	13
Figura 10 - Mensagem END.	13
Figura 11 - Diagrama de Entidades e Relacionamentos.....	14
Figura 12 - Base de dados.....	14
Figura 13 - Página inicial da interface web.....	15
Figura 14 - Tabela das amostras dos sensores PIR.	16
Figura 15 - Tabela das amostras dos sensores LDR.....	16
Figura 16 - Tabela das posições dos sistemas sensores.	16
Figura 17 - Gráfico dos luxs.	16
Figura 18 - Criação do socket.....	17
Figura 19 - Comunicação TCP entre concentrador e sistema central.	18
Figura 20 - Criação das threads.	18
Figura 21 - Fluxograma do concentrador.	19
Figura 22 - Código Gestor de serviço.....	20
Figura 23 - Fluxograma Gestor de serviço.....	21
Figura 24 - Comunicação entre 2 sistemas sensores simulados e um concentrador.	22
Figura 25 - Bases de dados.	23
Figura 26 - Amostras dos sensores de movimento na interface web.	23

Índice de tabelas

Tabela 1 - Tecnologias ao nível do hardware.	8
Tabela 2 - Tecnologias ao nível do software.	9

Lista de abreviaturas

LTI II – Laboratórios de Telecomunicações e Informática II

TCP – Transmission Control Protocol

UDP – User Datagram Protocol

CSV – Comma-Separated Values

Introdução

No âmbito da Unidade Curricular de LTI II (Laboratórios de Telecomunicações e Informática II) foi-nos proposto o desenvolvimento de um sistema de iluminação inteligente que permita monitorizar a ocupação e a luminosidade em diferentes áreas de uma residência, bem como controlar a luminosidade desses ambientes através do acionamento de lâmpadas.

Este relatório diz respeito à segunda fase de 3 do desenvolvimento deste projeto em que nesta fase os objetivos passam pela implementação de sistemas sensores simulados e de um sistema central que será responsável pela comunicação com os vários concentradores através de um protocolo de comunicação por nós realizado. Para além disso este sistema central terá como objetivo apresentar ao utilizador um menu com várias funções como observar gráficos sobre a luminosidade e movimento numa dada área.

De modo a sermos capazes de cumprir com os objetivos deste projeto semestral devemos pôr em prática conhecimentos adquiridos noutras unidades curriculares, nomeadamente Redes de Computadores, Sistemas Operativos, Sistemas Distribuídos, Eletrónica e Laboratórios de Telecomunicações e Informática I.

1. Planeamento do projeto

1.1. Planeamento temporal

De modo que o grupo se mantenha focado no trabalho e com um compromisso para cumprir horários, resolvemos planear as tarefas a fazer nesta fase. Na figura 1 observamos em forma de tabela os vários assuntos a ser tratados nesta fase com um período dado por nós para cumprir. Em relação à figura 2 demonstramos em forma de gráfico o tempo despendido nas várias tarefas. Ambas as figuras foram retiradas do programa *Gantt*.

GANTT project		
Nome	Data de início	Data de fim
☐ Fase B	14-04-2021	14-06-2021
• Planeamento Temporal	14-04-2021	16-04-2021
• Definição da Arquitetura do Sistema	20-04-2021	20-04-2021
• Início da elaboração do relatório	21-04-2021	21-04-2021
• Desenvolvimento código sensores simulados	22-04-2021	28-04-2021
• Desenvolvimento código Concentrador	29-04-2021	05-05-2021
• Desenvolvimento código Sistema Central	06-05-2021	11-05-2021
• Realização de Testes	12-05-2021	12-05-2021
• Discussão dos resultados obtidos	13-05-2021	13-05-2021
• Conclusão e revisão do relatório	14-06-2021	14-06-2021

Figura 1 - Planeamento temporal tabela.

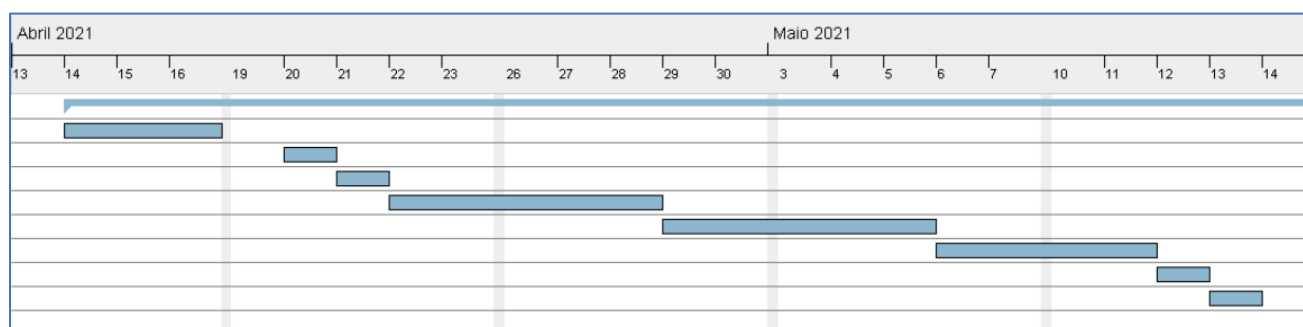


Figura 2 - Planeamento temporal gráfico.

1.2. Tecnologias/Ferramentas necessárias

Em qualquer projeto são necessárias certas tecnologias/ferramentas que nos facilitem o trabalho e ao mesmo tempo que aumentem a nossa produtividade. Para este trabalho em específico teremos de recorrer a tecnologias específicas tanto ao nível do *hardware* como ao nível do *software*, de modo a cumprir com os objetivos propostos.

1.2.1. Ao nível do hardware

No que toca à parte do *hardware* deste nosso trabalho as tecnologias por nós utilizadas estão presentes na tabela 1, sendo que estas podem vir a ser alteradas no futuro.









Tabela 1 - Tecnologias ao nível do hardware.

FERRAMENTA	QUANTIDADE	UTILIZAÇÃO
Computador Portátil	1	Desenvolvimento de código, elaboração do relatório, pesquisa e testes
Placa ESP32	1	Comunicação com os sensores e os LEDs
<i>Sensor de movimento</i> (PIR HC-SR501)	1	Deteção de movimento
Sensor de luminosidade (LDR 5...10kΩ)	1	Medição da luminosidade
LED	3	Simulação de lâmpadas
Cabo Micro USB	1	Ligações físicas entre o PC e a placa ESP32
Fios de ligação	Vários	Efetuar as ligações entre as placas ESP32 e os sensores e LEDs
Resistência	Várias	Limitar a corrente drenada pelos LEDs

1.2.2. Ao nível do *software*

Em relação ao *software* que irá ser utilizado na execução do nosso projeto este passará maioritariamente pelas aplicações apresentadas na tabela 2. Nesta realçamos as mais importantes e fundamentais, sabendo que, no entanto, poderão sofrer alterações no futuro dado ainda nos encontrarmos numa fase mais introdutória do trabalho.

