MÓDULO 5:

COMPLEX EVENT PROCESSING PARA CAUSAS RAÍZES

Documento TR-01
Versão 1.0 de março de 2024
Projeto da disciplina STR
www.feelt.ufu.br

Execução:

LRI - LABORATÓRIO DE REDES INTELIGENTES
www.lri.ufu.br
UNIVERSIDADE FEDERAL DE UBERLÂNDIA

Alan Petrônio Pinheiro
Coordenador do projeto – UFU/LRI

Execução e pesquisa:
Leonardo Vecchi Meirelles
Lucas Humberto Jesus de Lima

1. DATA VERSÃO ORIGINAL 29-3-2024	**2. DATA ÚLTIMA ATUALIZAÇÃO** 03-4-2024	**3. DATA COBERTA** JAN/24 ATÉ MAI/24
4. TÍTULO DESTE DOCUMENTO COMPLEX EVENT PROCESSING PARA CAUSAS RAÍZES		**5a. PROCESSO SEI DO P&D** -
		5b. NÚMERO PROJETO P&D 00123456789....
6. AUTOR(ES) LEONARDO VECCHI MEIRELLES LUCAS HUMBERTO JESUS DE LIMA		**5c. ETAPA DO PROJETO** TODAS
		5d. TIPO DE PRODUTO DOCUMENTAÇÃO TÉCNICA DE SOFTWARE DE DISCIPLINA STR
7. ENDEREÇO AV. JOÃO NAVES DE ÁVILA, 2121, BLOCO 3N – UBERLÂNDIA - MG		**8. NÚMERO DO DOCUMENTO** **TR-01**
9. DISTRIBUIÇÃO DESTE DOCUMENTO DISTRIBUIÇÃO ABERTA A TODOS OS INTERESSADOS.		
10. NOTAS COMPLEMENTARES -		
11. RESUMO ESTE DOCUMENTO DESCREVE A MODELAGEM DOS ELEMENTOS DA APLICAÇÃO QUE RECEBE OS DADOS DE TELEMETRIA DAS UNIDADES CONSUMIDORAS DE UMA CIDADE, ANALISA TAIS PACOTES, IDENTIFICA EVENTOS BÁSICOS E TENTA ASSOCIÁ-LOS A UMA CAUSA RAÍZ.		
12. PALAVRAS-CHAVE P&D; IOT; SISTEMA EM TEMPO REAL, MERGE UNIT, SISTEMA SUPERVISÓRIO, MEDIÇÃO EM SUBESTAÇÕES DE ENERGIA.		
13. CLASSIFICAÇÃO SEGURANÇA: **ABERTA**	**14. NÚMERO DE PÁGINAS** -	**15. NOME DO RESPONSÁVEL PRINCIPAL E CONTATO** ALAN PETRÔNIO PINHEIRO. EMAIL: alan_petronio@yahoo.com.br. TELEFONE: (34)3239-4701

HISTÓRICO DE VERSÕES DESTE TR

Tabela 1 – *Histórico de versões deste reporte técnico.*

Versão	Data	Modificações
1.0	março/2024	• Principais elementos de projeto • Requerimentos básicos • Modelagem de pacotes e fluxo de pacotes
2.0	Abril/2024	• Identificar os feitos de cada integrante

SUMÁRIO

RESUMO GERAL

O módulo de Complex Event Processing (CEP) para identificação de causas raízes em dados de telemetria das unidades consumidoras de energia em uma cidade ou grande região desempenha um papel essencial na análise e interpretação dos dados de telemetria. Este módulo recebe dados de telemetria de cada unidade consumidora em intervalos regulares e realiza análises em tempo real para identificar eventos básicos, como quedas de energia em massa ou padrões incomuns de consumo. Em seguida, ele utiliza algoritmos específicos para associar esses eventos a possíveis causas raízes, fornecendo informações valiosas para a gestão eficaz da rede elétrica. O sistema é capaz de lidar com grandes volumes de dados de forma eficiente, garantindo uma resposta rápida a eventos adversos e contribuindo para a melhoria da qualidade do serviço aos consumidores finais.

Este tópico foi realizado em conjunto por ambos os membros do grupo.

1 – Introdução: visão geral da solução

1.1 – Propósito e escopo

O propósito do módulo de Complex Event Processing (CEP) para identificação de causas raízes em dados de telemetria das unidades consumidoras de uma cidade é fornecer uma solução eficaz para análise em tempo real dos dados de telemetria provenientes de diversas fontes, como medidores inteligentes e sensores distribuídos pela cidade. Este módulo tem como objetivo identificar eventos básicos e associá-los a possíveis causas raízes, contribuindo assim para a detecção precoce de problemas na rede elétrica e para a tomada de decisões proativas.

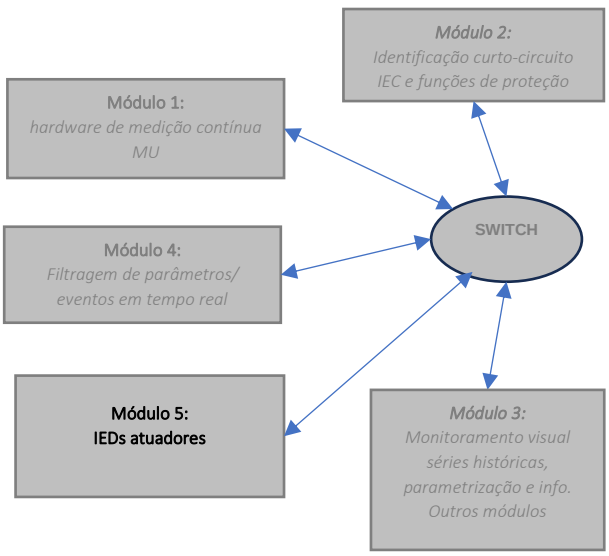


Figura 1.1.1: Visão geral de escopo.

O escopo deste módulo abrange as seguintes áreas-chave:

1) **Recebimento e Processamento de Dados:** O sistema será capaz de receber dados de telemetria de todas as unidades consumidoras em intervalos regulares, realizando análises em tempo real para identificar padrões e anomalias significativas.
2) **Identificação de Eventos Básicos:** O sistema irá identificar eventos básicos, tais como quedas de energia, flutuações de consumo de energia e anomalias de rede, a partir dos dados de telemetria recebidos.
3) **Associação a Causas Raízes:** Utilizando algoritmos específicos, o sistema irá associar os eventos identificados a possíveis causas raízes, permitindo uma melhor compreensão dos problemas na rede elétrica.
4) **Integração com Outros Subsistemas:** O módulo de CEP será integrado com outros subsistemas do sistema de monitoramento em tempo real, fornecendo informações valiosas para a gestão eficaz da rede elétrica e contribuindo para a melhoria da qualidade do serviço aos consumidores finais.

