Uma Abordagem Declarativa e Modular para Adaptação Dinâmica da Camada de Enlace de Redes Heterogêneas

Antônio Cleber de Sousa Araújo[1] , **Leobino N. Sampaio**[2]

[1]Instituto Federal de Educação, Ciência e Tecnologia da Bahia (IFBA)
Campus Ilhéus – Rod. Jorge Amado, KM 13 – Vila Cachoeira – Ilhéus – Bahia

[2]Programa de Pós-Graduação em Ciência da Computação (PGCOMP)
Instituto de Computação – Universidade Federal da Bahia (UFBA)
Salvador – BA – Brasil

antoniocleber@ifba.edu.br, leobino@ufba.br

Abstract. *The data link layer of contemporary networks remains tightly coupled to technology-specific mechanisms and low-level configuration models, which limits its ability to support dynamic adaptation and incremental evolution. This paper proposes a declarative and modular approach for dynamic adaptation at L2, grounded in a clear separation between declarative specification and technological realization. Through the L2i language, the approach enables applications and protocols to express link-layer communication requirements in an abstract and technology-agnostic manner. The proposal is evaluated in a realistic experimental testbed built with native Linux isolation primitives (network namespaces and veth links) and real control/data-plane mechanisms, encompassing Linux traffic control, NETCONF/YANG, and P4-based domains. The results indicate improved predictability, flow isolation, and stability under contention, at the cost of a limited and controlled operational overhead. These findings demonstrate that declarative approaches at the data link layer are both technically viable and relevant for modern network architectures.*

Resumo. *A camada de enlace das redes contemporâneas permanece fortemente acoplada a mecanismos específicos e a modelos de configuração pouco expressivos, limitando sua adaptação dinâmica e evolução incremental. Este trabalho propõe uma abordagem declarativa e modular para adaptação dinâmica em L2, baseado em uma separação explícita entre especificação declarativa e materialização tecnológica. A partir da linguagem L2i, a abordagem permite que aplicações e protocolos expressem requisitos de comunicação em L2 de forma abstrata e independente de tecnologia. A proposta foi avaliada em um testbed experimental realista, construído com primitivas nativas do Linux (namespaces de rede e enlaces veth) e componentes reais de controle e plano de dados, envolvendo domínios Linux, NETCONF/YANG e P4. Os resultados indicam maior previsibilidade, isolamento entre fluxos e estabilidade sob contenção, ao custo de um overhead operacional limitado. Esses achados demonstram que abordagens declarativas em L2 são tecnicamente viáveis e relevantes para redes modernas.*

1. Introdução

A evolução das infraestruturas de rede tem sido marcada por heterogeneidade tecnológica, diversidade de protocolos e multiplicidade de domínios administrativos. Arquiteturas orientadas a *software*, como *Software-Defined Networking* e *Intent-Based Networking*, ampliaram significativamente a flexibilidade de controle, sobretudo nas camadas de rede e transporte. Em contraste, mesmo com o avanço do *hardware* programável, a camada de enlace permanece fortemente acoplada a mecanismos específicos, interfaces procedurais e modelos de configuração de baixa expressividade, o que restringe sua capacidade de adaptação dinâmica [Patetta et al. 2022]. Esse acoplamento sustenta uma rigidez operacional frequentemente associada à "ossificação" da camada L2.

A introdução de novas políticas de qualidade de serviço ou de mecanismos avançados de comunicação, como *multicast* orientado à origem, tende a exigir modificações profundas em *pipelines*, *firmwares* e modelos proprietários. Embora a programabilidade do plano de dados represente uma alternativa para a implementação dessas políticas, o custo dessa mudança permanece elevado e pouco portável entre infraestruturas heterogêneas [Han and Li 2023]. Essa limitação torna-se evidente em ambientes que combinam inovações em diferentes camadas da pilha, como mecanismos de roteamento baseados em codificação na L2 [Dominicini et al. 2020], arquiteturas centradas em nomes [Sampaio et al. 2021] e protocolos de transporte modernos, como o QUIC [Iyengar and Thomson 2021]. Nesses contextos, a ausência de uma abstração adequada na camada de enlace dificulta a integração entre requisitos de comunicação e sua materialização tecnológica, reduzindo a portabilidade e elevando o custo de adaptação.

Com o objetivo de enfrentar essas limitações, este trabalho apresenta um *framework* modular para adaptação dinâmica na camada de enlace, estruturado em três componentes centrais: uma Camada de Especificações Declarativas (CED), um Mecanismo de Adaptação Dinâmica (MAD) e um Aplicador de Configurações (AC). Nesse modelo, aplicações e protocolos expressam seus requisitos por meio de especificações declarativas independentes de tecnologia, enquanto o *framework* se encarrega de validá-las, adaptá-las e materializá-las nos domínios de enlace disponíveis. Além disso, introduzimos a linguagem *L2i* (*Layer 2 – Intent*), uma linguagem declarativa concebida especificamente para a camada de enlace. A L2i define um espaço semântico próprio para a descrição de requisitos de comunicação, viabilizando a validação antecipada e a separação explícita entre especificação e execução.