Tabela 2 - Tecnologias ao nível do *software*.

LOGO	DESCRIÇÃO	UTILIZAÇÃO
	Microsoft Word	Realização e sincronização do relatório do projeto
	GanttProject	Planeamento do projeto através de diagramas de Gantt
	Arduino IDE	Software para a conexão com a placa ESP32 e comunicação com o mesmo
	Visual Studio Code	Execução e compilação de código
	GitHub	Sincronização do código da aplicação
	Visual Paradigm	Implementação de fluxogramas para a elaboração da aplicação e do projeto
	Google Drive	Sincronização de ficheiros essenciais à realização do trabalho
	Discord	Comunicação entre o grupo

2. Fundamentos

2.1. Sistema central

O sistema central deve recolher a informação enviada pelos concentradores da rede de forma a apresentar um conjunto de dados de monitorização dos valores recolhidos nos sensores. Esta comunicação será baseada no protocolo TCP/IP. Os dados recebidos dos concentradores podem ser filtrados, antes de serem registados (em base de dados simples), para que não tenham o mesmo nível de detalhe. Nesta fase, o sistema central deve possuir uma interface de administração através de serviço web, para que o sistema possa ser consultado local ou remotamente. O papel do sistema central é o de aplicação ou de serviço TCP/IP, contendo uma lista dos concentradores com os quais pode estabelecer comunicação.

Nesta fase será efetuada a implementação em linguagem C de um gestor de serviços responsável por comunicar com os concentradores e guardar na base de dados os valores das amostras recebidas. Após isso através da linguagem *JavaScript* será utilizada essa base de dados para apresentar ao utilizador numa interface web as mais variadas informações sobre os sensores assim como gráficos resultantes dessas informações.

2.2. Concentrador

Relativamente ao concentrador, para esta fase foi implementada a comunicação com o sistema central. Para isto será necessária a criação de uma nova *thread* de modo que este pudesse estar a receber dados dos vários sensores e a comunicar com o sistema central ao mesmo tempo. Para além disso será necessário implementar o protocolo de comunicação UDP para que este consiga comunicar com os sensores simulados ao mesmo tempo sem perdas. Este protocolo difere do TCP pois não realiza uma conexão cliente-servidor apenas envia e recebe tramas. No que toca a segurança na comunicação o protocolo TCP é mais fiável visto este realizar correção de erros pois mantém uma conexão estabelecida como demonstrado na figura 3 [1].

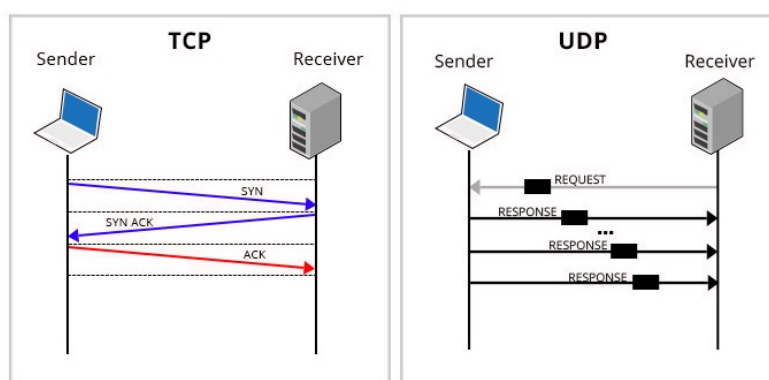


Figura 3 - Protocolo TCP vs. Protocolo UDP.

2.3. Sistema sensor simulado

A implementação de um sistema sensor simulado permite o teste das funcionalidades do sistema central e concentradores num ambiente com múltiplos sistemas sensores ativos em simultâneo.

Estes sistemas sensores devem ser implementados numa aplicação software e devem incluir um ou mais ficheiros com valores previamente armazenados com o resultado da monitorização do funcionamento de um sistema sensor real. Deve também utilizar o protocolo de comunicação definido anteriormente na Fase A, mas para estes sistemas simulados, as mensagens deverão ser encapsuladas em pacotes UDP.

Na figura seguinte apresentamos o esquema da Fase B com o sistema central a comunicar com o concentrador através do protocolo TCP/IP e o concentrador com o sistema sensor simulado através de uma ligação UDP/IP.

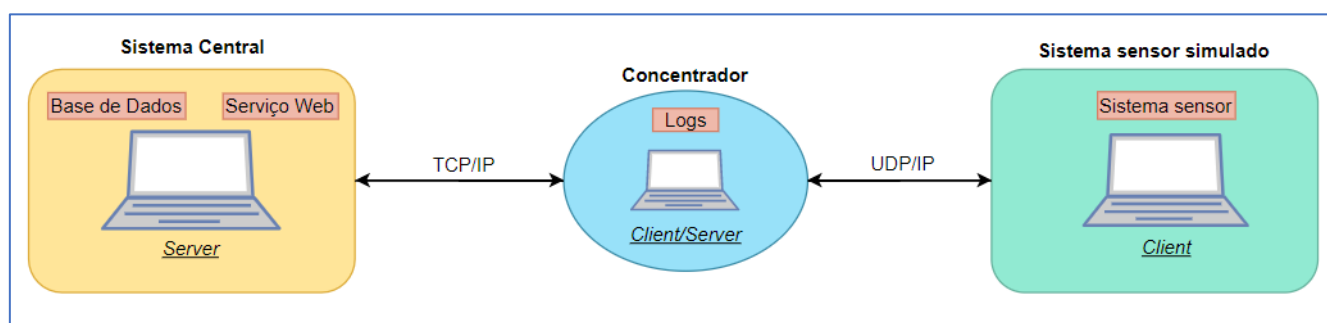


Figura 4 - Esquema da Fase B.

3. Desenvolvimento

3.1. Protocolo de comunicação entre concentradores e sistema central

O protocolo de comunicação serve para os concentradores enviarem para o sistema central os dados recolhidos nos sistemas sensores através de um conjunto de mensagens, as quais devem ser encapsuladas através do protocolo TCP. Como o TCP já é um protocolo confirmado e orientado à conexão não é necessário implementar mecanismos de controlo de fluxo. No entanto é necessário incluir uma forma de os gestores saberem quando a comunicação com um determinado concentrador é terminada. Para isso decidimos implementar vários tipos de mensagens, que são apresentados em seguida.

- **START** – A mensagem *START* é enviada do concentrador para o sistema central a pedir autorização para enviar dados. Esta mensagem tem os campos apresentados na figura 5.

Tipo (1 byte)	Local (1 byte)	Password (4 bytes)
------------------	-------------------	-----------------------

Figura 5 - Mensagem *START*.

- **STOP** – A mensagem *STOP* é enviada do concentrador para o sistema central a indicar que já não tem dados para enviar e que deseja terminar a sessão TCP. Esta mensagem tem os campos apresentados na figura 6.

Tipo (1 byte)	Local (1 byte)	Password (4 bytes)
------------------	-------------------	-----------------------

Figura 6 - Mensagem *STOP*.