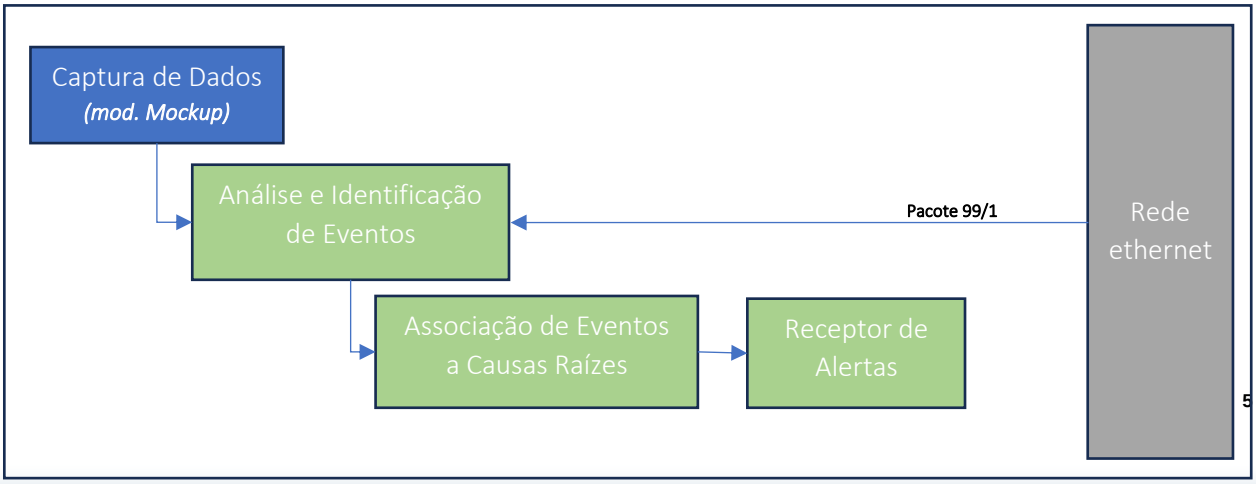
Este módulo será projetado para operar em larga escala, abrangendo toda a área de cobertura da cidade, e será capaz de lidar com grandes volumes de dados de forma eficiente, garantindo uma resposta rápida a eventos adversos.

Este tópico foi realizado em conjunto por ambos os membros do grupo.

1.2 – Produto: perspectivas e funções

O módulo de Complex Event Processing (CEP) para identificação de causas raízes em dados de telemetria das unidades consumidoras de uma cidade apresenta uma perspectiva inovadora e crucial para a gestão eficaz da rede elétrica em tempo real. Ao integrar tecnologias avançadas de processamento de eventos complexos, o sistema oferece uma visão abrangente do estado da rede elétrica, permitindo a detecção precoce de problemas e ações proativas para melhorar a confiabilidade e a eficiência operacional. Para isso, tem como principais funções:

1) Recebimento de Dados de Telemetria
2) Análise em Tempo Real
3) Identificação de Eventos Básicos
4) Associação a Causas Raízes
5) Integração com Outros Subsistemas



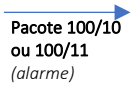


Figura 1.2.1: *Principais elementos de projeto de IEDs atuadores.*

Para entender o sistema, comecemos a análise observando a **Figura 1.2.1**. Com base nisto, descreve-se os elementos:

- Módulo de Captura de Dados:
  - Responsável por adquirir dados de telemetria das unidades consumidoras em tempo real.
  - Utiliza técnicas de captura eficientes para garantir a coleta precisa e contínua dos dados.
  - Realiza o pré-processamento inicial dos dados, incluindo limpeza e formatação, para prepará-los para análise subsequente.
  - Garante a integridade e a qualidade dos dados capturados antes de encaminhá-los para os módulos seguintes.

- Módulo de Análise e Identificação de Eventos:
  - Recebe os dados capturados e realiza análises em tempo real para identificar eventos significativos na rede elétrica.
  - Utiliza algoritmos avançados para detectar padrões e anomalias nos dados de telemetria, como quedas de energia, flutuações de consumo e comportamentos suspeitos.
  - Opera de forma eficiente e escalável, fazendo uso de técnicas de multiprocessamento para processar grandes volumes de dados em paralelo.
  - Fornece uma visão abrangente do estado atual da rede elétrica, permitindo a detecção precoce de problemas e ações proativas para mitigar riscos.

- Módulo de Associação de Eventos a Causas Raízes:
  - Recebe os eventos identificados pelo módulo de análise e os associa a possíveis causas raízes na rede elétrica.
  - Utiliza técnicas avançadas de correlação e análise causal para determinar as relações entre os eventos detectados e as possíveis fontes de problemas.
  - Gera alertas ou alarmes precisos e contextualizados, fornecendo informações úteis para a tomada de decisões pelos operadores.
  - Opera de forma adaptável, ajustando dinamicamente as associações de acordo com as mudanças na rede elétrica e nas condições operacionais.

- Módulo de Receptor de Alertas:
  - Recebe os alertas ou alarmes gerados pelo módulo de associação de eventos.
  - Processa os alertas recebidos e os encaminha para a interface do usuário ou para sistemas externos, conforme necessário.
  - Permite a visualização e o acompanhamento em tempo real do status da rede elétrica, facilitando a rápida resposta a eventos e ações corretivas.
  - Oferece recursos de gerenciamento e registro de alertas para análises posteriores e melhoria contínua do sistema.

Este tópico foi realizado em conjunto por ambos os membros do grupo.

1.3 – Restrições do produto e considerações

A solução geral aqui prevista foi testada para condições específicas e nestas, foram identificadas as seguintes restrições ou limitações para os quais o sistema proposto não foi projetado para atuar. Estas restrições e limitações são mostradas na tabela da sequência.

Tabela 1.3.1: *Restrições e limitações previstas para sistema.*

Nº	Restrição/limitação	Descrição/detalhamento
1	Limitação de Dados de Entrada	O sistema é capaz de processar apenas os dados de telemetria das unidades consumidoras fornecidos pelo mockup designado para simular a geração de dados. Não suporta outros formatos de entrada de dados ou fontes externas de telemetria.
2	Restrição de Capacidade de Processamento	O sistema pode enfrentar limitações de capacidade de processamento, especialmente ao lidar com grandes volumes de dados de telemetria de uma cidade inteira em tempo real. Isso pode afetar a velocidade de análise e a capacidade de detecção de eventos em tempo hábil.
3	Dependência de Threads para Multiprocessamento	O processamento em paralelo depende da capacidade do sistema de gerenciar threads eficientemente para alcançar o multiprocessamento. Limitações de hardware ou implementação inadequada de threads podem afetar o desempenho geral do sistema.
4	Limitação de Emulação de Eventos	O sistema é limitado na capacidade de emular eventos complexos para fins de teste e validação. A complexidade dos eventos que podem ser emulados pode ser restrita pela capacidade de simulação do sistema.
5	Necessidade de Conectividade de Rede Adequada	O sistema requer uma conexão de rede confiável e de baixa latência para receber dados de telemetria e enviar alertas de causas raízes para o módulo receptor de alertas. Limitações na conectividade de rede podem afetar a comunicação eficaz entre os módulos do sistema.
6	Requisitos de Segurança de Dados	O sistema pode estar sujeito a limitações de segurança de dados, especialmente durante a transmissão e o armazenamento de dados de telemetria sensíveis. É essencial implementar medidas robustas de criptografia e autenticação para proteger a integridade e a confidencialidade dos dados.

