



## **Relatório - Entrega 3**

**Alunos:** Felipe Lopes Gibin Duarte (231025207)  
Artur Kohara Guerra (231025181)

### **1) Objetivos da terceira entrega**

A terceira entrega tem como objetivos a otimização, validação experimental e consolidação do sistema de monitoramento ambiental via LoRa. Nesta fase, o foco deslocou-se da simples implementação funcional para a eficiência energética e a confiabilidade operacional do sistema.

### **2) Implementação de estratégias para redução de consumo**

A primeira estratégia foi substituir o modelo tradicional de laço infinito com `loop()` e `delay()` por uma arquitetura orientada a ciclos curtos de atividade seguidos de sono profundo (deep sleep). Em cada ciclo, o ESP32S3 acorda, lê as medições, decide se deve transmitir com base em thresholds (limiares) e entra imediatamente em deep sleep, usando apenas o RTC interno para manter o tempo e algumas variáveis na memória. Essa abordagem reduz drasticamente o tempo em que a CPU permanece ativa e maximiza o tempo em deep sleep, o que contribui para uma economia de energia significativa.

A segunda estratégia foi a amostragem adaptativa. Em vez de transmitir em todos os ciclos, o nó compara os valores atuais de temperatura, umidade e poeira com os últimos valores enviados, armazenados em variáveis. Só há transmissão se a variação superar limites pré-definidos (por exemplo, uma diferença de 0,5 °C em temperatura, 2% em umidade, 10 unidades em poeira) ou quando um heartbeat periódico (mandado de tempos em tempos para sinalizar que

o nó sensor ainda está funcionando) é atingido após um certo número de ciclos. Isso reduz significativamente o número de transmissões LoRa, que são a parte mais cara em energia do ciclo, mantendo ainda assim a qualidade das informações ambientais reportadas.

No subsistema de transmissão, foi adotado um payload binário compacto em vez das strings de texto utilizadas na entrega 2. Os valores de temperatura, umidade, poeira, bateria e número de sequência são codificados em inteiros escalonados (por exemplo, temperatura  $\times 100$  em `int16_t`), resultando em um pacote de 10 bytes. Pacotes menores significam menos símbolos LoRa transmitidos, reduzindo o tempo em rádio e, conseqüentemente, a energia consumida em cada envio. Após a transmissão, o firmware chama explicitamente `radio.sleep()` no SX1262 (módulo LoRa), colocando em modo de baixo consumo.

### **3) Atraso médio**

O intervalo entre 2 leituras no dashboard nunca é igual ao intervalo entre 2 envios do cliente-sensor. Entre o envio e a atualização do dashboard, há um atraso que corresponde à soma da latência de transmissão via LoRa, tempo de inicialização da ESP32 após sair do deep sleep, tempo de processamento pelo gateway, tempo de envio e processamento até o servidor/backend, e pequenos atrasos referentes ao socket e aos métodos HTTP.

Levando isso em conta, é possível observar que embora a ESP32 esteja configurada para permanecer no modo deep sleep por 5 segundos no exemplo do vídeo, a informação coletada pelo cliente-sensor só aparece no dashboard 7 segundos depois. Portanto, a latência média do sistema como um todo é aproximadamente de dois segundos por leitura, o que corresponde com a expectativa para sistemas de comunicação baseados em LoRa com processamento distribuído.

### **4) Perda de pacotes**

Para avaliar a conexão da rede sem fio, o projeto adotou uma estratégia baseada em números de sequência embutidos no payload binário. Cada vez que o nó sensor (cliente) transmite um novo pacote de dados, ele incrementa um contador interno de 16 bits que é anexado ao final dos dados enviados. É importante ressaltar que essa variável é armazenada na memória RTC do microcontrolador ESP32, marcada com o atributo `RTC_DATA_ATTR`. Isso garante que o contador persista e não seja zerado durante os ciclos de Deep Sleep, permitindo uma contagem contínua.

Dessa forma, foi possível avaliar se houve perdas de pacote durante a transmissão por meio da visualização do número de sequência dos pacotes apresentados no dashboard. Entretanto, não foi implementada nenhuma técnica para explicitamente tratar perdas de pacote, como retransmissões. Essa escolha foi feita sobretudo com o intuito de economizar ao máximo o gasto energético no nó transmissor, visto que a perda de pacotes não é um problema crítico no sistema.

Contudo, em contextos em que perdas não podem ser toleradas, é preciso que haja um mecanismo completo de comunicação confiável, conforme estudado nos princípios de Reliable Data Transfer (RDT), envolvendo detecção, confirmação e retransmissão de pacotes.

