

# Trabalho Prático 2

## Comunicações por Computador

Universidade do Minho  
Licenciatura em Engenharia Informática

Outubro 2025

### 1 Introdução

Nos últimos anos, a comunicação de dados tem-se tornado uma parte fundamental em diversos domínios, desde a gestão de redes de computadores até aplicações espaciais avançadas. Em particular, o desenvolvimento de protocolos de comunicação robustos e eficientes é crucial em ambientes onde a fiabilidade e o desempenho são imperativos. Este trabalho prático visa explorar e aprofundar o conhecimento dos alunos sobre esses aspectos, com foco específico na comunicação entre uma Nave-Mãe e rovers numa missão espacial simulada.

#### 1.1 Contexto da Missão

Imagine uma missão espacial onde uma Nave-Mãe, localizada em órbita de um planeta, deve coordenar e monitorizar múltiplos rovers que operam na superfície do planeta. A comunicação entre a Nave-Mãe e os rovers é vital para o sucesso da missão. A Nave-Mãe envia comandos e missões para os rovers e, por sua vez, os rovers enviam dados de monitorização contínuos e confirmações de execução de volta para a Nave-Mãe. Para simular este cenário, será usada uma topologia de rede que inclui uma Nave-Mãe, uma rede de satélites atuando como routers intermediários, e os rovers.

#### 1.2 Desafios de Comunicação

Numa missão espacial, a comunicação pode ser afetada por diversos fatores, como atrasos na rede, perda de pacotes e interferências. Neste cenário, a comunicação deve ser realizada de forma eficiente e confiável. Portanto, os protocolos de comunicação devem ser projetados para lidar com esses desafios. Em particular, o trabalho prático abordará os seguintes aspetos:

- **Comunicação Crítica via UDP (MissionLink - ML):** A comunicação crítica envolve o envio de missões da Nave-Mãe para os rovers e a receção de confirmações dos rovers. O UDP (User Datagram Protocol) será utilizado para esta comunicação, exigindo que os alunos implementem mecanismos adicionais para garantir a entrega e a integridade dos pacotes a nível aplicacional, dada a natureza não confiável do UDP.
- **Comunicação de Monitorização via TCP (TelemetryStream - TS):** Os dados de monitorização contínuos, como posições e estado dos rovers, serão enviados via TCP (Transmission Control Protocol). O TCP oferece garantias de entrega e ordem dos pacotes.
- **Interface de Observação (Ground Control):** Para além da comunicação entre Nave-Mãe e rovers, a Nave-Mãe deve expor uma **API de Observação**, acessível através de HTTP ou WebSockets/SSE, que disponibiliza informação sobre missões e telemetria em tempo real. Um novo nó da topologia, designado **Ground Control**, irá consumir esta API e apresentar os dados de forma legível. Este componente introduz desafios adicionais de desenho, tais como a definição de endpoints, a escolha entre comunicação por polling ou em tempo real, e a organização do estado a disponibilizar.

Num cenário real, o **MissionLink** (ML) utilizaria TCP para garantir a fiabilidade da comunicação crítica, enquanto o **TelemetryStream** (TS) poderia ser implementado com UDP para otimizar a transmissão de dados contínuos. No entanto, para este trabalho, o objetivo é que os alunos desenvolvam e explorem mecanismos de fiabilidade para um protocolo aplicacional que use UDP na camada de transporte, complementado com a experiência de desenvolver uma API de observação para integrar os dados de missão e telemetria num ponto central de visualização.

### 1.3 Objetivos do Trabalho

O trabalho prático tem como objetivos principais:

- **Desenvolver um protocolo aplicacional (MissionLink - ML)** sobre UDP para comunicação crítica que seja robusto e confiável, mesmo usando um protocolo de transporte que não garante a entrega.
- **Implementar um protocolo aplicacional (TelemetryStream - TS)** sobre TCP para comunicação de dados de monitorização, aproveitando as garantias de entrega e ordenação do TCP.
- **Configurar e simular** a topologia de rede utilizando o CORE, incluindo a Nave-Mãe, a malha de satélites e os rovers, para testar e avaliar os protocolos desenvolvidos.
- **Desenvolver uma API de Observação** na Nave-Mãe, acessível através de HTTP ou WebSockets/SSE, que disponibilize informação sobre missões e telemetria de forma estruturada.
- **Desenvolver uma aplicação Ground Control** capaz de consumir a API de Observação e apresentar os dados dos rovers e missões de forma legível, permitindo acompanhar a mobilidade e o estado da missão em tempo real.