Este tópico foi realizado em conjunto por ambos os membros do grupo.

2 – Requisitos

2.1 – Cenários de uso

Os seguintes cenários foram identificados para este sistema.

Cenário 1 – Sem Detecção de Eventos: neste cenário, o sistema está operando em condições normais, sem detectar qualquer atividade suspeita de roubo de eletricidade ou queda de energia. Ele mantém seu funcionamento padrão, registrando os dados de telemetria e aguardando novas informações.

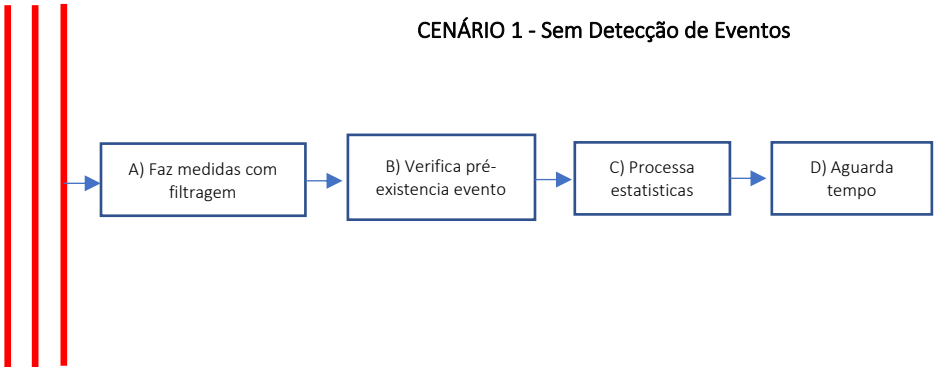


Figura 2.1.1: *Cenário de aplicação.*

Cenário 2 – Detecção de Possível Roubo de Eletricidade: o algoritmo dedicado à detecção de suspeitas de roubo de eletricidade está constantemente em execução, monitorando os padrões de consumo de energia das unidades consumidoras. Ele realiza uma análise contínua dos dados de telemetria, comparando-os com os padrões históricos e comportamentos esperados. Quando identifica discrepâncias significativas nos padrões de consumo, o algoritmo consulta a árvore de decisão para verificar os critérios de suspeita. Com base nessa análise contínua e automatizada, o sistema determina a probabilidade de ocorrência de roubo de eletricidade em tempo real. Essa abordagem proativa permite que o sistema identifique rapidamente potenciais casos de roubo de eletricidade e tome as medidas necessárias para mitigar essas atividades ilegais.

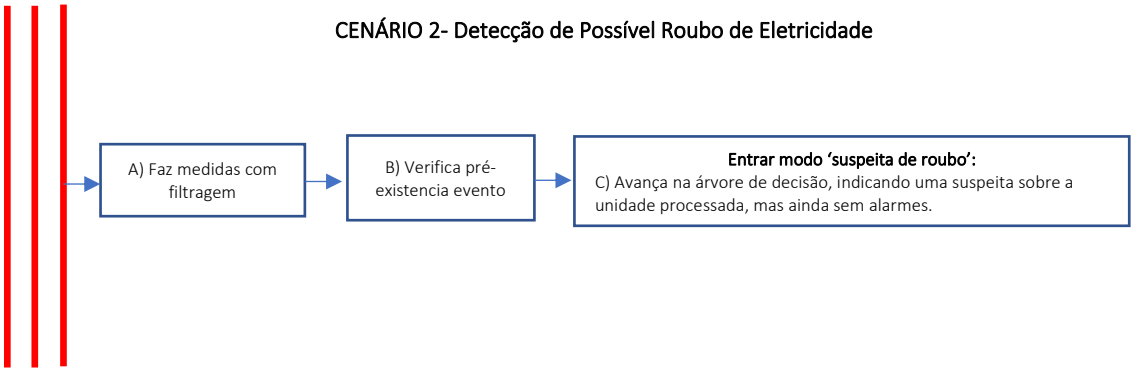


Figura 2.1.2: *Cenário de aplicação.*

Cenário 3 – Verificação de Suspeita Falsa e Retorno ao Normal: após investigações adicionais ou monitoramento contínuo, o sistema determina que uma suspeita anterior de roubo de eletricidade

não era fundamentada e que o consumo de energia da unidade consumidora retornou aos níveis normais. O algoritmo de detecção de padrões de consumo de energia confirma que não há mais discrepâncias significativas nos dados de telemetria. Como resultado, o sistema encerra a suspeita e retorna ao seu modo de operação padrão. Esse cenário destaca a capacidade do sistema de adaptar-se dinamicamente às mudanças nas condições e de evitar alarmes falsos, garantindo uma detecção precisa e eficaz de casos reais de roubo de eletricidade.

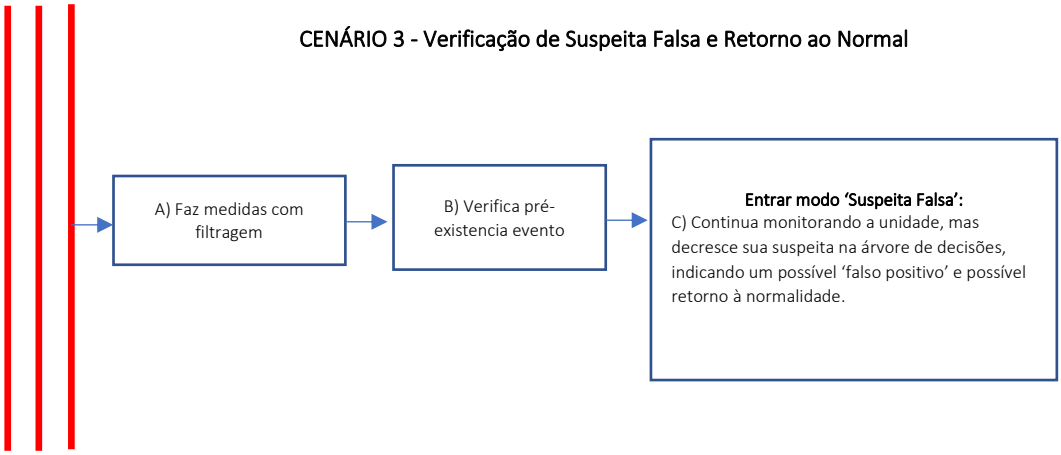


Figura 2.1.3: *Cenário de aplicação.*

Cenário 4 - Alerta de Suspeita de Roubo de Eletricidade: Se a análise do sistema indicar uma alta probabilidade de roubo de eletricidade, ultrapassando um threshold pré-definido na árvore de decisão, ele lança um alerta para o módulo receptor de alertas. Esse alerta informa sobre a suspeita de roubo de eletricidade e pode acionar ações adicionais, como o envio de equipes de inspeção para investigar a situação.

