Reutilização Sustentável de TV Boxes como Centrais de Tratamento de Dados para Aplicações de IoT em Nuvem

Ana Clara Custodio[1], André Luiz Vicente Silva[1], Andressa Francisco Vieira[1], Matheus Martins de Sousa[2], Ruan Pablo De Assis Neres[1], Vinícius Oliveira Magalhães[1], Walteno Parreira Junior[1]

[1]Instituto Federal do Triângulo Mineiro (IFTM) – Campus Uberlândia Centro
Rua Blanche Galassi, 150 – Morada da Colina – 38.411-104 – Uberlândia – MG – Brasil

[2]Escola Superior de Agricultura Luiz de Queiroz – Universidade de São Paulo (ESALQ/USP)
Av. Pádua Dias, 11 – Agronomia – 13.418-900 – Piracicaba – SP – Brasil

{ana.custodio,andre.silva,andressa.vieira,ruan.neres,vinicius.magalhaes}@estudante.iftm.edu.br, matheusnerdmartins@gmail.com, waltenomartins@iftm.edu.br

Abstract. *This paper introduces UAI.py, a sustainable computing initiative focused on repurposing TV Box devices for Internet of Things (IoT) applications. By deploying an Armbian-based Linux distribution, these devices function as low-cost Data Processing Centers (DPC), fully integrated with an Amazon Web Services (AWS) cloud infrastructure. The proposed solution was validated through a series of performance tests, which confirmed the technical feasibility of the idea.*

Resumo: *Este artigo apresenta o projeto UAI.py, uma iniciativa de computação sustentável que reutiliza dispositivos TV Box para aplicações de Internet das Coisas (IoT). Utilizando uma distribuição Linux, baseada em Armbian, os dispositivos atuam como centrais de tratamento de dados (CTD) de baixo custo, integradas a uma infraestrutura em nuvem na Amazon Web Services (AWS). A solução foi validada após testes de desempenho que evidenciaram a viabilidade técnica da ideia.*

1. Introdução

O avanço tecnológico impulsiona um ciclo contínuo de consumo e descarte de dispositivos eletrônicos, gerando um volume crescente de lixo (*e-waste*), um dos desafios ambientais mais significativos da atualidade (FRANZ; SILVA, 2022). Paralelamente, a expansão de soluções de Internet das Coisas (IoT) em setores como a agricultura e a indústria é frequentemente limitada pelo alto custo do *hardware* especializado.

Nesse cenário, a Receita Federal do Brasil anualmente apreende e descarta milhares de dispositivos *TV Box*, equipamentos originalmente destinados ao acesso não autorizado a conteúdo de mídia. Apenas entre 2019 e 2020, mais de 97 mil unidades foram destruídas (BRASIL, 2021), gerando custos operacionais, desperdício de materiais, componentes eletrônicos valiosos e também contribuindo para a poluição ambiental.

O projeto UAI.py, desenvolvido em parceria com a Receita Federal através do programa “Além do Horizonte” (ANATEL, 2023), surge como uma ponte entre esses dois problemas. A iniciativa propõe uma solução sustentável que transforma esse passivo em um ativo tecnológico: a reutilização das *TV Boxes* apreendidas, descaracterizando-as e convertendo-as em Centrais de Tratamento de Dados (CTD) de baixo custo. A CTD resultante é uma plataforma agnóstica, projetada para atuar como

um gateway local que coleta, processa e envia dados de sensores para um *backend* robusto na nuvem, fomentando a economia circular e a inovação social.

2. Metodologia

O processo de conversão de uma *TV Box* em uma CTD funcional envolve uma seleção criteriosa de *hardware* e um procedimento técnico rigoroso de descaracterização, que é o núcleo desta pesquisa.

2.1. Seleção e Análise do *Hardware*

A diversidade de modelos de *TV Box* disponibilizados pela Receita Federal exigiu uma análise técnica. Foram realizados testes de estresse computacional para avaliar a estabilidade do processador, o gerenciamento de memória e a performance de rede sob carga. Os testes incluíram a execução de *benchmarks* sintéticos para CPU e memória, monitoramento da temperatura do *System-on-a-Chip* (SoC) durante longos períodos de operação e testes de *throughput* de rede.