## **5) Consumo de energia estimado**

Para estimar o consumo de energia do nó sensor, partimos da sequência de operação implementada: o ESP32 acorda do deep sleep, gera (ou lê, em caso em que sensores estão disponíveis) os valores de temperatura, umidade e poeira, decide se deve transmitir, realiza a transmissão LoRa quando necessário e, ao final do ciclo, volta a entrar em sono profundo por 5 segundos, conforme configurado no código.

Desse modo, cada ciclo é composto por um período curto de atividade, seguido por 5 segundos em deep sleep. A partir dos valores de corrente encontrados nos datasheets do ESP32 e do módulo SX1262, consideramos que, durante o período ativo, o conjunto ESP32 + rádio SX1262 consome na ordem de 100–110 mA (somando o microcontrolador em operação e o rádio em modo de transmissão), enquanto no deep sleep a corrente cai para a ordem de 0,1 mA,

permanecendo apenas o RTC ligado. Assumindo, de forma conservadora, que o nó fica aproximadamente 0,5s ativo e 5s em deep sleep por ciclo, obtém-se uma corrente média em torno de 10–11 mA.

Com uma bateria de 2000 mAh, a autonomia teórica, no cenário extremo em que há transmissão em todos os ciclos de 5 segundos, fica na faixa de 7 a 8 dias. Na prática, como o nó sensor utiliza amostragem adaptativa e só transmite quando há variações significativas ou em heartbeats periódicos, o tempo em que o rádio permanece em TX é menor, o que reduz ainda mais a corrente média, aumentando o tempo de autonomia do sistema.

Vale ressaltar que em aplicações reais desse sistema, o tempo de deep sleep seria muito maior, a fim de conservar ainda mais energia, de forma a verificar os dados em períodos de tempo mais usuais.

## **6) Melhoria do dashboard**

Uma das principais melhorias no dashboard foi a inclusão de colunas dedicadas às métricas de qualidade de conexão (RSSI e SNR). Para tornar esses dados técnicos mais compreensíveis, foi implementada uma lógica de coloração condicional:

- Valores de RSSI acima de -85 dBm são exibidos em verde, indicando conexão robusta.
- Valores entre -85 dBm e -100 dBm aparecem em laranja, sinalizando uma conexão funcional mas que exige atenção.
- Valores abaixo de -100 dBm são destacados em vermelho, alertando para um sinal com um maior risco de perda de pacotes.

Essa visualização permite que o operador identifique rapidamente se um nó sensor está mal posicionado ou sofrendo interferência, sem a necessidade de analisar arquivos de log eventualmente mais complexos.

Para fazer um monitoramento de segurança, foi implementado um sistema de alertas visuais para condições ambientais consideradas extremas. O código do dashboard agora analisa dinamicamente cada leitura de temperatura recebida. Caso o valor ultrapasse o limiar de segurança configurado (40°C), a célula correspondente na tabela é renderizada com fundo vermelho. Essa mudança visual garante que

situações de risco, como superaquecimento de equipamentos ou início de incêndio, não passem despercebidas, podendo ser discernida imediatamente das leituras normais do sistema.

Por fim, a adição da coluna "Seq #" (Sequence Number) ao dashboard permite uma checagem visual rápida da continuidade dos dados. O usuário pode verificar, olhando para a tabela, se os números de sequência estão incrementando linearmente ou se há saltos, o que indicaria visualmente a perda de pacotes ocorrida naquele intervalo de tempo.

## 7) Demonstração da comunicação real

Todos os casos de teste se referem à mesma sequência de envio de pacotes, que se baseia em variações manuais dos parâmetros ambientais simulados. A falta de disponibilidade de sensores reais forçou a criação de uma sequência de envio manipulada explicitamente a fim de testar as funcionalidades gerais do sistema, incluindo as estratégias de economia de energia e as melhorias do dashboard. A sequência é a seguinte:

1. **Boots 0 a 2:** a temperatura inicia em 25,0 °C e aumenta em incrementos de 0,1 °C a cada ciclo (25,0 → 25,1 → 25,2). Umidade, poeira e bateria permanecem fixas em 60%, 100 unidades e 95%. O primeiro boot é enviado obrigatoriamente, porém os boots 1 e 2 não são, visto que os limiares discutidos não são ultrapassados.
2. **Boot 3:** ocorre um salto de temperatura para 26,0 °C, o que ultrapassa o limiar de temperatura de 0.5 graus definido e há o envio da leitura.
3. **Boots 4 e 5:** a temperatura se estabiliza em 26,1 °C, representando uma condição quase constante. Não há envio dos pacotes.
4. **Boot 6:** a temperatura varia levemente novamente, indo para 26,2 °C. Note que embora não haja variações nos parâmetros acima dos limiares estabelecidos, a leitura ainda é enviada via heartbeat, que envia todas as leituras com boot múltiplo de 6.