- **ERROR** – A mensagem *ERROR* é enviada do concentrador para o sistema central a indicar que ocorreu um erro. Esta mensagem tem os campos apresentados na figura 7.

Tipo (1 byte)	Local (1 byte)	Password (4 bytes)
------------------	-------------------	-----------------------

Figura 7 - Mensagem *ERROR*.

- **DATA LDR** – A mensagem *DATA LDR* é enviada do concentrador para um gestor de serviços com a posição dos sistemas sensores que estão conectados e os valores das amostras recolhidas relativas ao LDR de todos os sistemas sensores que estão conectados. Esta mensagem tem os campos apresentados na figura 8.

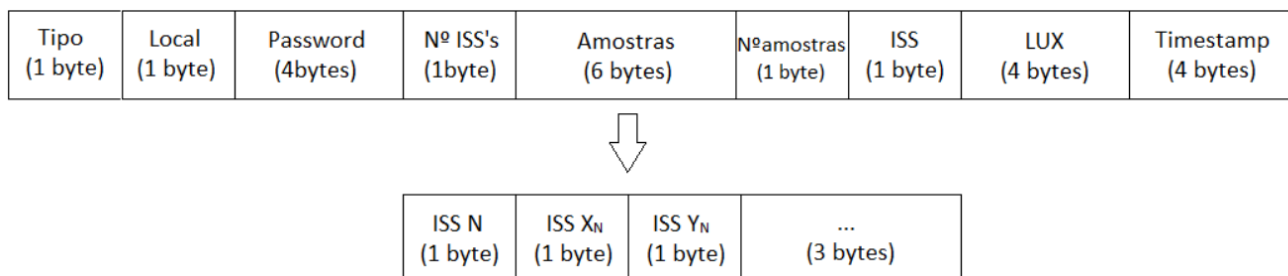


Figura 8 - Mensagem DATA.

- **DATA MOV** – A mensagem *DATA MOV* é enviada do concentrador para um gestor de serviços com os valores das amostras recolhidas relativas ao sensor de movimento de todos os sistemas sensores que estão conectados. Esta mensagem tem os campos apresentados na figura 9.

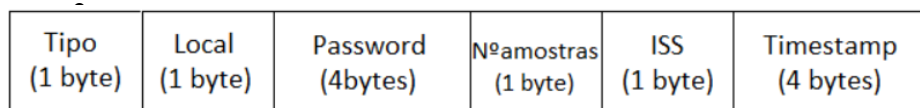


Figura 9 - Mensagem BEGIN.

- **END** – A mensagem *END* é enviada do sistema central para um concentrador a indicar que não está mais disponível para receber dados. Esta mensagem tem os campos apresentados na figura 10.



Figura 10 - Mensagem END.

3.2. Base de dados

De modo a armazenar todas as informações enviadas pelos vários concentradores, iremos elaborar uma base de dados para guardar todos esses dados numa forma mais eficiente e mais prática aos olhos do utilizador. O grupo decidiu implementar a base de dados em linguagem MySQL e o planeamento inicial da mesma num diagrama de entidades e relacionamentos é apresentado na figura seguinte.

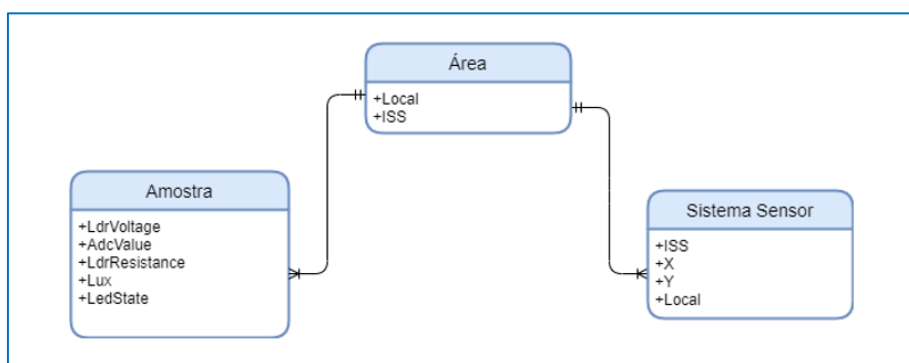


Figura 11 - Diagrama de Entidades e Relacionamentos.

Como nesta fase não foi necessária a elaboração de uma base de dados relacional optamos por simplificar utilizando ficheiros CSV. Na próxima fase iremos fazer a base de dados em MySQL como era pretendido inicialmente.

Relativamente aos ficheiros CSV foram criados 3 por parte do sistema central, um para armazenar as amostras enviadas pelos concentradores sobre o sensor de luminosidade, outro sobre o sensor de movimento e finalmente um com a informação relativa à localização dos vários sensores conectados a um dado concentrador.

Na figura seguinte apresentamos as amostras presentes nos vários ficheiros após receção por parte do sistema central.

The image shows a VS Code editor with two CSV files open. The left file, 'sistemaCentralDR.csv', contains 32 rows of data with columns for ISS, IDC, LUX, and TIMESTAMP. The right file, 'sistemaCentralMOV.csv', contains 22 rows of data with the same columns. The editor interface includes a sidebar with file explorer and a top bar with file names.

Figura 12 - Base de dados.

3.3. Interface Web

O sistema central que vai ser implementado numa interface Web terá como principais funções a gestão do armazenamento de dados recebidos pelos vários concentradores assim como controlar o começo e término destes começarem a fornecer dados. Este também terá como função apresentar uma interface simples para que o utilizador consiga visualizar vários aspetos das várias áreas como gráficos dos luxs que permitem retirar conclusões sobre a luminosidade num dado local e gráficos de movimentos que permitem saber quando houve movimento naquele local.

Para esta fase a implementação da interface ainda se encontra numa fase prematura pelo que foi só realizada em HTML uma forma de apresentar ao utilizador o conteúdo dos vários ficheiros da base de dados assim como gráficos simples que relacionam a quantidade de luxs de um dado sistema sensor com o *timestamp* de quando foi retirado a amostra.

Na figura 13 apresentamos o ecrã com o qual o utilizador se depara aquando do acesso à página web. Este acesso é feito localmente pelo que só é possível aceder através do *localhost* de quem corre o servidor. Neste ecrã o utilizador encontra 6 botões em que cada um fornece uma informação diferente. Os primeiros 3 (verdes) apresentam as várias tabelas das bases de dados como mostrado na próxima página e os últimos 3 apresentam gráficos que relacionam a luminosidade num dado local com a altura do dia.