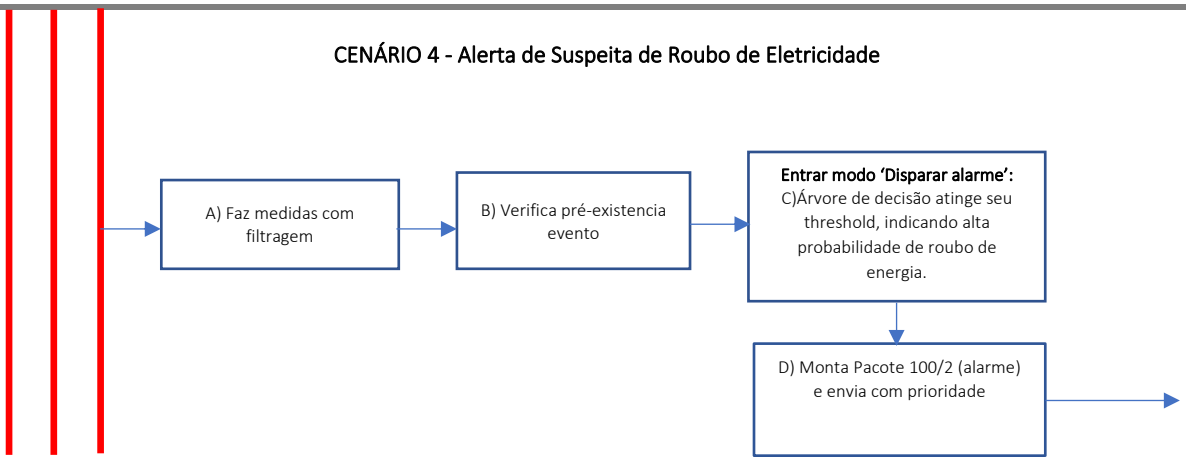


Figura 2.1.4: *Cenário de aplicação.*

Cenário 5 – Não há falta de energia: Quando esse evento é detectado, o sistema segue a sequência de passos indicada na figura da sequência. Ela ilustra como deve ser seu comportamento.

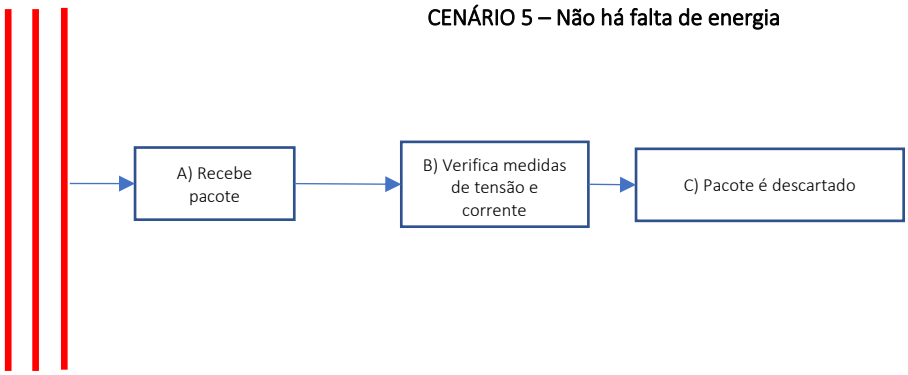


Figura 2.1.5: *Cenário de aplicação.*

Cenário 6 – Detecção de falta de energia: Nesse cenário, há uma detecção de que a MU que enviou o pacote possui falta de energia. Para esse caso, o serviço deve agrupá-lo com outras MU's que apresentem tal comportamento, e enviar um pacote que identifique todas para os módulos relevantes.

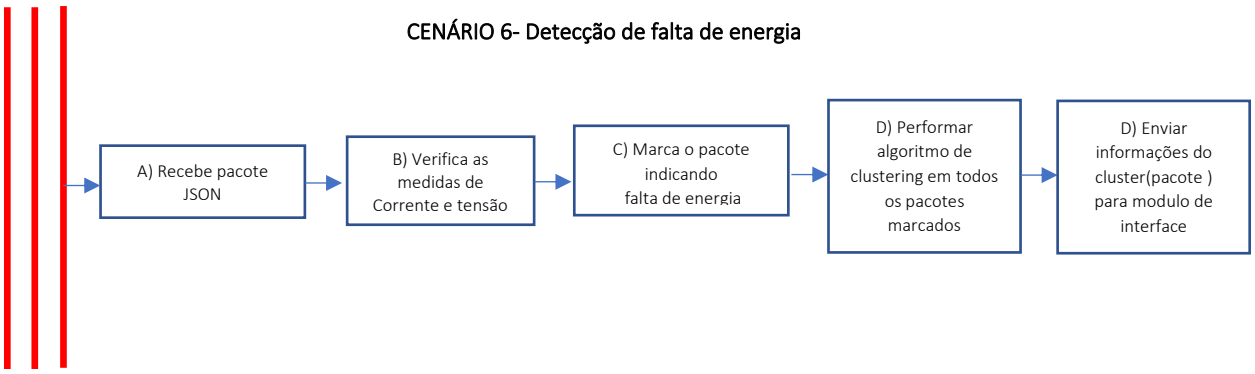


Figura 2.1.6: *Cenário de aplicação.*

O cenário 1 foi realizado de forma conjunta; os cenários de 2 a 4 foram realizados pelo autor Leonardo Vecchi Meirelles e, os cenários 5 e 6 pelo autor Lucas Humberto Jesus de Lima.

2.2 – Requisitos e validação

Com base nas entrevistas com os clientes, equipe de engenharia e avaliações de cenário de uso, desenvolveu-se na sequência a seguinte lista de requerimentos, vista na tabela da sequência.