5. **Boot 7:** mantém-se a última temperatura, mas a umidade aumenta para 65%, o que aciona o mecanismo de envio.
6. **Boot 8:** a concentração de poeira sobe de 100 para 120 unidades, simulando um evento de piora na qualidade do ar que ultrapassa o limiar definido e envia a leitura.
7. **Boot 9:** há uma queda brusca de temperatura para 24,0 °C. Há o envio da leitura.
8. **Boot 10:** uma variação extrema é simulada com temperatura de 50.5 °C, o que permite testar a visualização no dashboard da medida de segurança discutida na seção 6.
9. **Boots posteriores:** os valores retornam ao padrão base (25 °C, 60%, 100 un), garantindo uma referência estável após os testes. A partir daqui, somente o heartbeat será enviado, a fim de garantir um monitoramento mínimo a cada 6 boots.

#### Monitoramento Ambiental & Rede LoRa

ID N6	Timestamp	Temp (°C)	Umid (%)	Poeira	Bat (%)	Seq #	RSSI (dBm)	SNR (dB)
sala_B_02	10:41:16	25.0	60.0	100.0	95.0	10	-69.0	8.75
sala_B_02	10:40:34	25.0	60.0	100.0	95.0	9	-71.0	9.0
sala_B_02	10:39:52	25.0	60.0	100.0	95.0	8	-71.0	8.5
sala_B_02	10:39:45	25.0	60.0	100.0	95.0	7	-72.0	9.25
sala_B_02	10:39:38	50.5	60.0	100.0	95.0	6	-77.0	9.0
sala_B_02	10:39:31	24.0	60.0	100.0	95.0	5	-80.0	8.75
sala_B_02	10:39:23	25.0	60.0	120.0	95.0	4	-80.0	9.25
sala_B_02	10:39:16	25.0	65.0	100.0	95.0	3	-80.0	8.0
sala_B_02	10:39:09	26.2	60.0	100.0	95.0	2	-76.0	8.5
sala_B_02	10:38:48	26.0	60.0	100.0	95.0	1	-73.0	8.75
sala_B_02	10:38:27	25.0	60.0	100.0	95.0	0	-69.0	8.25

Dashboard resultante do primeiro caso de teste.

No primeiro caso de teste do sistema, a comunicação foi feita com uma distância pequena entre os nós LoRa. O intuito deste primeiro teste era minimizar a interferência de fatores externos (distância e barreiras físicas) para verificar apenas o correto funcionamento do sistema desenvolvido e suas funcionalidades.

A imagem acima apresenta o dashboard que foi obtido neste teste. Pode-se, primeiramente, observar que a sequência de dados foi corretamente transmitida, sendo que:

- O pacote 0 representa o boot 0, o qual é a primeira medida feita, logo é sempre enviada (os boots 1 e 2 não são

transmitidos, pois a mudança de temperatura é inferior ao limiar definido);

- O pacote 1 representa o boot 3, o qual é enviado, pois a temperatura aumenta uma quantidade superior ao limiar definido (os boots 4 e 5 não são enviados, pois não ultrapassam o limiar);
- O pacote 2 representa o boot 6, o qual é enviado por ser um heartbeat, indicando que o nó sensor está funcionando;
- O pacote 3 representa o boot 7, o qual é enviado devido a um aumento na umidade superior ao limiar;
- O pacote 4 representa o boot 8, o qual é enviado devido a um aumento no nível de poeira superior ao limiar;
- O pacote 5 representa o boot 9, o qual é enviado, pois a temperatura reduz uma quantidade que ultrapassa o limiar;
- O pacote 6 representa o boot 10, o qual é enviado, pois a temperatura aumenta drasticamente;
- Os pacotes seguintes representam os boots posteriores, os quais são enviados como heartbeat.

