Universidade do Minho
Escola de Engenharia

METI - 2024/2025

Projeto Integrador em Telecomunicações e Informática

Relatório de Especificação - Fase C

Grupo 5

Fernando João Santos Mendes (PG55807)
Bruno Miguel Fernandes Araújo (PG55806)
Junlin Lu (A101270)

Índice

Lista de Acrónimos

LED - Light-Emitting Diode.
UART - Universal Asynchronous Receiver /Transmitter
CSV - Comma-Separated Values

Lista de Figuras

1. Introdução

A Fase C do projeto marca a conclusão do sistema de comunicação, com foco na camada de aplicação. Nesta fase, o objetivo principal é implementar uma aplicação útil e funcional para um ambiente de aeroporto, permitindo a consulta de informações em tempo real, bem como a transferência de ficheiros digitais, como revistas, mapas ou cartões de embarque.

2. Objetivos

A Fase C do projeto visa a implementação da camada de aplicação (nível 7) sobre o protocolo de transmissão fiável desenvolvido na Fase B, criando uma aplicação funcional voltada para o ambiente de um aeroporto. Os principais objetivos desta fase são::

1. Desenvolver e implementar um protocolo de aplicação (nível 7).
2. Conceber uma aplicação de consulta, que permita aceder a informações de voo (check-ins, horários, portões de embarque, etc.) e receber documentos digitais (revistas, mapas, cartões de embarque).
3. Assegurar que a aplicação permite o uso simultâneo de várias funcionalidades (consulta + transferência), através de multiplexagem.
4. Desenvolver uma interface de utilizador (gráfica ou textual)
5. Integrar o sistema completo (software + hardware) .

3.Arquitetura

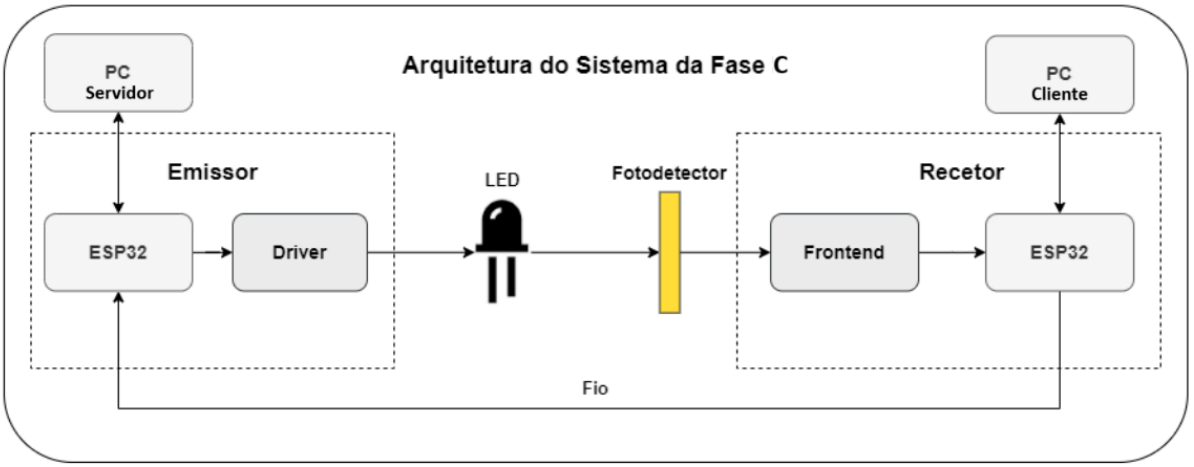

Figura 1: Arquitetura geral da Fase C

A Fase C do projeto tem como objetivo a implementação da camada de aplicação (nível 7), permitindo a criação de uma aplicação cliente-servidor com funcionalidades de consulta de informações e transferência de ficheiros no contexto de um aeroporto.

Nesta fase, o PC cliente envia pedidos formatados segundo um protocolo de aplicação para o PC servidor, que responde de acordo com o tipo de pedido (.consulta de voo ou envio de ficheiro). A camada 2 desenvolvida na fase anterior é reutilizada para garantir a transmissão fiável dos dados.