Assim, as principais contribuições deste trabalho são: (i) a definição de um *framework* modular para adaptação dinâmica na camada de enlace; (ii) a introdução da linguagem declarativa L2i, com semântica independente de tecnologia; e (iii) a avaliação experimental da proposta em cenários representativos de comunicação *unicast* e *multicast* orientados à origem, analisando previsibilidade, isolamento entre fluxos, estabilidade sob contenção e o custo operacional do ciclo declarativo.

O restante do artigo está organizado da seguinte forma. A Seção 2 discute os trabalhos relacionados. A Seção 3 apresenta a arquitetura do *framework*. A Seção 4 descreve a implementação e o ambiente experimental. A Seção 5 analisa os resultados obtidos. Por fim, a Seção 6 conclui o trabalho e discute direções futuras.

2. Trabalhos Relacionados

A adaptação dinâmica na camada de enlace tem sido abordada por meio da reconfiguração de mecanismos específicos de L2, do provisionamento dinâmico de QoS e do suporte a comunicação *multicast*. Em arquiteturas baseadas em SDN, controladores ajustam dinamicamente os parâmetros da L2 para atender aos requisitos das aplicações [Gonzalez et al. 2023], incluindo mecanismos de *failover* e atualização incremental de *firmware* [Maurya et al. 2018, Birrittella et al. 2016]. No entanto, essas soluções permanecem fortemente acopladas à tecnologia e dependentes de controle centralizado, o que limita sua portabilidade em ambientes multidomínio.

No contexto de QoS dinâmico, propostas adaptam diretamente filas, políticas de escalonamento e regras de encaminhamento conforme variações de tráfego ou demandas das aplicações [Xue et al. 2020]. Abordagens voltadas à comunicação *multicast* exploram a provisão dinâmica de QoS e otimizações de gerenciamento de grupos [Gatouillat et al. 2018, Huang et al. 2024], mas permanecem focadas em mecanismos específicos, sem tratar de forma sistemática a adaptação multidomínio ou o *multicast* orientado à origem.

Arquiteturas orientadas a intenção elevam o nível de abstração do controle de redes por meio de linguagens de alto nível, como LUMI [Jacobs et al. 2021], ou abordagens baseadas em *Large Language Models* (LLMs) [Mekrache and Ksentini 2024]. Esses esforços concentram-se nas camadas superiores da pilha e não oferecem mecanismos explícitos para capturar e adaptar requisitos semanticamente ricos na camada de enlace.

Em síntese, embora o estado da arte avance na flexibilização do controle e na elevação do nível de abstração, requisitos, lógica de adaptação e tecnologia permanecem fortemente acoplados na L2. Diferentemente dessas abordagens, este trabalho propõe uma adaptação declarativa na camada de enlace, separando explicitamente a especificação da materialização. A próxima seção apresenta a arquitetura proposta.

3. *Framework* de Adaptação Dinâmica na Camada de Enlace

Este trabalho propõe um *framework* modular para adaptação dinâmica na camada de enlace, concebido para tratar a L2 como um espaço semântico explícito, no qual requisitos de comunicação podem ser declarados, validados e adaptados de forma sistemática e independente da tecnologia subjacente. Diferentemente de abordagens que reduzem a camada de enlace a um conjunto de mecanismos de encaminhamento, filas ou parâmetros configuráveis, a proposta assume que a L2 pode incorporar um modelo declarativo próprio, capaz de mediar, de forma estruturada, a relação entre protocolos das camadas superiores e domínios tecnológicos heterogêneos.

O princípio central do *framework* é a separação explícita entre *intenção* e *execução*. Protocolos e aplicações expressam *"o que"* deve ser garantido em termos de comunicação, enquanto o sistema decide *"como"* essas garantias serão materializadas em cada domínio de enlace, considerando capacidades locais, políticas administrativas e restrições operacionais. Essa separação não visa ocultar a complexidade da L2, mas torná-la governável por meio de abstrações semânticas estáveis, mitigando a rigidez associada à "ossificação" dessa camada, mesmo em ambientes que incorporam planos de dados programáveis ou mecanismos avançados de controle.

3.1. Visão Geral da Arquitetura

A Figura 1 apresenta uma visão geral da arquitetura proposta, destacando seus principais componentes e o fluxo lógico de processamento das intenções declaradas. Conceitualmente, o *framework* posiciona-se entre as camadas de enlace e rede, atuando como uma camada intermediária responsável por interpretar, adaptar e aplicar requisitos de comunicação na L2 de forma transparente para as camadas superiores.