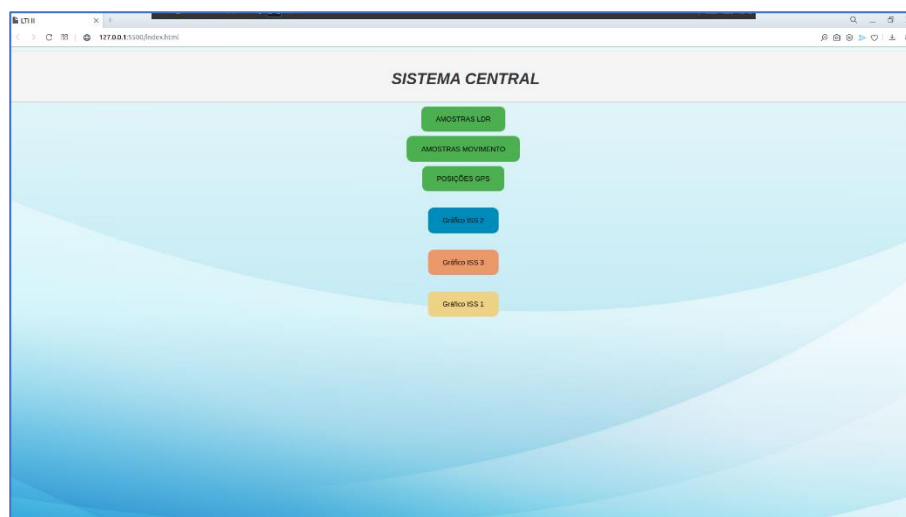
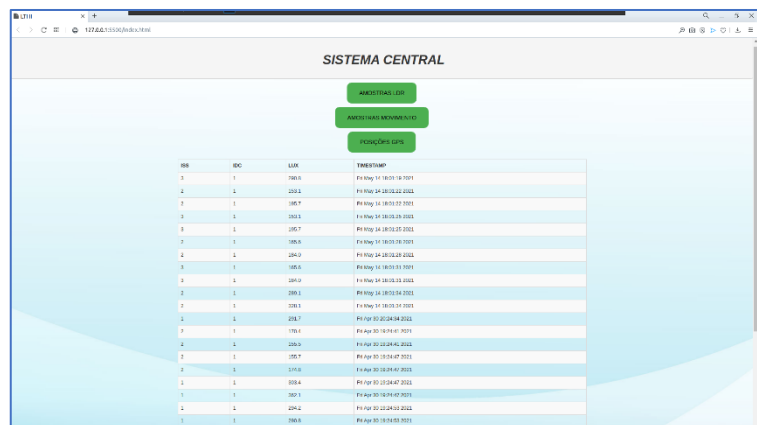


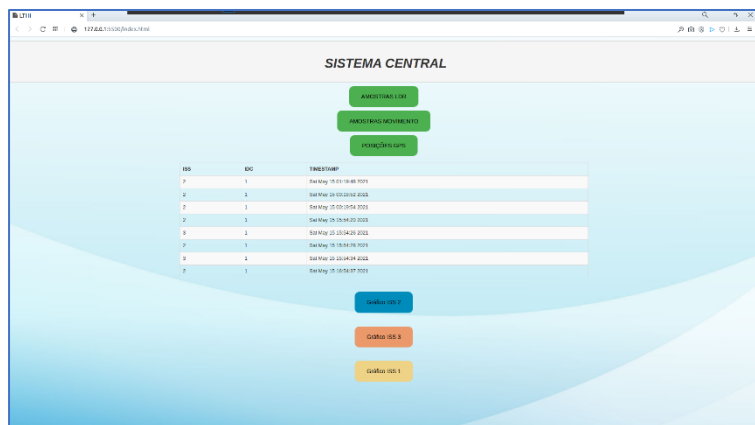
Figura 13 - Página inicial da interface web.

Nas figuras 15 e 14 são apresentados em interface web os conteúdos da base de dados relativos às amostras dos vários sensores de luminosidade e das amostras dos valores dos sensores de movimento. Na figura 16 é apresentado uma tabela relativa à base de dados com a posição de cada sistema sensor com conexão ao sistema central.



ID	IDC	LUM	TIMESTAMP
1	1	780.5	Fri May 14 16:01:18 2021
2	1	203.1	Fri May 14 16:02:42 2021
2	1	185.7	Fri May 14 16:03:12 2021
3	1	183.2	Fri May 14 16:03:15 2021
3	1	185.7	Fri May 14 16:03:25 2021
3	1	185.8	Fri May 14 16:03:38 2021
2	1	284.3	Fri May 14 16:03:48 2021
3	1	185.5	Fri May 14 16:03:53 2021
3	1	289.9	Fri May 14 16:03:58 2021
2	1	286.1	Fri May 14 16:03:58 2021
2	1	286.1	Fri May 14 16:03:58 2021
1	1	258.7	Fri May 14 16:04:58 2021
1	1	780.1	Fri May 14 16:05:05 2021
3	1	282.9	Fri May 14 16:05:06 2021
2	1	185.7	Fri May 14 16:05:47 2021
2	1	1416.0	Fri May 14 16:06:47 2021
1	1	338.4	Fri May 14 16:07:47 2021
1	1	267.1	Fri May 14 16:07:59 2021
1	1	284.2	Fri May 14 16:08:05 2021
1	1	780.8	Fri May 14 16:08:05 2021

Figura 15 - Tabela das amostras dos sensores LDR.



ID	IDC	TIMESTAMP
1	1	Sat May 15 01:16:46 2021
2	1	Sat May 15 01:03:02 2021
2	1	Sat May 15 01:05:04 2021
2	1	Sat May 15 01:04:20 2021
3	1	Sat May 15 01:04:20 2021
2	1	Sat May 15 01:04:16 2021
3	1	Sat May 15 01:04:14 2021
2	1	Sat May 15 01:04:07 2021

Figura 14 - Tabela das amostras dos sensores PIR.



ID	IDC	ISS_X	ISS_Y
1	1	7	6
2	1	6	6
3	1	5	5

Figura 16 - Tabela das posições dos sistemas sensores.

Para além de apresentarmos os valores das bases de dados também permitimos ao utilizador observar um gráfico da quantidade de luminosidade numa dada área ao longo do tempo, como demonstrado na figura 17.

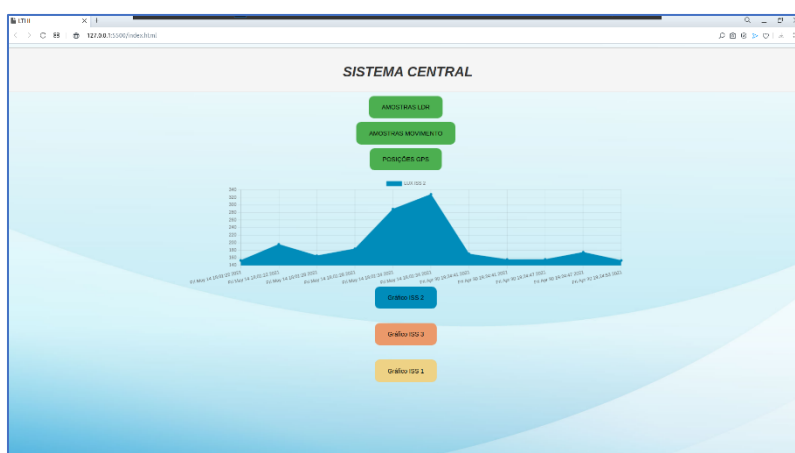


Figura 17 - Gráfico dos luxs.