Fluxo de comunicação

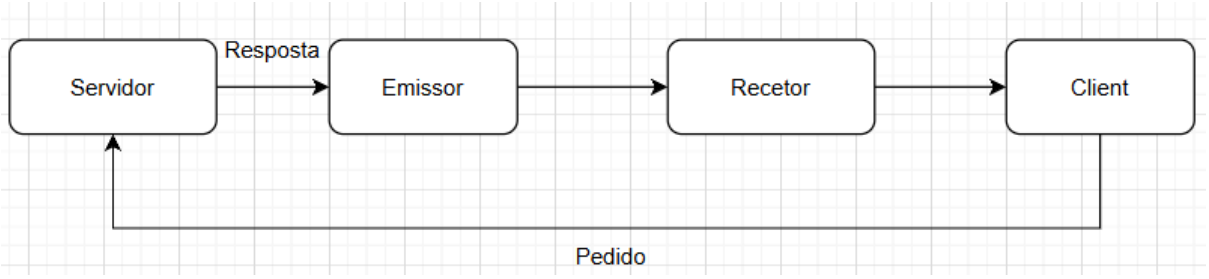

Figura 2: Fluxo de comunicação da Fase C

1. O utilizador no Cliente interage com a interface gráfica.

2. O pedido é enviado ao Servidor.

3. O Servidor processa o pedido e gera uma resposta (consulta ou envio de ficheiro).

4. Os pacotes da resposta são transmitidos pelo Emissor.

5. O Recetor capta o sinal e entrega os dados ao Cliente.

4. Protocolo de Camada de Aplicação

A. Definição das tramas

Na Fase C do projeto, o campo Payload passou a incorporar uma estrutura interna adicional que permite a multiplexagem de diferentes tipos de conteúdo, como dados de consulta e blocos de ficheiros, respeitando os requisitos da camada de aplicação e mantendo a separação das camadas inferiores.,

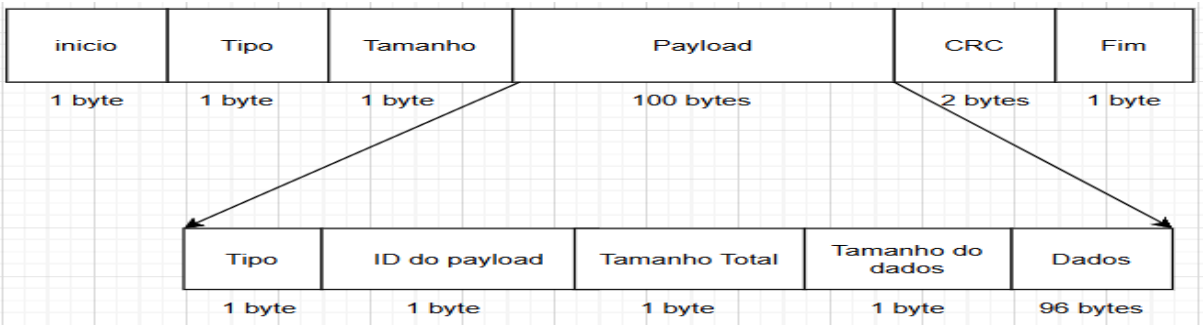

Figura 3: Estrutura da trama da camada ligação e aplicação

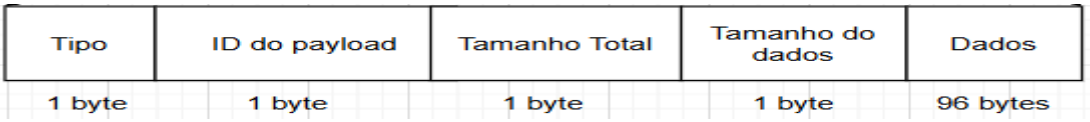

Figura 4: Estrutura do payload

Os seguintes campos da trama são definidos por:

- **Bloco Tipo**:
  Este bloco é responsável por definir o tipo das tramas que estão a ser enviadas.