Os modelos TX6-p e TX9, foram selecionados por apresentarem o melhor equilíbrio entre capacidade computacional e eficiência energética. Ambos são equipados com o SoC Amlogic S905W, um processador *Quad-core* ARM Cortex-A53 e 2 GB de memória RAM. Essas especificações, embora modestas para computação de *desktop*, são mais do que adequadas para atuar como um orquestrador local de dados, capaz de gerenciar múltiplas conexões de sensores e manter a comunicação com a nuvem.

2.2. O Processo de Descaracterização

A descaracterização é um requisito obrigatório da Receita Federal e o coração técnico do projeto. Seu objetivo é garantir a remoção permanente do *software* de pirataria, adaptando o dispositivo para um novo propósito. O processo é dividido em etapas fundamentais, detalhadas a seguir.

A primeira etapa consiste em substituir o sistema Android nativo. Por isso, a distribuição Armbian, baseada em Debian Linux, foi escolhida por seu baixo uso de *hardware*, estabilidade e, principalmente, pelo robusto suporte da comunidade para a arquitetura ARM, incluindo os SoCs Amlogic. Foram selecionadas imagens específicas do sistema com versões de *kernel* customizadas (5.7.0 para o modelo TX6-p e 5.9.0 para o TX9), que continham os drivers necessários para componentes essenciais como a controladora de rede Ethernet e as portas USB, garantindo compatibilidade e desempenho.

A imagem do sistema foi gravada em um cartão microSD e, também foi incluído um *bootloader* personalizado (U-Boot), que é o primeiro *software* a ser executado quando o dispositivo é ligado. O U-Boot foi configurado para alterar a sequência de inicialização padrão do *hardware*, instruindo-o a carregar o sistema operacional a partir do cartão microSD em vez da memória *flash* interna, onde o Android original reside. Uma vez que o sistema está no ar, o acesso de superusuário (*root*) é obtido e o último passo adapta o sistema a partir da execução de um *script* de instalação (install-aw.sh).

Este *script* foi modificado para reconhecer o *layout* de partição da memória interna (eMMC) da *TV Box* e forçar a instalação do novo sistema operacional diretamente nela. O *script* automatiza o particionamento da eMMC para fazer a cópia do

sistema de arquivos raiz (*rootfs*) do cartão para a nova partição e a instalação do *bootloader* na área de inicialização da memória interna. Este procedimento sobrescreve completamente o sistema Android original, finalizando o processo de descaracterização e consolidando a *TV Box* como um dispositivo computacional de propósito geral.

3. Desenvolvimento

Uma vez descaracterizada, a UAI.py torna-se o núcleo de uma arquitetura de IoT distribuída, atuando como ponte entre o ambiente físico local e a nuvem.

3.1 Orquestrador de Dados Local

A CTD executa um ambiente de *software* estável para gerenciar a coleta e o envio de dados. O componente central é um servidor desenvolvido em Python, que utiliza o *framework* FastAPI para criar uma API local de gerenciamento. A principal responsabilidade deste servidor é operar como um servidor web HTTP, o ponto central para toda a comunicação com os dispositivos de campo, como sensores e microcontroladores.

O HTTP (Hypertext Transfer Protocol) é um protocolo de comunicação baseado no modelo de requisição e resposta (*request/response*), amplamente adotado por sua simplicidade e compatibilidade universal entre diferentes plataformas de *hardware* e *software*. A necessidade de uma interface HTTP no projeto é fundamental para padronizar a entrada de dados proveniente dos microcontroladores (como ESP32 e ESP8266), uma vez que a padronização das interfaces de comunicação constitui um requisito essencial para garantir a interoperabilidade em ecossistemas de IoT (MELO; VIEIRA, 2016).

Nesta arquitetura, os microcontroladores estabelecem uma comunicação direta com o servidor por meio de requisições HTTP. As leituras de telemetria são enviadas através de métodos POST para *endpoints* específicos da API */device/actor-data*, permitindo que o sistema identifique e organize a origem de cada dado. O servidor, por sua vez, processa as requisições de forma síncrona ou assíncrona, garantindo a eficiência no recebimento e a imediata validação das informações antes da sua persistência no banco de dados.

Essa arquitetura torna o sistema organizado e de fácil integração, refletindo o princípio da modularidade, considerado fundamental no projeto de sistemas distribuídos robustos (COULOURIS et al., 2000; BURNS, 2018). Assim, novos sensores podem ser adicionados à rede simplesmente direcionando suas requisições para as rotas definidas no servidor. O servidor gerencia as requisições recebidas, garantindo o processamento das mensagens, que são padronizadas em formato JSON.