#### Monitoramento Ambiental & Rede LoRa

ID Nó	Timestamp	Temp (°C)	Umid (%)	Poeira	Bat (%)	Seq #	RSSI (dBm)	SNR (dB)
sala_B_02	11:08:03	25.0	60.0	100.0	95.0	9	-114.0	0.25
sala_B_02	11:07:19	25.0	60.0	100.0	95.0	8	-116.0	-2.25
sala_B_02	11:07:12	25.0	60.0	100.0	95.0	7	-114.0	0.25
sala_B_02	11:07:04	50.5	60.0	100.0	95.0	6	-113.0	1.25
sala_B_02	11:06:56	24.0	60.0	100.0	95.0	5	-118.0	-4.0
sala_B_02	11:06:48	25.0	60.0	120.0	95.0	4	-118.0	-3.75
sala_B_02	11:06:41	25.0	65.0	100.0	95.0	3	-124.0	-10.0
sala_B_02	11:06:33	26.2	60.0	100.0	95.0	2	-126.0	-12.0
sala_B_02	11:06:11	26.0	60.0	100.0	95.0	1	-126.0	-12.0

Dashboard resultante do teste feito no vídeo.

No caso de teste apresentado no vídeo, a comunicação foi feita em um cenário onde os nós LoRa estavam geograficamente distantes, porém com poucos obstáculos físicos entre eles. O intuito deste teste era verificar o funcionamento do sistema em longas distâncias e como

isso afeta a qualidade da comunicação, comparando com o primeiro teste.

Na imagem acima apresenta-se o dashboard resultante da comunicação feita nesse caso de teste. Percebe-se, portanto, uma nítida piora na qualidade do sinal ao verificarmos que o RSSI indicou uma conexão ruim (em vermelho) em todos os dados transmitidos, sendo que apenas o primeiro pacote da sequência de transmissão foi perdido (transmissão relacionada ao boot 0).

Link do vídeo demonstrativo deste teste:

<https://drive.google.com/file/d/18Q4CcmaGK-GTh7vZ6ApF7xLIINGBmbSi/view?usp=sharing>

Monitoramento Ambiental & Rede LoRa								
ID N°	Timestamp	Temp (°C)	Umid (%)	Poeira	Bat (%)	Seq #	RSSI (dBm)	SNR (dB)
sala_B_02	11:19:11	25.0	60.0	100.0	95.0	9	-59.0	9.0
sala_B_02	11:18:28	25.0	60.0	100.0	95.0	8	-91.0	8.0
sala_B_02	11:18:21	25.0	60.0	100.0	95.0	7	-97.0	8.25
sala_B_02	11:18:14	50.5	60.0	100.0	95.0	6	-94.0	9.0
sala_B_02	11:18:07	24.0	60.0	100.0	95.0	5	-100.0	7.25
sala_B_02	11:17:59	25.0	60.0	120.0	95.0	4	-116.0	-2.75
sala_B_02	11:17:59	25.0	60.0	100.0	95.0	11	-125.0	-11.75
sala_B_02	11:08:47	25.0	60.0	100.0	95.0	10	-116.0	-2.25
sala_B_02	11:08:03	25.0	60.0	100.0	95.0	9	-114.0	0.25
sala_B_02	11:07:19	25.0	60.0	100.0	95.0	8	-116.0	-2.25

Dashboard do terceiro caso de teste.

Nesse terceiro caso de teste, a comunicação foi feita em um cenário onde os nós LoRa inicialmente estão geograficamente distantes e com múltiplas barreiras físicas entre eles. Ao longo da comunicação, os nós foram se aproximando e contornando as barreiras aos poucos, de forma que o intuito dele era verificar como a presença de obstáculos físicos afetam a qualidade da comunicação LoRa. A aproximação gradual dos nós foi realizada para determinar a partir de qual momento a comunicação começaria a se tornar viável, sem ocorrer perda de pacotes.

A imagem acima mostra o dashboard resultante deste teste a partir do tempo 11:17:59 (os tempos anteriores presentes no dashboard



são resquícios do segundo caso de teste). Nota-se que desta vez os primeiros 4 pacotes da sequência foram perdidos (pacotes 0, 1, 2 e 3), pois os nós ainda estavam muito distantes e as barreiras físicas no caminho ainda eram múltiplas. A medida que os nós foram se aproximando, pode-se perceber que os dados começaram a chegar (pacote 4) no nó receptor, porém ainda com um sinal muito ruim (indicado pelo RSSI) e mais adiante, com uma maior aproximação dos nós e redução dos obstáculos, a conexão começa a melhorar gradativamente.

A fim de deixar mais claro exatamente qual a disposição geográfica das ESP32 e quais as limitações encontradas, fica a descrição do caso de teste. O nó cliente-sensor, ao início do teste, se encontrava em frente à direção da Faculdade de Tecnologia (FT), enquanto que o nó do gateway se encontrava na parada de ônibus em frente à entrada da FT voltada ao ICC. Note que a distância física e os obstáculos na forma de paredes são consideráveis. Ao longo do teste, ambos os nós se deslocaram um em direção ao outro, de modo que ao final ambos estavam a poucos metros de distância e sem obstáculos.