Tabela 2.2.1: *Mapa de requerimentos.*

Classe/ Componente	Nº req.	Requisito	Origem requisito	Prio	Tipo validação
1 - Robustez	1.1	O sistema deve ser capaz de lidar com grandes volumes de dados de telemetria sem comprometer o desempenho.	Especificação do Projeto	1	Testes de Carga para avaliar a capacidade de processamento e resposta do sistema sob cargas de trabalho pesadas.
	1.2	O sistema deve ser resistente a falhas de hardware e software, garantindo alta disponibilidade e confiabilidade operacional.		1	Testes de Falha para simular condições de falha e garantir que o sistema permaneça operacional e recupere-se adequadamente.
2 - Funcional	2.1	O sistema deve ser capaz de identificar padrões anômalos nos dados de telemetria, com uma latência máxima de 5 segundos a partir da detecção do evento até a geração do alerta correspondente.	Especificação do Projeto	1	Testes de Unidade para verificar a precisão e eficácia dos algoritmos de detecção de padrões.
	2.2	O sistema deve ser capaz de analisar continuamente os padrões de consumo de energia das unidades consumidoras em tempo real, garantindo que cada ciclo de análise seja concluído em menos de 1 minuto.		1	Testes de Integração para garantir que todos os componentes do sistema funcionem de forma coesa e integrada.
3 – Não funcionais	3.1	O sistema deve garantir a segurança e confidencialidade dos dados de telemetria, aderindo às normas de segurança da informação, como ISO/IEC 27001.	Normas de Segurança da Informação	1	Auditoria de Segurança para validar a conformidade do sistema com os requisitos de segurança estabelecidos.
	3.2	O sistema deve ser capaz de operar em ambientes desafiadores, como subestações de energia, seguindo as normas de segurança elétrica, como a IEC 61850.	Normas de Segurança Elétrica	1	Certificação de Conformidade para garantir que o sistema atenda aos padrões de segurança elétrica estabelecidos.
	3.3	O sistema deve ser altamente escalável, permitindo a adição de novas unidades consumidoras e expansão para áreas urbanas maiores.	Especificação do Projeto	2	Testes de Escalabilidade para avaliar a capacidade do sistema de lidar com um aumento significativo no volume de dados e carga de trabalho.

Este tópico foi realizado em conjunto por ambos os membros do grupo.

2.3 – Versionamento

Os recursos do software são distribuídos em versões conforme estimado pela tabela na sequência.

Tabela 2.3.1: *Tabela de recursos do sistema e versão.*

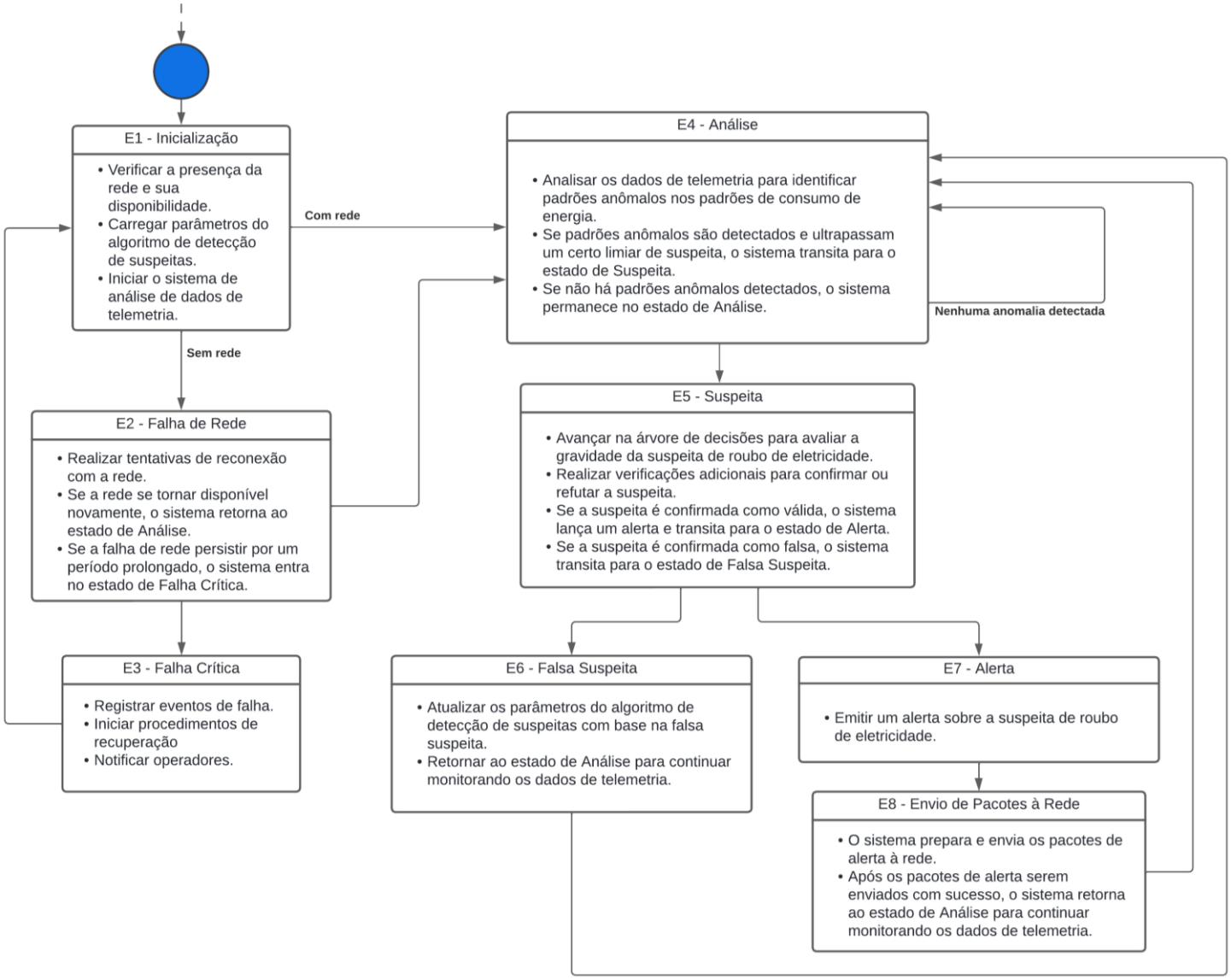
Versão	Recurso
1.0 *(mar/24)*	▪ O sistema deve ser capaz de identificar padrões anômalos nos dados de telemetria, com uma latência máxima de 5 segundos a partir da detecção do evento até a geração do alerta correspondente. ▪ O sistema deve ser capaz de analisar continuamente os padrões de consumo de energia das unidades consumidoras em tempo real, garantindo que cada ciclo de análise seja concluído em menos de 1 minuto.