- **Bloco do ID do Payload:**
  Este campo serve como um identificador único de cada fluxo de dados associado a um cliente.

- **Bloco Tamanho Total:**
  Indica o **tamanho total do ficheiro ou mensagem** completa que está a ser enviado. Esse campo só é útil no primeiro Payload de uma sequência ou em mensagens únicas. Ajuda o recetor a saber quando terminou a receção completa.

- **Bloco do Tamanho dos Dados**
  Contém o número de bytes efetivamente usados no campo Dados deste Payload.

- **Bloco Dados**
  É o conteúdo efetivo transportado neste segmento.

A decisão de definir o payload com 100 bytes surgiu após identificarmos uma limitação na transmissão de dados dos ficheiros através do link óptico, sempre que o valor ultrapassava os 100 bytes, os dados eram enviados de forma corrompida.

B. Multiplexagem

Para garantir que vários utilizadores possam utilizar simultaneamente o sistema — seja para consultar informações de voo, seja para transferir ficheiros digitais — a camada 7 foi desenvolvida com suporte a multiplexagem.

Do lado do emissor (servidor), o programa Python recorre a multithreading, criando uma nova thread para tratar cada pedido de cliente individualmente. Desta forma, é possível responder a diferentes utilizadores ao mesmo tempo, sem bloquear o sistema durante uma transferência de ficheiro, por exemplo.

Além disso, a multiplexagem é suportada na camada 7 através da inclusão de um ID de Payload em cada pacote. Este identificador funciona como um marcador único do fluxo de dados, permitindo ao sistema distinguir a que cliente ou a que pedido pertence cada pacote. Assim, mesmo com múltiplos pedidos a ocorrer em paralelo, é possível manter a organização e a integridade das sessões de comunicação.

5. Aplicações

A aplicação utilitária que será desenvolvida para um aeroporto permitirá a consulta de informações relacionadas com voos, bem como o download de documentos digitais, como mapas e trajetos.
A comunicação entre os dois sistemas será baseada num protocolo de camada 7 (camada de aplicação do modelo OSI), comum a ambos os computadores.

O computador emissor atuará como servidor, contendo localmente uma base de dados ou um ficheiro CSV (Comma-Separated Values) com as informações dos voos, assim como os ficheiros disponíveis para transferência. Esta base de dados/ficheiro CSV não será atualizada(o) dinamicamente pelo emissor, apenas será consultada.

A aplicação a nível do emissor incluirá uma interface gráfica simples destinada à monitorização dos logs de comunicação (mensagens enviadas e recebidas). Está prevista a expansão desta interface com funcionalidades adicionais, como a visualização dos ficheiros disponíveis para transferência ou a possibilidade de encerrar a aplicação de forma controlada.

O computador recetor funcionará como cliente, assumindo o papel de utilizador na comunicação. Este contará com uma interface gráfica que disponibilizará várias funcionalidades: consulta de informações sobre voos, transferência de ficheiros digitais disponíveis no servidor e envio de mensagens de erro (para fins de teste). Adicionalmente, a interface incluirá uma área de visualização dos logs de comunicação, permitindo ao utilizador acompanhar as mensagens trocadas entre si e o servidor.

Seria interessante desenvolver um pequeno programa para a simulação do funcionamento do aeroporto, que atualizasse aleatoriamente o ficheiro ou base de dados contendo as informações dos voos, com o objetivo de simular a introdução dinâmica de novos voos nos horários.

6. Ferramentas utilizadas

As ferramentas a níıvel do software que usamos as seguintes:

- Programa **Arduino IDE** , para o desenvolvimento de código para os **ESP32**.
- Programa **VisualStudio Code** , para o desenvolvimento de Python script e C + + script.
- Programa **Discord**, para a comunicação e partilha de ficheiros entre os membros do grupo.
- Programa **Tina**, para a simulação dos circuitos do emissor e do recetor.
- Programa **Excel**, para a criação do diagrama de Gantt.
- Plataforma **Draw.io**, para a criação do diagrama da arquitetura do sistema desta fase.
- Plataforma **Google Docs**, para o desenvolvimento deste relatório.