Através deste trabalho, os alunos terão a oportunidade de aplicar conceitos teóricos num cenário prático e desafiador, desenvolvendo capacidades essenciais para a criação de soluções de comunicação eficazes em ambientes complexos, desde a conceção de protocolos fiáveis sobre UDP até à integração de mecanismos de visualização e monitorização.

## 2 Duração e Entregas

### 2.1 Duração do Trabalho

O trabalho prático tem a duração de 8 semanas. Durante este período, os alunos deverão desenvolver e testar os protocolos aplicacionais, configurar a topologia de rede no CORE e preparar a documentação necessária. O trabalho deve ser realizado em grupos de 3 alunos.

### 2.2 Documentação a Ser Submetida

Os alunos devem preparar e submeter os seguintes documentos e arquivos através da plataforma de e-learning:

- **Relatório Final:**
  - Descrição detalhada da implementação dos protocolos **MissionLink** e **TelemetryStream**, incluindo o design, arquitetura, formato das mensagens e mecanismos de controlo aplicados.
  - Especificação completa da **API de Observação** exposta pela Nave-Mãe, incluindo:
    - \* Endpoints definidos (HTTP, WebSockets ou SSE).
    - \* Estrutura das mensagens trocadas (exemplos de requests e responses).
  - Resultados dos testes realizados, incluindo cenários com diferentes níveis de latência e perda de pacotes, análise da robustez dos protocolos e comparação entre variantes experimentadas pelo grupo.

- Reflexão crítica sobre limitações encontradas e potenciais melhorias que poderiam ser exploradas.
- **Ambiente CORE Configurado:**
  - Arquivo de configuração da topologia de rede no CORE , incluindo a Nave-Mãe, a malha de satélites, os rovers e o nó **Ground Control**.
  - Descrição das configurações aplicadas (endereçamento, latência, perda, etc.) e scripts auxiliares utilizados.
- **Código Fonte:**
  - Todo o código desenvolvido para os protocolos aplicacionais, para a Nave-Mãe, para os rovers e para o nó **Ground Control**.
  - Instruções claras sobre como compilar, executar e testar cada componente.

### 2.3 Local de Submissão

Todos os documentos e arquivos devem ser submetidos através da plataforma de e-learning designada para a UC de Comunicações por Computador. Certifique-se de seguir as instruções específicas fornecidas na plataforma para garantir que todos os arquivos sejam enviados corretamente.

## 3 Definição de Missão para o MissionLink (ML)

Uma missão é uma ordem enviada pela Nave-Mãe, via ML, para um rover, com o objetivo de explorar uma área geográfica e executar uma tarefa definida. O protocolo **MissionLink** deve transportar, a nível aplicacional, toda a informação necessária para que o rover compreenda e execute a missão atribuída.

Cada grupo deve projetar o formato das mensagens de missão, respeitando os seguintes **requisitos mínimos**:

- **Identificação da Missão:** cada missão deve possuir um identificador único.
- **Área Geográfica:** a missão deve incluir informação sobre a área a explorar. A forma de representar esta área (coordenadas, polígonos, raio, etc.) é da responsabilidade do grupo.
- **Tarefa:** a missão deve especificar a ação a realizar. Podem ser adotadas tarefas como coleta de amostras, captura de imagens, análise ambiental ou outras propostas pelo grupo, desde que devidamente justificadas.
- **Duração:** deve ser indicado o tempo máximo permitido para a execução da missão.
- **Atualizações:** a missão deve definir como e com que frequência o rover reporta progresso e estado à Nave-Mãe. A métrica e unidade escolhida (segundos, minutos, número de eventos, etc.) é decisão do grupo.

Para além destes elementos obrigatórios, cada grupo é livre de acrescentar outros campos que considere relevantes (por exemplo: prioridade da missão, parâmetros ambientais a monitorizar, tolerância a falhas, instruções adicionais).