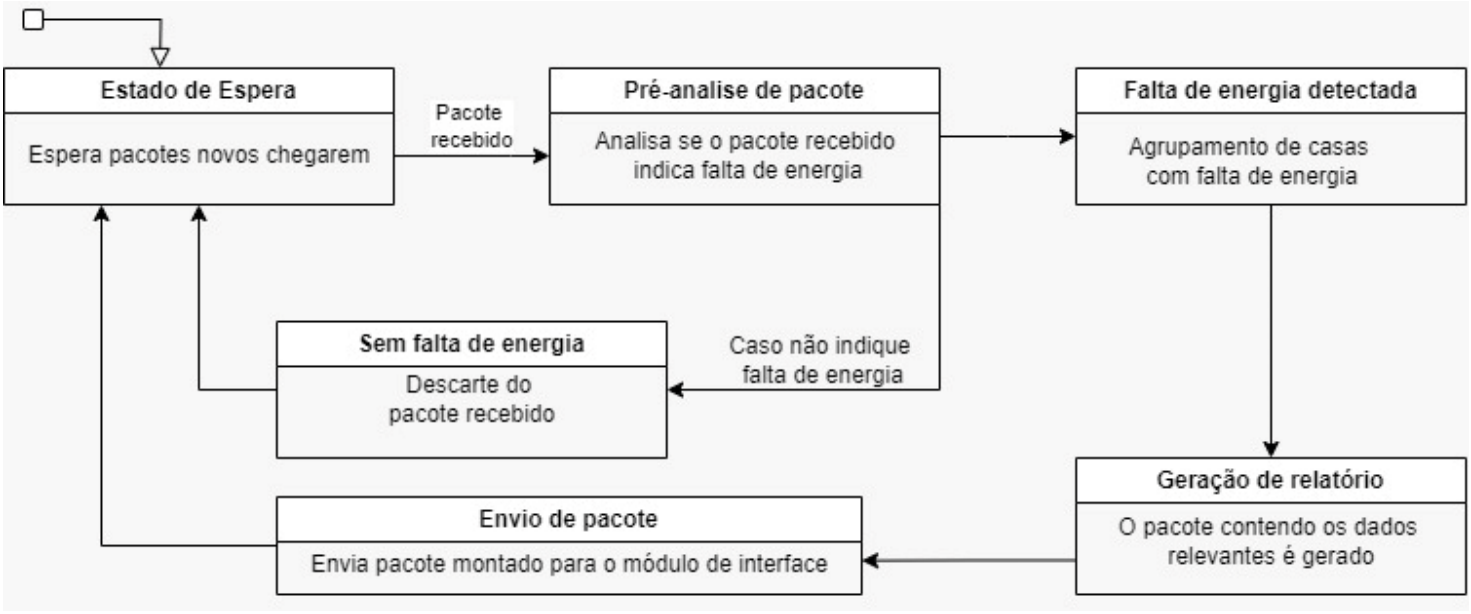
Este tópico foi realizado em conjunto por ambos os membros do grupo.

2.4 – Elementos de projeto

2.4.1 – Máquina de estados

Baseado nos cenários identificados e requerimentos construídos, tem-se a seguinte proposição para as máquinas de estados de um CEP para detecção de roubos de eletricidade e falta de energia.





O primeiro diagrama de estados apresentado foi modulado pelo autor Leonardo Vecchi Meirelles, enquanto o segundo, pelo aluno Lucas Humberto Jesus de Lima.

3 – Modelagem

3.1 – Blocos de elementos principais

Na sequência é mostrado um conjunto de diagramas de blocos para exemplificar a arquitetar do sistema. Cada bloco é um objeto e estes são os principais objetivos previstos na solução. As setas indicam o fluxo das informações.

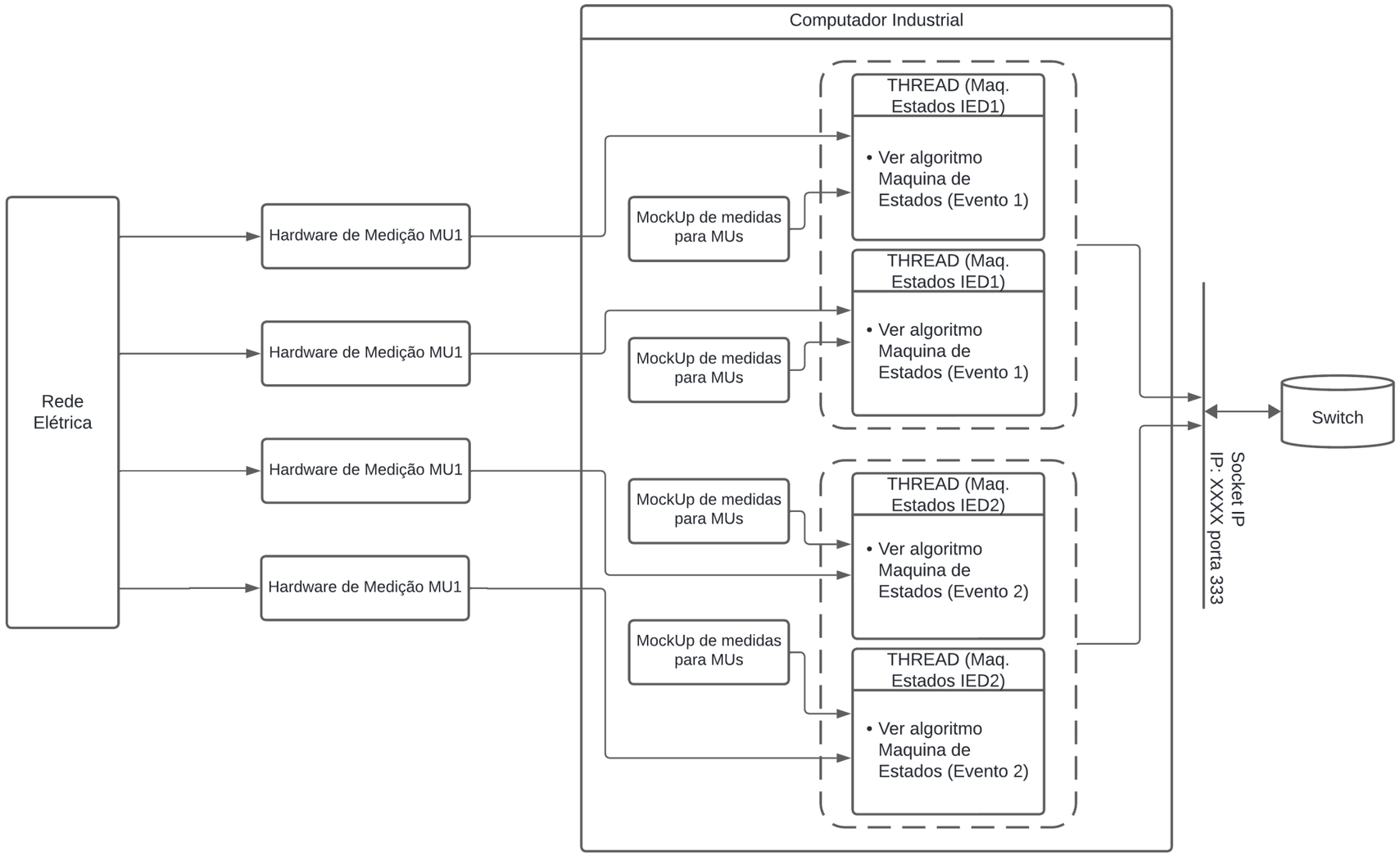


Figura 3.1.1: Componentes básicos

Este tópico foi realizado em conjunto por ambos os membros do grupo.