4. Implementação

4.1. Sistema sensor simulado

No que toca ao sistema sensor simulado este foi bastante fácil de implementar visto tratar-se de uma cópia do sistema real já implementado na fase A. Como este sistema era simulado não era possível obter valores reais em tempo real dos sensores pelo que foram utilizadas amostras previamente retiradas por parte de um sensor real com o objetivo de obter amostras idênticas. A comunicação entre os sensores e o concentrador também sofreu uma pequena alteração pois teve de se usar o protocolo UDP o que levou a surgirem alguns problemas na viabilização da comunicação com mais de um sistema sensor [2].

4.1.1. Código

O código elaborado para o sistema sensor simulado foi realizado em C com o auxílio do ambiente de desenvolvimento Visual Studio Code. Como na fase A já foi descrito o código realizado para o sistema sensor em Arduino, são pequenas as mudanças relativamente ao simulado pelo que apenas iremos enunciar estas.

A primeira diferença é na forma da criação do *socket* para a comunicação com o concentrador apresentado no excerto de código da figura 18. Nesta função realizamos a abertura do *socket* e anunciamos o IP e porta a utilizar. A segunda diferença está na forma como recebemos e enviamos informação através do protocolo UDP que é utilizada uma biblioteca diferente do Arduino, a <sys/socket.h>, uma biblioteca do Linux.

```
void createSocket()
{
    sockfd = socket(AF_INET, SOCK_DGRAM, 0);
    if (sockfd == -1)
    {
        perror("socket:");
        exit(1);
    }
    printf("Socket criada com sucesso.\n");

    memset(&servaddr, 0, sizeof(servaddr));
    servaddr.sin_family = AF_INET;
    servaddr.sin_port = htons(SERV_PORT);

    struct timeval read_timeout;
    read_timeout.tv_sec = 0;
    read_timeout.tv_usec = 10;
    setsockopt(sockfd, SOL_SOCKET, SO_RCVTIMEO, &read_timeout, sizeof read_timeout);

    if ((servaddr.sin_addr.s_addr = inet_addr(host)) == INADDR_NONE)
    {
        perror("ine_addr:");
        exit(1);
    }
}
```

Figura 18 - Criação do socket.

Relativamente ao resto do código este é muito parecido com o realizado na fase A pelo que não achamos que haja necessidade de repetir o que foi dito.

4.2. Concentrador

Em relação à implementação do concentrador este já tinha sido implementado na fase anterior. No entanto, como queríamos realizar uma conexão ao sistema central tivemos de adicionar código para esta fase.

4.2.1. Código

Para esta fase foi necessária a utilização de um novo protocolo para a comunicação do concentrador, o protocolo TCP. A implementação deste no código é demonstrada na figura seguinte tendo em conta que o concentrador tem o papel de cliente e o sistema central o papel de servidor.

```
struct sockaddr_in servaddr;
char startp[7];

sockfd2 = socket(AF_INET, SOCK_STREAM, 0);
if (sockfd2 == -1)
{
    printf("socket creation failed...\n");
    exit(0);
}

if (fcntl(sockfd2, F_SETFL, fcntl(sockfd2, F_GETFL) | O_NONBLOCK) < 0)
{
    printf("fcntl error:");
    exit(0);
}

bzero(&servaddr, sizeof(servaddr));
servaddr.sin_family = AF_INET;
servaddr.sin_addr.s_addr = inet_addr("192.168.1.119");
servaddr.sin_port = htons(7778);
```

Figura 19 - Comunicação TCP entre concentrador e sistema central.

Na programação do concentrador da fase B foi necessária a implementação de uma nova *thread* para que o concentrador pudesse estar a receber dados dos sistemas sensores assim como enviar essas mesmas amostras para o sistema central. Esta implementação está demonstrada na figura 20 com a criação de 2 *threads* que correspondem ao menu fornecido ao utilizador e a função que permite comunicar com o sistema central.

```
pthread_t threadN[2];
if (pthread_create(&(threadN[0]), NULL, threadFunction, NULL) != 0)
{
    return 1;
}

if (pthread_create(&(threadN[1]), NULL, threadSistemaCentral, NULL) != 0)
{
    return 1;
}
```

Figura 20 - Criação das threads.

Esta *thread* está constantemente a enviar uma mensagem de *start* para o sistema central à espera de uma confirmação. Após receber esta confirmação, envia os valores das amostras relativos aos vários sensores conectados ao concentrador. Quando não houver mais amostras para enviar o concentrador envia uma mensagem de *stop* com o objetivo de informar o sistema central que não tem mais amostras para enviar.

4.2.2. Fluxograma

A figura seguinte contém o fluxograma que ajuda na visualização da solução encontrada por nós para a implementação do concentrador nesta fase.