### 3.1 Exemplo ilustrativo de especificação de missão

A título de exemplo, uma missão poderia conter:

- **ID da Missão:** M-001
- **Área Geográfica:** região delimitada pelas coordenadas (x1, y1) a (x2, y2).
- **Tarefa:** Captura de imagens.
- **Duração:** 30 minutos.

- **Atualizações:** a cada 120 segundos.

Este exemplo serve apenas como referência: cada grupo deve propor, justificar e documentar o seu próprio formato de mensagens e as escolhas realizadas.

## 4 Dados de Monitorização - TelemetryStream (TS)

Os rovers enviados pela Nave-Mãe devem reportar dados de monitorização continuamente para garantir que estão a operar corretamente e dentro dos parâmetros. Estes dados serão enviados utilizando o protocolo **TelemetryStream (TS)**, que se baseia em comunicação via TCP. A Nave-Mãe será responsável por receber e processar estas informações em tempo real.

Cada grupo deve projetar o formato das mensagens de telemetria, respeitando os seguintes **requisitos mínimos**:

- **Identificação do Rover:** cada mensagem deve permitir identificar inequivocamente o rover que a enviou.
- **Posição:** a mensagem deve incluir alguma forma de localização (coordenadas, grid, vetor de deslocamento, etc.).
- **Estado Operacional:** a mensagem deve indicar o estado geral do rover (por exemplo: em missão, a caminho, parado, erro).

Para além destes requisitos mínimos, cada grupo é livre de definir e justificar outros parâmetros a monitorizar, tais como:

- **Nível de bateria** ou disponibilidade energética.
- Temperatura interna ou **indicadores de saúde dos sistemas**.
- **Velocidade e direção**.
- Estado das rodas, do sistema de locomoção ou de sensores.
- Outros parâmetros relevantes definidos pelo grupo (ex.: carga útil, condições ambientais).

### 4.1 Exemplo ilustrativo de dados de telemetria

A título de exemplo, uma mensagem de telemetria poderia conter:

- **ID do Rover:** R-002
- **Posição:** (x, y, z)
- **Estado Operacional:** em missão
- **Bateria:** 75%
- **Velocidade:** 1.5 m/s

Este exemplo é apenas ilustrativo. Cada grupo deve propor e documentar a sua própria estrutura de dados de telemetria, explicando no relatório as decisões tomadas quanto ao nível de detalhe, frequência de envio e formato das mensagens.

## 5 Comunicação entre o Rover e a Nave-Mãe

A comunicação entre os rovers e a Nave-Mãe será realizada através de dois protocolos distintos: **MissionLink (ML)** para o envio e gestão de missões, e **TelemetryStream (TS)** para a monitorização contínua dos rovers. A Nave-Mãe atua como servidor e os rovers como clientes.

### 5.1 Protocolo MissionLink (ML) - UDP

O **MissionLink (ML)** é o protocolo aplicacional responsável pela atribuição e execução das missões. Deve utilizar UDP como protocolo de transporte, exigindo que cada grupo desenvolva mecanismos adicionais de fiabilidade a nível aplicacional.

O desenho do protocolo deve garantir, no mínimo, os seguintes aspetos:

- O rover deve ser capaz de solicitar uma missão à Nave-Mãe.
- A Nave-Mãe deve responder com os parâmetros da missão, incluindo pelo menos: identificador, área de atuação, tarefa e duração.
- O rover deve reportar o progresso da missão de acordo com parâmetros definidos na própria missão.
- Devem ser implementados mecanismos de controlo que compensem as limitações do UDP, tais como números de sequência, confirmações de receção, retransmissão de pacotes ou outros mecanismos equivalentes.

Para além destes requisitos mínimos, cada grupo é livre de decidir a estrutura detalhada das mensagens, a frequência de envio de atualizações, os algoritmos de temporização e quaisquer extensões (ex.: prioridades de missão, níveis de fiabilidade, mensagens de cancelamento).

### 5.2 Protocolo TelemetryStream (TS) - TCP

O **TelemetryStream (TS)** é responsável pela transmissão contínua de dados de monitorização do rover para a Nave-Mãe, utilizando TCP como protocolo de transporte.

O protocolo deve garantir, no mínimo:

- Estabelecimento de uma ligação confiável entre cada rover e a Nave-Mãe.
- Envio periódico de dados de monitorização, incluindo pelo menos: identificador do rover, posição e estado operacional.
- Capacidade de lidar com múltiplos rovers em paralelo, assegurando que a Nave-Mãe recebe e organiza corretamente os dados.