Após receber os dados via HTTP, o servidor Python realiza a validação e o processamento das informações para, então, armazená-las temporariamente em um banco de dados PostgreSQL local. A escolha pelo PostgreSQL se deu por sua robustez e suporte a tipos de dados complexos, útil para dados de telemetria, além de ser de código aberto (*open source*).

Para garantir o isolamento e a portabilidade da aplicação, tanto o servidor Python quanto o banco de dados rodam em containers Docker. Essa abordagem

simplifica a implantação da solução em diferentes *TV Boxes* descaracterizadas, gerenciando todas as dependências de *software* de forma contida e estável, sem interferir com o sistema operacional base. Essa arquitetura local é intrinsecamente agnóstica, podendo se conectar a qualquer tipo de sensor ou dispositivo que suporte comunicações via HTTP.

3.2 Integração Segura com a Nuvem AWS

A comunicação com o ambiente remoto foi projetada com foco em segurança e escalabilidade. Os dados consolidados localmente são enviados para um *back-end* hospedado na nuvem por meio de requisições HTTPS, assegurando a confidencialidade e integridade das informações durante o transporte. A autenticação é gerenciada por *tokens* JWT (JSON Web Tokens), um padrão amplamente utilizado para autenticação baseada em tokens. Nesse processo, a CTD autentica-se inicialmente no servidor utilizando suas credenciais e recebe um token de acesso de curta duração, que é posteriormente incluído no cabeçalho de todas as requisições subsequentes para o envio de dados.

A infraestrutura da nuvem, implementada na Amazon Web Services (AWS), utiliza uma abordagem *serverless*. Nesse modelo, as requisições enviadas pela CTD chegam inicialmente ao AWS API Gateway, que atua como um ponto de entrada seguro, realizando a validação do token de autenticação. Em seguida, o API Gateway aciona uma função AWS Lambda responsável pelo processamento inicial da requisição. Para garantir maior resiliência e desacoplamento do sistema, essa função Lambda não realiza a escrita direta no banco de dados; em vez disso, os dados são enviados para uma fila no Amazon SQS (Simple Queue Service). A partir da chegada de novas mensagens nesta fila, uma segunda função Lambda é acionada automaticamente para executar a lógica final de negócio e persistir os dados no Amazon RDS (Relational Database Service), que hospeda uma instância do PostgreSQL.

4. Resultados e Desafios Técnicos

Dentre os principais resultados, destaca-se a validação da CTD como uma orquestradora local de IoT eficiente e estável. Para tal, foram realizados testes de desempenho utilizando diferentes modelos de *TV Box* com arquiteturas ARM distintas e capacidades variadas de memória RAM. Os experimentos foram conduzidos com a ferramenta *K6* em conjunto com uma infraestrutura conteinerizada baseada em Docker Compose, simulando múltiplas requisições simultâneas ao gateway IoT. Foram definidos três cenários de operação: *smoke* (sanidade inicial), carga nominal sustentada e *stress* (saturação), utilizando respectivamente 4, 20 e 36 utilizadores virtuais (VUs) em todas as plataformas. As métricas analisadas incluíram vazão média (req/s), latências p50/p95/p99, taxa de erro, utilização média de CPU e consumo de memória RAM sob carga contínua. A Figura 1 apresenta um resumo comparativo dos resultados obtidos.