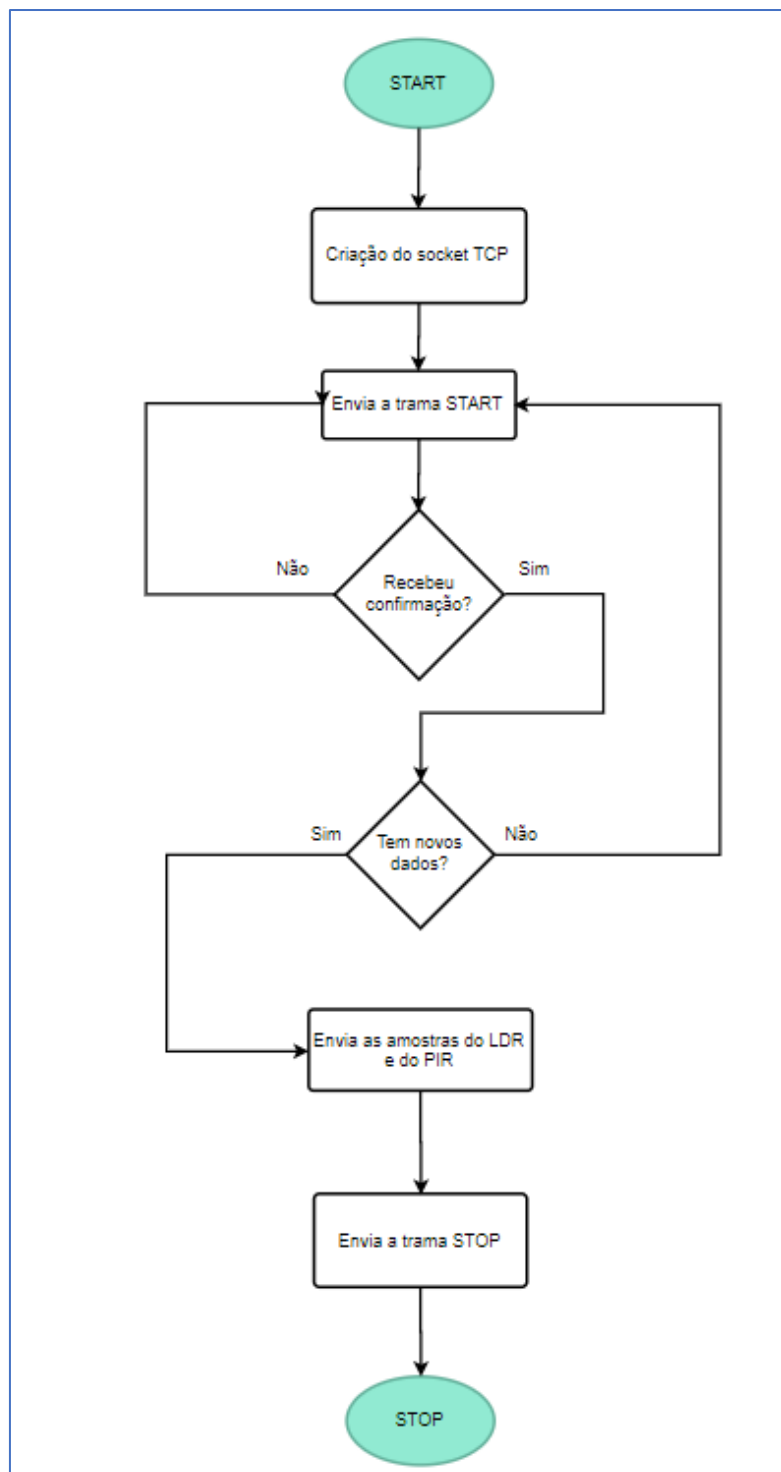


Figura 21 - Fluxograma do concentrador.

4.3. Gestor de Serviços

Como explicado anteriormente, o gestor de serviços tem como objetivo recolher os dados de todos os concentradores e enviar os mesmos para a base de dados, nesta fase a nossa base de dados consiste apenas num ficheiro CSV, no entanto para a próxima fase será necessário implementar uma base de dados relacional.

4.3.1. Código

No início do código é criado e aberto o *socket* para a comunicação com os diferentes concentradores. É criada uma variável *readfds*, que monitoriza os *files descriptors* dos concentradores e do socket principal. Sempre que um concentrador se conectar com o socket principal é-lhe atribuído um *file descriptor* e na próxima iteração será monitorizada a atividade do mesmo, através do *select()* (Figura 22) [3].

Quando é detetada atividade é sinal de que um concentrador se conectou e que pretende comunicar com o gestor de serviços. É verificado o endereço IP e *password* para certificar que o concentrador está apto a enviar dados, se sim, é apurado o tipo da trama. Consoante o tipo de trama o gestor de serviço tem diferentes comportamentos. Se receber uma trama do tipo *START*, o gestor envia para o concentrador uma confirmação para que este envie as amostras, caso receba uma trama do tipo *DATA*, este guarda as amostras no ficheiro a utilizar para a base de dados. Existem 2 tipos de trama *DATA*, a trama *DATA LDR* que envias as amostras relativas ao sensor de luminosidade e a trama *DATA MOV* que envias as amostras relativas ao sensor de movimento.

```
FD_ZERO(&readfds);
FD_SET(mainSocket, &readfds);
max_sd = mainSocket;

for (i = 0; i < maxConcentradores; i++)
{
    sd = concentradores[i];
    if (sd > 0)
        FD_SET(sd, &readfds);

    if (sd > max_sd)
        max_sd = sd;
}

activity = select(max_sd + 1, &readfds, NULL, NULL, NULL);

if ((activity < 0) && (errno != EINTR))
{
    printf("select error");
}

if (FD_ISSET(mainSocket, &readfds))
{
    if ((new_socket = accept(mainSocket,
        (struct sockaddr *)&address, (socklen_t *)&addrlen)) < 0)
    {
        perror("accept");
        exit(EXIT_FAILURE);
    }

    printf("Concentrador conectado com ip: %s\n", inet_ntoa(address.sin_addr));

    for (i = 0; i < maxConcentradores; i++)
    {
        if (concentradores[i] == 0)
        {
            concentradores[i] = new_socket;
            break;
        }
    }
}
```

Figura 22 - Código Gestor de serviço

4.3.2. Fluxograma

A figura seguinte contém o fluxograma que ajuda na visualização da solução encontrada por nós para a implementação do gestor de serviço nesta fase.