Cada grupo pode definir livremente:

- Quais os parâmetros adicionais incluídos nos pacotes de telemetria.
- A granularidade temporal do envio (intervalos fixos, adaptativos, eventos).
- Estratégias de codificação e formatação das mensagens.

## 6 Interface de Observação e Ground Control

Para além da comunicação entre a Nave-Mãe e os rovers via **MissionLink (ML)** e **TelemetryStream (TS)**, cada grupo deve implementar um componente adicional para disponibilizar e visualizar a informação da missão:

### 6.1 API de Observação na Nave-Mãe

A Nave-Mãe deve expor uma **API de Observação**, que disponibiliza informação sobre as missões e o estado dos rovers. Esta API deve ser acessível por outros nós da rede e permitir que o estado atual do sistema seja consultado em tempo real ou próximo do tempo real.

- A API deve disponibilizar, no mínimo:
  - Lista de rovers ativos e respetivo estado atual.
  - Lista de missões (ativas e concluídas), incluindo parâmetros principais.

- Últimos dados de telemetria recebidos pela Nave-Mãe.
- A API pode ser implementada através de HTTP (REST) ou de comunicação em tempo real (WebSockets ou Server-Sent Events).
- O formato das mensagens (ex.: JSON, XML, texto) é de escolha do grupo, devendo ser claramente especificado e documentado.

Cada grupo deve incluir no relatório uma descrição completa da API: endpoints ou canais disponibilizados, formato das mensagens e justificação das decisões de design.

## 6.2 Nó Ground Control

Deve ser adicionado à topologia um nó **Ground Control**, ligado à Nave-Mãe. Este nó atua como cliente da API de Observação e deve permitir ao utilizador acompanhar a operação da missão.

- O cliente deve consumir a API da Nave-Mãe e apresentar de forma legível:
  - Os rovers em operação e o respetivo estado.
  - As missões atribuídas e o seu progresso.
  - Os valores de telemetria mais recentes.
- A forma de apresentação (linha de comandos, interface gráfica simples, aplicação web) é livre, desde que funcional e documentada.
- Opcionalmente, o grupo pode incluir na interface a representação gráfica da mobilidade dos rovers ao longo da missão. A decisão de integrar esta funcionalidade e a forma de o fazer (ex.: mapa, grelha, histórico de posições) é inteiramente responsabilidade do grupo.

O relatório deve incluir instruções sobre como executar o nó **Ground Control**, bem como exemplos de utilização (prints, logs ou capturas de ecrã).

**Importante:** Este componente complementa a experiência com protocolos de transporte, permitindo aplicar também conceitos de design de interfaces e integração de sistemas. No entanto, o foco principal deste trabalho prático continua a ser o **desenvolvimento e validação dos protocolos aplicativos**, implementação fiável do **MissionLink (UDP)** e a comunicação contínua do **TelemetryStream (TCP)**. A *API de Observação* e o *Ground Control* devem ser vistos como ferramentas de suporte à visualização e teste das comunicações.

## 7 Ambiente de Testes e Topologia

Os alunos deverão configurar e testar o sistema no ambiente de emulação CORE, que será utilizado para simular a topologia de rede entre a Nave-Mãe e os rovers, bem como qualquer infraestrutura intermediária, como satélites. A topologia deve refletir as condições do cenário espacial proposto, mas cada grupo tem liberdade para decidir a organização detalhada da rede, desde que respeite os requisitos mínimos definidos abaixo.

**Importante:** A topologia configurada no ambiente CORE é totalmente estática. Embora os rovers simulem o seu movimento na aplicação, com base nas missões recebidas, eles serão representados como **endpoints** fixos (nós estáticos no CORE). Não é necessário, nem expectável, implementar scripts de mobilidade no CORE para mover fisicamente os rovers no ambiente de emulação. O movimento dos rovers será simulado apenas a nível da aplicação, controlado pelo código desenvolvido pelos alunos. Embora a API do CORE permita controlar a mobilidade dos nós, esse não é o foco deste trabalho, dado o nível de complexidade adicional envolvido.