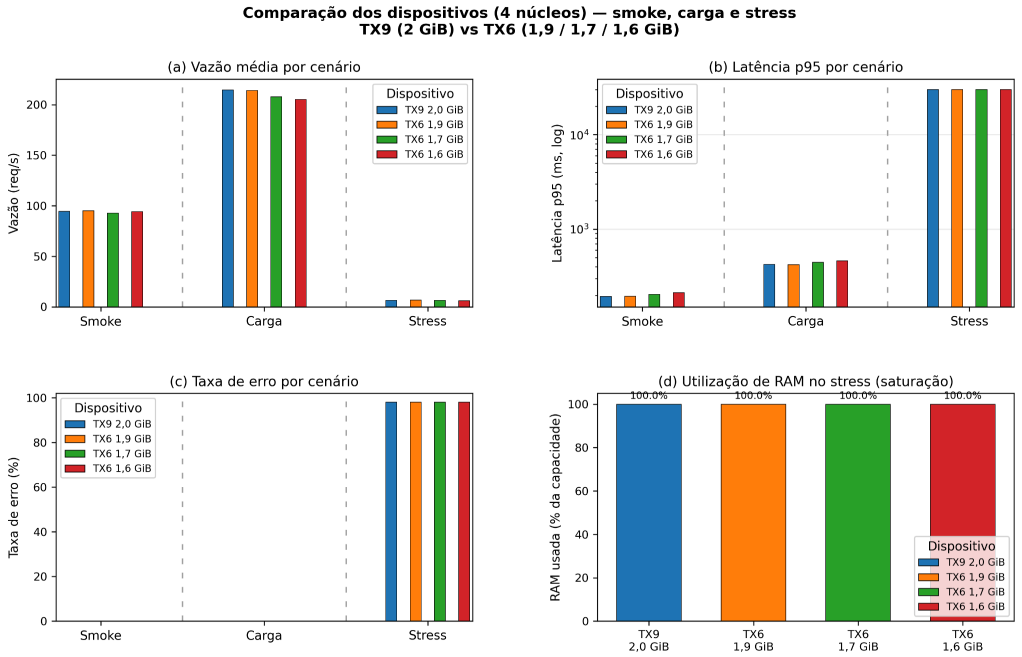

Figura 1. Prancha de testes de *smoke*, carga e *stress* em diferentes modelos de *TVbox*

Os resultados demonstraram que, nos cenários de *smoke* e carga nominal, as plataformas apresentaram comportamento semelhante em vazão e latência, confirmando a viabilidade das *TV Boxes* como centrais de tratamento de dados para aplicações IoT. Entretanto, dispositivos com menor quantidade de RAM apresentaram degradação gradual de desempenho com o aumento da concorrência e, no cenário de *stress*, observou-se saturação da memória, aumento expressivo da latência e elevada taxa de erro, evidenciando o limite operacional das plataformas. A representação da latência p95 em escala logarítmica facilitou a visualização das diferenças entre os cenários moderados e o estado de saturação.

Entre os principais desafios técnicos destacou-se a heterogeneidade dos modelos de *TV Box*, dificultando a obtenção de imagens Armbian e *kernels* compatíveis com determinados SoCs Amlogic. Problemas como ausência de *drivers*, superaquecimento e instabilidades exigiram testes com diferentes versões de *kernel* e compilação manual de *drivers*.

5. Conclusão

O projeto UAI.py demonstra a viabilidade técnica de transformar lixo eletrônico em soluções tecnológicas funcionais e de baixo custo, reutilizando *TV Boxes* como Centrais de Tratamento de Dados. A arquitetura híbrida entre processamento local e serviços em nuvem mostrou-se eficaz e escalável, reforçando o potencial da iniciativa para a democratização da tecnologia IoT e promoção da economia circular, abrindo um precedente para o reaproveitamento inteligente de dispositivos eletrônicos em larga escala. Futuramente, pretende-se validar a solução em ambientes reais, como, por exemplo, propriedades de pequenos produtores rurais, a fim de comparar os resultados já obtidos através da solução mais acessível apresentada com *hardwares* de mercado existentes.

6. Referências

ANATEL. Anatel colabora com Receita Federal no programa Além do Horizonte — Agência Nacional de Telecomunicações. Disponível em: www.gov.br. Acesso em: ago. 2023.

BRASIL. Receita Federal; Associação Brasileira de Televisão por Assinatura. Destroem mais de 97 mil aparelhos de TV Box piratas no Rio de Janeiro. 2021. Disponível em: https://www.gov.br/receitafederal/pt-br/assuntos/noticias/2021/maio/receita-federal-e-associacao-brasileira-de-televisao-por-assinatura-destroem-mais-de-97-mil-aparelhos-de-tv-box-piratas-no-rio-de-janeiro. Acesso em: ago. 2023.

BURNS, Brendan. Designing Distributed Systems. O'Reilly Media, 2018. 162 p. ISBN: 1491983647.

COULOURIS, George; DOLLIMORE, Jean; KINDBERG, Tim. Distributed Systems: Concepts and Design. 3. ed. Addison-Wesley, 2000.

FRANZ, N. M.; SILVA, C. L. Waste Electrical and Electronic Equipment (WEEE): global and contemporary challenge to production chains and the urban environment. Gestão & Produção, v. 29, e6621, 2022. DOI: https://doi.org/10.1590/1806-9649-2022v29e6621.

MELO, Alexander Bento; VIEIRA, Jardel. Desenvolvimento de uma plataforma de comunicação para ambientes IoT. In: XXXIV Simpósio Brasileiro de Telecomunicações (SBrT 2016), 2016.