3.2 – Tabela geral de objetos IPSO e recursos de URI

Todos os recursos previstos entre as aplicações do TWINS são feitos por códigos IPSO[1] em padrão URI. Eles podem ser a designação de uma conexão para um evento, o formato de um pacote, o formato de um JSON específico dentre outras coisas. Ajudam a racionalizar e identificar facilmente o recurso e a estruturar o pensamento. Estes códigos estão dispostos na tabela da sequência.

Tabela 3.2.1.: *Tabela IPSO de recursos do projeto.*

Objeto	Recurso		Significado	Link
	Nível 1	Nível 2		
99 *Mensagens com medidas*	**1** Medidas periódicas		Medidas elétricas trafegando na rede quando o MU está operando em condições normais.	
	2 Medidas em situação alteração		Medidas elétricas trafegando na rede quando o MU identificou uma grande mudança nos parâmetros e aí começa a transmitir mensagens com maior frequência.	
100 *Mensagens de alerta*	**10** Alerta de queda de energia		Mensagem contendo dados relevantes para identificação de evento referente à queda de energia.	
	11 Alerta de furto de energia		Mensagem contendo dados relevantes para identificação de evento referente à furto de energia.	

Este tópico foi realizado em conjunto por ambos os membros do grupo.

3.3 – Fluxo geral de mensagens

A figura na sequência ilustra resumidamente as mensagens que são trocadas, em diferentes circunstâncias, entre o sistema MU e demais módulos.

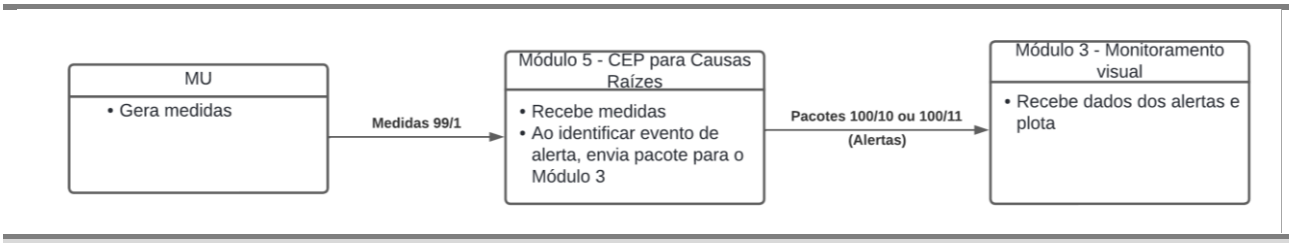


Figura 3.3.1: Diagrama ilustrando a ideia de como o front tem suas informações atualizadas dinamicamente

Como observado, o MU envia para a rede em regime de broadcasting o pacote 99/1.

O Módulo 5 constantemente recebe as medidas e, ao detectar evento de alerta, como roubo de eletricidade ou queda de energia, envia um pacote 100/10 (alerta de roubo de eletricidade) ou 100/11 (alerta de queda de energia) para o Módulo 3, que recebe o alerta e apresenta visualmente em sua interface.

Para entender melhor esta dinâmica, elas são tratadas individualmente com mais detalhes nas próximas subseções.

Este tópico foi realizado em conjunto por ambos os membros do grupo.

3.4 – Modelagem detalhada dos recursos

3.4.1 – Envio de mensagens de medição

A figura na sequência ilustra como deve ser o comportamento temporal da dinâmica de envio de pacotes de medidas elétricas do MU para o Módulo 5, que será o envio de pacotes ininterruptos, independente da ocorrência de evento o qual, ao ocorrer um cenário de evento, envia um pacote de alerta para o Módulo 3 de visualização.

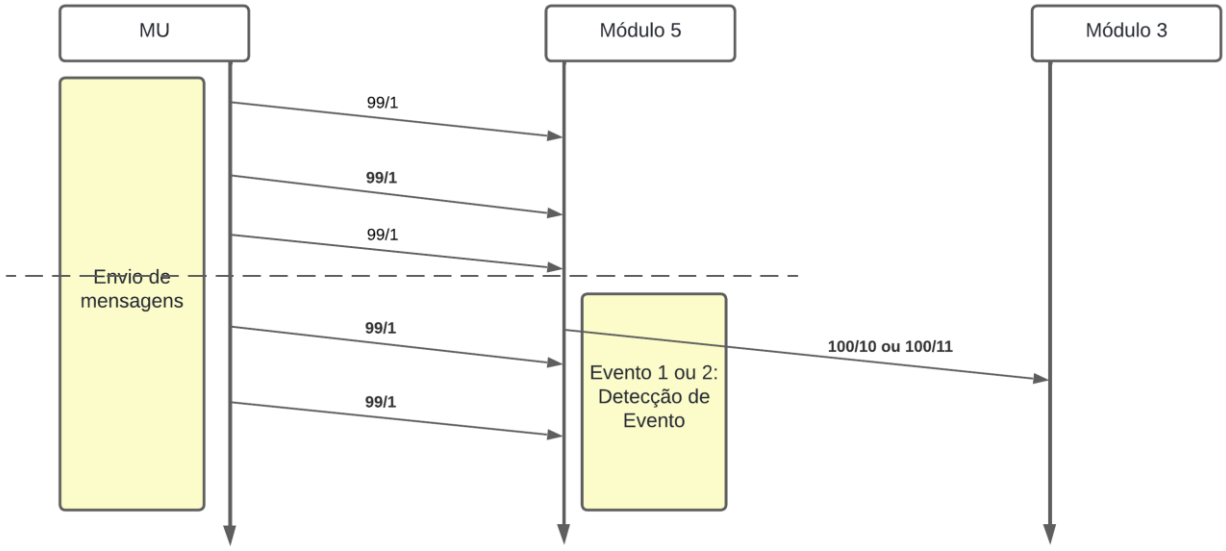


Figura 3.4.1.1: Diagrama pacotes e eventos associados ao envio de mensagens.

Com base no diagrama mostrado na figura anterior, descreve-se agora o comportamento dos principais eventos identificados.