**Nota sobre as Comunicações Espaciais:** Embora, em cenários reais de missões espaciais, as comunicações sejam realizadas através de sinais de rádio ou tecnologia de comunicação por laser, neste trabalho iremos abstrair essa complexidade técnica. Para simplificar a implementação e focar nos conceitos de protocolos aplicativos e camadas de transporte, todas as comunicações entre a Nave-Mãe, os rovers e os satélites serão realizadas através de ligações com fios, usando o protocolo Ethernet (IEEE 802.3).

### 7.1 Requisitos Mínimos de Topologia

- Deve existir uma **Nave-Mãe**, configurada como servidor para os protocolos **MissionLink (ML)** via UDP e **TelemetryStream (TS)** via TCP, e como fornecedor da **API de Observação**.
- Devem existir pelo menos dois **Rovers**, configurados como clientes, capazes de solicitar missões, executar missões simuladas e enviar dados de telemetria.
- Deve existir pelo menos uma rede de **Satélites/routers intermediários** entre a Nave-Mãe e os rovers, de modo a simular comunicação multi-salto. O número e organização dos satélites é uma decisão do grupo (ex.: estrela, malha, árvore).
- Deve existir um nó adicional de **Ground Control**, que consome a API de Observação da Nave-Mãe e apresenta informação sobre missões, telemetria e estado dos rovers.

### 7.2 Decisões de Design dos Grupos

Para além dos requisitos mínimos, cada grupo pode (e deve) tomar decisões sobre:

- A topologia detalhada (número de satélites, ligações, estrutura).
- A configuração de latência, perda e variação de atraso nos links, para simular condições adversas de comunicação.
- A frequência de envio de telemetria e atualizações de missão.
- A forma de apresentar a mobilidade dos rovers no **Ground Control** (opcional), caso o grupo decida incluir esta funcionalidade.

### 7.3 Testes a Realizar

O sistema deverá ser testado sob diferentes condições de rede, como:

- Cenários com latência e perda de pacotes configurados em diferentes níveis.
- Situações com múltiplos rovers em paralelo, assegurando que a Nave-Mãe consegue processar missões e telemetria em simultâneo.
- Casos em que a fiabilidade do MissionLink (sobre UDP) é posta à prova, avaliando retransmissões e outros mecanismos implementados.

## 8 Plano de Desenvolvimento

O trabalho tem a duração de 8 semanas. Cada grupo deve organizar o seu plano de trabalho de forma autónoma, garantindo que ao longo do período todas as fases principais abaixo são cumpridas. O cronograma interno pode ser definido livremente pelo grupo, desde que as entregas finais incluam todos os componentes exigidos.

### Fases de Desenvolvimento

- **Fase 1 – Preparação e Ambiente:** Estudo detalhado do enunciado, instalação das ferramentas necessárias e configuração inicial do ambiente de emulação no CORE.

- **Fase 2 – Design dos Protocolos:** Definição do protocolo **MissionLink (ML)** sobre UDP e do protocolo **TelemetryStream (TS)** sobre TCP, incluindo formatos de mensagem, mecanismos de controlo, diagrama de sequencia e troca de mensagens. Definição da **API de Observação**, especificando endpoints/canais e formato de dados.
- **Fase 3 – Implementação do MissionLink (ML):** Desenvolvimento da comunicação entre a Nave-Mãe e os rovers para atribuição e execução de missões, incluindo mecanismos de fiabilidade a nível aplicacional.
- **Fase 4 – Implementação do TelemetryStream (TS):** Desenvolvimento da comunicação contínua de telemetria dos rovers para a Nave-Mãe, assegurando gestão simultânea de múltiplos rovers.
- **Fase 5 – Implementação da API de Observação e Ground Control:** Implementação da API na Nave-Mãe e do cliente **Ground Control**, capaz de apresentar informação sobre missões, telemetria e estado dos rovers.
- **Fase 6 – Integração e Testes:** Integração de todos os componentes com a topologia no CORE. Testes em diferentes cenários de rede (latência, perda, múltiplos rovers). Validação da robustez dos protocolos e da API de Observação.
- **Fase 7 – Documentação e Revisão Final:** Consolidação da documentação e do relatório final, incluindo descrição dos protocolos, API e decisões de design, bem como resultados dos testes. Preparação da submissão final.