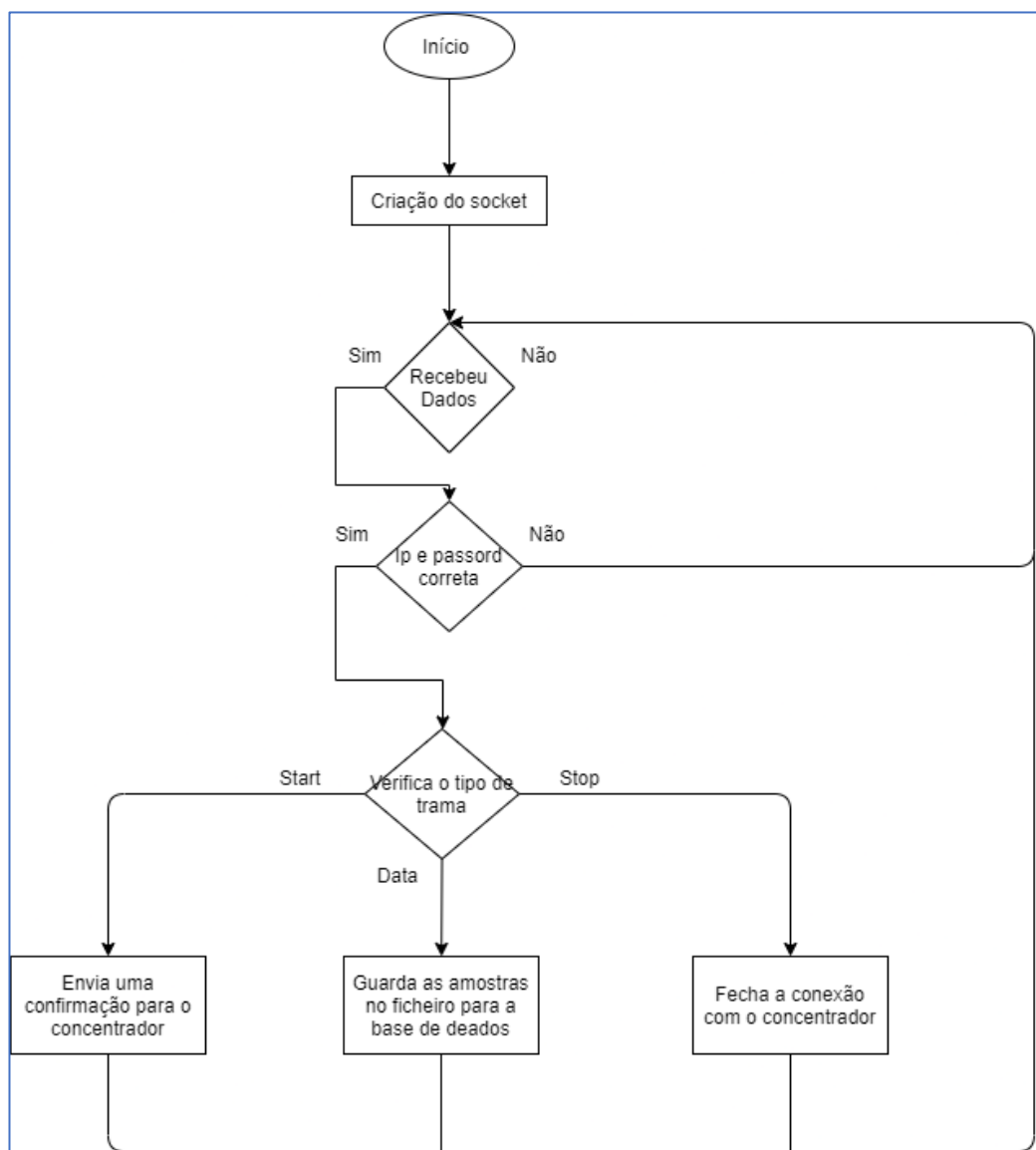
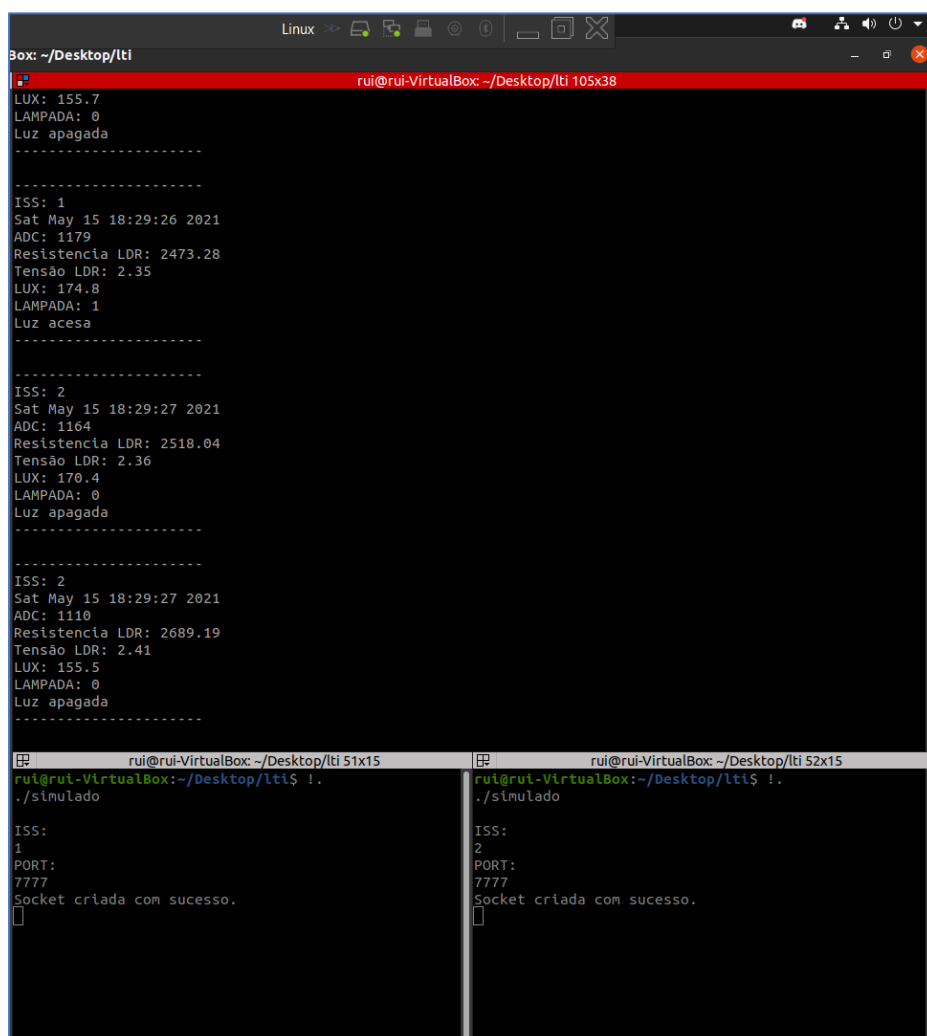


Figura 23 - Fluxograma Gestor de serviço

5. Testes e discussão de resultados

Após o desenvolvimento e implementação do sensor simulado, do concentrador e do sistema central tivemos de realizar vários testes de modo a garantir que tudo funcionava como expectável, ou seja, se o sistema central recebia os valores das amostras dos vários sensores simulados conectados aos concentradores e se guardava estes valores numa base de dados simples que era posteriormente utilizada pela interface web para apresentação dos valores.

Na figura seguinte apresentamos 2 sensores simulados a comunicar eficazmente com um concentrador passando tramas de dados enquanto este transmite estas amostras para o sistema central.



The image shows three terminal windows from a Linux environment. The top window, titled 'rui@rui-VirtualBox: ~/Desktop/lti 105x38', displays a series of sensor data readings separated by dashed lines. The data includes LUX, LAMPADA status, ISS (1 and 2), date/time, ADC, Resistencia LDR, Tensão LDR, and LUX values. The bottom left window, titled 'rui@rui-VirtualBox: ~/Desktop/lti 51x15', shows the execution of './simulado' and a log indicating 'Socket criada com sucesso.' for ISS 1 on PORT 7777. The bottom right window, titled 'rui@rui-VirtualBox: ~/Desktop/lti 52x15', shows the execution of './simulado' and a log indicating 'Socket criada com sucesso.' for ISS 2 on PORT 7777.

```
Box: ~/Desktop/lti
rui@rui-VirtualBox: ~/Desktop/lti 105x38
LUX: 155.7
LAMPADA: 0
Luz apagada
-----
ISS: 1
Sat May 15 18:29:26 2021
ADC: 1179
Resistencia LDR: 2473.28
Tensão LDR: 2.35
LUX: 174.8
LAMPADA: 1
Luz acesa
-----
ISS: 2
Sat May 15 18:29:27 2021
ADC: 1164
Resistencia LDR: 2518.04
Tensão LDR: 2.36
LUX: 170.4
LAMPADA: 0
Luz apagada
-----
ISS: 2
Sat May 15 18:29:27 2021
ADC: 1110
Resistencia LDR: 2689.19
Tensão LDR: 2.41
LUX: 155.5
LAMPADA: 0
Luz apagada
-----
rui@rui-VirtualBox: ~/Desktop/lti 51x15
rui@rui-VirtualBox: ~/Desktop/lti$ ./simulado
ISS:
1
PORT:
7777
Socket criada com sucesso.
[]

rui@rui-VirtualBox: ~/Desktop/lti 52x15
rui@rui-VirtualBox: ~/Desktop/lti$ ./simulado
ISS:
2
PORT:
7777
Socket criada com sucesso.
[]
```

Figura 24 - Comunicação entre 2 sistemas sensores simulados e um concentrador.