Tabela 3.4.1.1: *Algoritmos equivalentes aos eventos vistos na Figura 3.4.1.1.*

Evento/Ação	Algoritmo

Evento 1 (detecção de roubos de eletricidade)	**Este evento deve implementar a máquina de estado mostrada na figura Figura 2.4.2.1. Algoritmo básico:**
	
	# Definição dos estados
	class State:
	Initialization = 1
	Analysis = 2
	Suspicion = 3
	FalseSuspicion = 4
	Alert = 5
	NetworkFailure = 6
	NoEvents = 7
	CriticalFailure = 8
	SendingPackets = 9
	
	# Função para iniciar o sistema
	def initialize_system():
	current_state = State.Initialization
	# Realizar a inicialização do sistema aqui
	# Transitar para o estado de Análise após a inicialização
	
	# Função para analisar os dados de telemetria
	def analyze_telemetry():

```
    current_state = State.Analysis
    # Realizar a análise dos dados de telemetria aqui
    # Verificar se há padrões anômalos e transitar para o estado de Suspeita se necessário

    # Transições
    if anomalous_patterns_detected:
        current_state = State.Suspicion

# Função para lidar com a suspeita de roubo de eletricidade
def handle_suspicion():
    current_state = State.Suspicion
    # Realizar verificações adicionais e confirmar ou refutar a suspeita aqui
    # Transitar para o estado de Alerta se a suspeita for confirmada como válida
    # Transitar para o estado de FalseSuspicion se a suspeita for confirmada como falsa

    # Transições
    if suspicion_confirmed:
        current_state = State.Alert
    elif false_suspicion:
        current_state = State.FalseSuspicion

# Função para emitir um alerta sobre a suspeita de roubo de eletricidade
def raise_alert():
    current_state = State.Alert
    # Emitir o alerta sobre a suspeita de roubo de eletricidade aqui
    # Transitar para o estado de SendingPackets para enviar os pacotes à rede

    # Transições
    if alert_sent_successfully:
        current_state = State.SendingPackets

# Função para enviar os pacotes de alerta à rede
def send_packets_to_network():
    current_state = State.SendingPackets
    # Preparar e enviar os pacotes de alerta à rede aqui
    # Transitar de volta para o estado de Análise após o envio dos pacotes

    # Transições
    if packets_sent_successfully:
        current_state = State.Analysis

# Função para lidar com a falha de rede
def handle_network_failure():
    current_state = State.NetworkFailure
    # Realizar tentativas de reconexão com a rede aqui
    # Transitar para o estado de CriticalFailure se a falha persistir por um período prolongado

    # Transições
    if network_reconnected:
        current_state = State.Analysis
    elif network_failure_persists:
        current_state = State.CriticalFailure

# Função principal
def main():
    initialize_system()
    while True:
        if current_state == State.Analysis:
            analyze_telemetry()
        elif current_state == State.Suspicion:
            handle_suspicion()
        elif current_state == State.Alert:
            raise_alert()
```

	elif current_state == State.SendingPackets:
	send_packets_to_network()
	elif current_state == State.NetworkFailure:
	handle_network_failure()
	
	if __name__ == "__main__":
	main()

Evento 2 (Falta de energia)	**Este evento deve implementar a máquina de estado mostrada na figura. Algoritmo básico:** 1) Inicialize um servidor para escutar pacotes JSON recebidos pela Ethernet. 2) Ao receber um pacote JSON: a) Analise o pacote para extrair a tensão elétrica, a corrente e as coordenadas geográficas de cada casa. b) Armazene os dados extraídos para processamento posterior. 3) Para cada casa nos dados recebidos: a) Verifique se a tensão ou corrente elétrica cai abaixo dos limites predefinidos indicando falta de energia. b) Se a casa for identificada como sem energia, marque-a em conformidade. 4) Use um algoritmo de agrupamento (por exemplo, K-means, DBSCAN) para agrupar casas sem energia com base em suas coordenadas geográficas. 5) Itere pelos clusters identificados: a) Calcule o centróide de cada cluster para representar seu centro geográfico. b) Armazene as informações do cluster, incluindo o centróide e as casas dentro do cluster. 6) Gere alertas ou relatórios com base nos aglomerados de casas sem energia identificados. 7) Para cada cluster: a) Determine o tamanho do cluster (número de casas). b) Gere um alerta ou relatório indicando a localização e o tamanho do cluster.

O algoritmo apresentado para o Evento 1 (detecção de roubos de eletricidade) foi realizado pelo autor Leonardo Vecchi Meirelles, enquanto o algoritmo do Evento 2 (Falta de energia), pelo autor Lucas Humberto Jesus de Lima.

Já o formato do pacote de dados é mostrado na sequência.

Tabela 3.4.1.2: *Formato do pacote de dados 99/1*

Campo	Valores	Significado
URI	**99/1**	Pacote de envio de medidas regulares.
idMU	int	Identificador de qual MU está gerando este pacote.
coord	(string,string)	Coordenadas geográficas em que a UM se encontra.
idAtivo	String (opcional)	Identificado de qual ativo o ID está monitorando.
numPct	inteito	Número do pacote gerado incrementalmente.
timpeStamping	Data UMC	Data e hora em que foi gerado o pacote.
freqEnvioMS	int	Taxa de aquisição e intervalo de tempo em que foram coletas as medidas e calculada a média.
medidas	[MedidasEletricas]	Contém um array com objetos do tipo "MedidasEletricas" **médias** que tem o seguinte formato: fase: String (A,B,C,N) tensao: float corrente: float angTensao: Float potApaVA: float potReatVAr: float potRealW: Float fatorP: float freq: float

E nas ocasiões em que acontecem grandes variações, gera-se no lugar do 99/1 o pacote 99/2 que tem mais informações e deve ter mais prioridade na rede.

Tabela 3.4.1.3: *Formato do pacote de dados 99/2*

Campo	Valores	Significado
URI	**99/2**	Pacote de envio de medidas urgentes com grande variação

idMU	int	Identificador de qual MU está gerando este pacote
coords	(string,string)	Coordenadas geográficas em que a UM se encontra.
idAtivo	String (opcional)	Identificado de qual ativo o ID está monitorando.
numPct	inteito	Número do pacote gerado incrementalmente
timpeStamping	Data UMC	Data e hora em que foi gerado o pacote
medidas	[MedidasEletricas]	Contém um array com objetos do tipo "MedidasEletricas" **instantâneas** que tem o seguinte formato: fase: String (A,B,C,N) tensao: float corrente: float angTensao: Float potApaVA: float potReatVAr: float potRealW: Float fatorP: float freq: float
variavelDescrepante	[String]	Indica qual é a variável que sofreu a grande variação e acarretou a ocorrência deste pacote.
faseDescrepante	[String]	Indica qual é a fase que sofreu a grande variação e acarretou a ocorrência deste pacote.

Para envio de alertas, são utilizados os pacotes 100/10 para cenários de roubo de eletricidade e 100/11 para cenários de queda de energia.

Tabela 3.4.1.4: *Formato do pacote de dados 100/10*

Campo	Valores	Significado
URI	100/10	Pacote de envio de controle do protocolo de medição.
idPKT	int	Identificador do pacote gerado. Auto incrementável.
idMU	int	Identificador de qual MU está gerando este pacote
Cords	[double, double]	Coordenadas geográficas em que a UM se encontra.

Tabela 3.4.1.5: *Formato do pacote de dados 100/11*

Campo	Valores	Significado
URI	**100/11**	Pacote de envio de controle do protocolo de medição.
idPKT	int	Identificador do pacote gerado. Auto incrementável.
idMUs	[int]	Identificador de quais UM's estão no cluster.
cords	[(string,string)]	Coordenadas geográficas em que as UM's se encontram.

Este tópico foi realizado em conjunto por ambos os membros do grupo.