7. Planeamento Temporal

Apresentaremos a planificação temporal da Fase C do projeto através de um Diagrama de Gantt, que contém a lista completa de todas as tarefas e subtarefas desta fase, este encontra-se ilustrado na Figura 6.

TAREFAS	Início	Fim
Tratamento de mensagens com erros	4/21/25	4/25/25
Relatório de especificação da Fase Final	4/25/25	4/25/25
Construção da base dados/ficheiro csv com as informações para os utilizadores.	4/28/25	4/28/25
Desenvolvimento inicial da aplicação com a transferência de ficheiros digitais	4/28/25	5/9/25
Implentação da consulta de informações de voo	5/5/25	5/12/25
Interface de utilizador no recetor	5/12/25	5/14/25
Implementação de multiplexagem na aplicação	5/12/25	5/16/25
Interface no servidor com visualização de logs e outras funcionalidades.	5/16/25	5/19/25
Integrar o sistema completo de software com hardware	5/16/25	5/25/25
Desenvolvimento do relatório final	4/21/25	5/25/25
Entrega do código, relatório final e apresentação powerpoint	5/25/25	5/25/25
Demo e apresentação da Fase Final	5/26/25	5/26/25

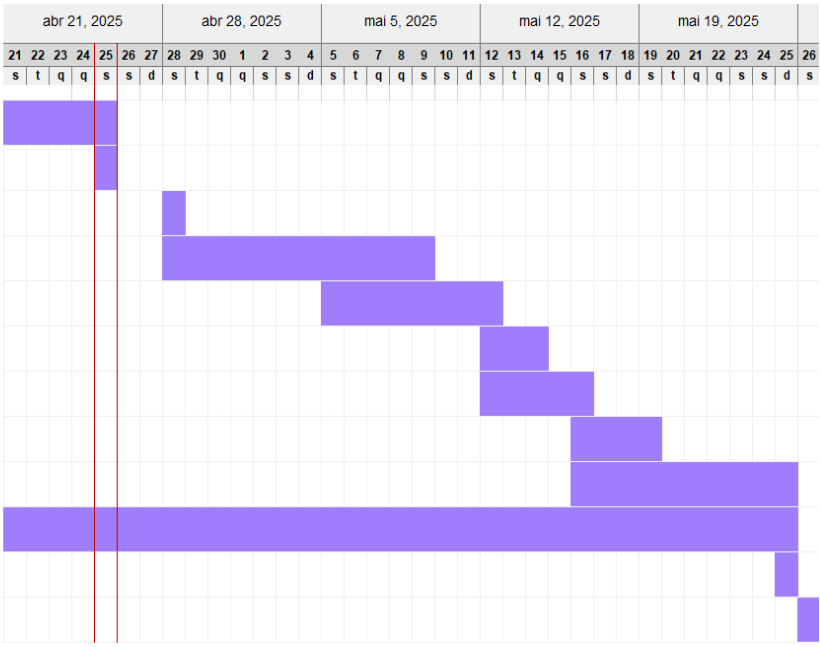

Figura 5: Diagrama de Gantt.

8. Referências

- Copperhill Technologies. (2021). *ESP32/ESP32S2 Serial Port (Native USB) Access Using Arduino IDE*. Copperhill Technologies. https://copperhilltech.com/blog/esp32-esp32s2-serial-port-native-usb-access-using-arduino-ide/

- GeeksforGeeks**.** (n.d.). *Stop and Wait ARQ*. GeeksforGeeks. https://www.geeksforgeeks.org/stop-and-wait-arq/

  Cloudflare. (n.d.). *O que é e como funciona a camada 7?* Cloudflare. https://www.cloudflare.com/pt-br/learning/ddos/what-is-layer-7/