Após ser feita a transmissão do concentrador para o sistema central, este guarda os valores recebidos dos vários sistemas sensores na base de dados simples feita em formato csv presente na figura 25.

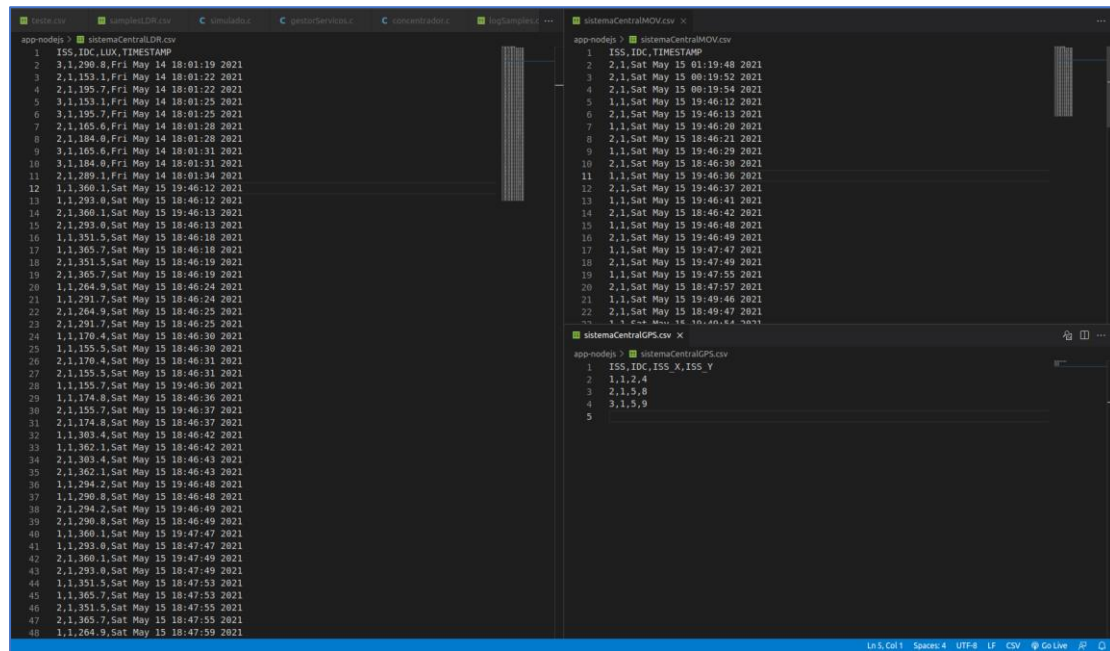


Figura 25 - Bases de dados.

Depois de guardar os valores na base de dados é efetuado uma leitura dos ficheiros por parte da interface web apresentando ao utilizador as várias tabelas de informação (Figura 26) assim como gráficos resultantes dessas informações.

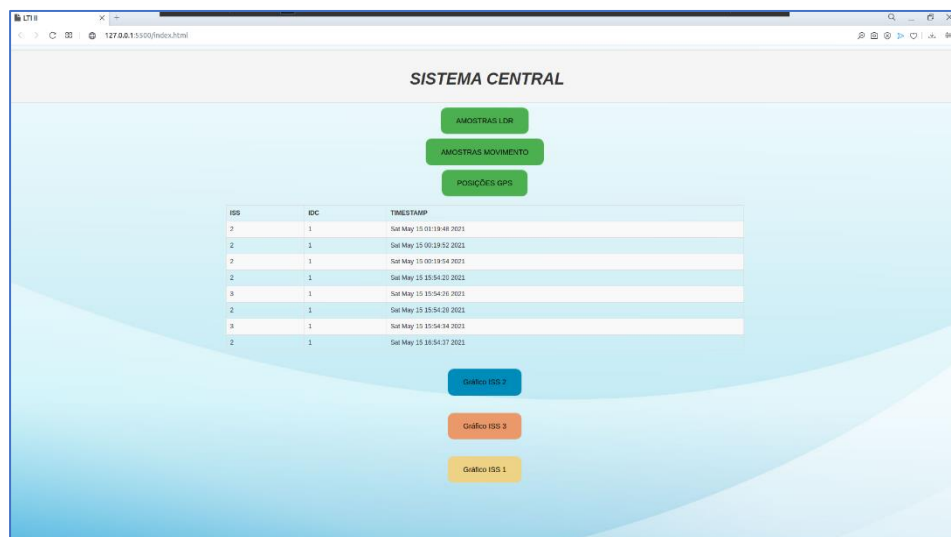


Figura 26 - Amostras dos sensores de movimento na interface web.

Achamos os resultados obtidos satisfatórios visto cumprirem com tudo o que era suposto, no entanto achamos que a interface com o utilizador poderia estar mais bem implementada e temos como objetivo melhorar na próxima fase.

Conclusão

Após a conclusão desta fase B podemos afirmar que nos sentimos satisfeitos com o resultado e com o nosso desempenho uma vez que cumprimos com os objetivos propostos. O facto de termos elaborado um planeamento antes da realização do projeto ajudou na coordenação e organização do que cada elemento do grupo tinha de fazer.

Relativamente às dificuldades que foram surgindo ao longo da realização desta fase estas foram ultrapassadas com sucesso apesar do desconhecimento inicial de alguns recursos e tecnologias como por exemplo no que toca a programação web.

A comunicação entre o sistema central e o concentrador que era a parte mais crucial desta fase foi implementada com sucesso assim como a transferência dos valores entre os sistemas sensores simulados e os concentradores. Começamos também pela implementação da interface web, mas esta ainda não ficou terminada. Na próxima fase temos por objetivo melhorar neste assunto assim como realizar uma base de dados relacional para o nosso projeto.

Referências

- [1] <https://www.muvi.com/wiki/udpuser-datagram-protocol.html>
- [2] <https://www.geeksforgeeks.org/udp-server-client-implementation-c/>
- [3] <https://www.geeksforgeeks.org/socket-programming-in-cc-handling-multiple-clients-on-server-without-multi-threading/>

Autoavaliação

Rui Filipe Ribeiro Freitas:

Nesta segunda fase do projeto ajudei no desenvolvimento do código do sensor simulado, do concentrador e do gestor de serviços assim como na pesquisa e elaboração do relatório.

Sandro Teixeira Ribeiro:

Nesta fase ajudei na pesquisa teórica, escrita e revisão do relatório.

Tiago João Pereira Ferreira:

Nesta fase ajudei no desenvolvimento do código do gestor de serviço e concentrador. Ajudei também na elaboração do relatório.