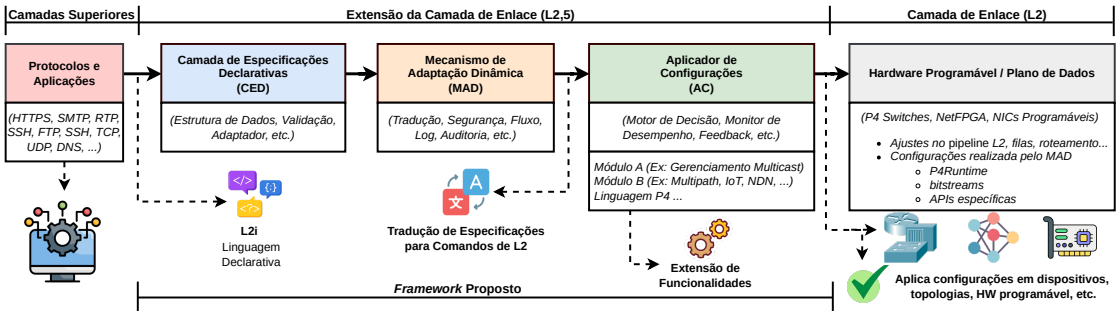


Figura 1. Visão geral do *framework* de adaptação dinâmica na camada de enlace. A arquitetura explicita o ciclo orientado a intenção, separando a especificação declarativa (L2i), a adaptação semântica multidomínio (MAD) e a materialização tecnológica (AC). Os domínios de execução ilustrados representam exemplos de tecnologias suportadas, não constituindo dependência estrutural da proposta.

Nesse modelo, aplicações e protocolos não interagem diretamente com comandos específicos, modelos YANG ou estruturas internas de *pipelines* programáveis. Em vez disso, expressam seus requisitos por meio de especificações declarativas que capturam as propriedades desejadas de comunicação. A partir dessas especificações, o *framework* coordena os recursos disponíveis nos diferentes domínios de enlace, assegurando que a intenção original seja preservada ao longo de todo o processo de adaptação, mesmo diante da heterogeneidade tecnológica e das variações dinâmicas de carga.

A arquitetura foi concebida de forma modular, permitindo evolução incremental e integração de novas tecnologias sem reengenharia completa. Essa característica é essencial em ambientes multidomínio, nos quais diferentes segmentos da rede podem empregar mecanismos de controle distintos ou estratégias divergentes para implementar funcionalidades semanticamente equivalentes.

3.2. Componentes Funcionais do *Framework*

O *framework* é estruturado em três componentes funcionais principais: a Camada de Especificações Declarativas, o Mecanismo de Adaptação Dinâmica e o Aplicador de Configurações. Essa organização reflete uma separação conceitual clara entre especificação, adaptação e execução, favorecendo o desacoplamento tecnológico e a clareza arquitetural, sem impor dependência de soluções ou plataformas específicas.

3.2.1. Camada de Especificações Declarativas (CED) e a L2i

A CED constitui o ponto de entrada do *framework* e é materializada, neste trabalho, pela linguagem declarativa L2i. A L2i foi concebida como uma linguagem específica para

a camada de enlace, capaz de expressar requisitos de comunicação de forma abstrata, validável e independente de tecnologia.

Por meio da L2i, aplicações e protocolos das camadas superiores podem declarar propriedades desejadas, como limites percentílicos de latência, banda mínima garantida, prioridades relativas e associações a grupos de comunicação *multicast*. Essas especificações são expressas em formato estruturado, com tipagem explícita e unidades normalizadas, sem qualquer referência direta a mecanismos concretos de execução, como filas de escalonamento, modelos NETCONF/YANG ou estruturas internas de *pipelines* P4.

A L2i não se limita a encapsular interfaces existentes ou a fornecer uma camada sintática adicional. Trata-se de uma linguagem com semântica própria, cuja gramática define tanto a estrutura quanto as restrições fundamentais das intenções declaradas. Esse modelo permite a validação antecipada, reduzindo a propagação de erros e estabelecendo um contrato claro entre a especificação e a execução. Embora esta instância da L2i suporte explicitamente parâmetros de latência, largura de banda, prioridade e *multicast*, a linguagem foi concebida para evolução incremental, permitindo a incorporação futura de métricas adicionais sem impacto estrutural no *framework*.

3.2.2. Mecanismo de Adaptação Dinâmica (MAD)

O Mecanismo de Adaptação Dinâmica atua como o elo entre a especificação declarativa e a execução concreta. Sua função é interpretar as intenções expressas em L2i e traduzi-las para uma representação intermediária independente de tecnologia, preservando os elementos semânticos centrais da intenção. As decisões de adaptação consideram simultaneamente os requisitos expressos e as capacidades dos domínios de enlace disponíveis, podendo envolver políticas de escalonamento, estratégias de priorização ou mecanismos de replicação *multicast*. Essas escolhas não são impostas pela linguagem, mas derivadas do contexto de execução, o que reforça o caráter não prescritivo da proposta e permite que diferentes domínios adotem estratégias distintas, desde que a semântica da intenção seja preservada. Essa flexibilidade é fundamental para ambientes heterogêneos e multidomínio, nos quais a uniformização de mecanismos é, em geral, inviável.

3.2.3. Aplicador de Configurações (AC)

O AC constitui o núcleo executor do *framework*. A partir das instruções produzidas pelo MAD, o AC materializa a intenção declarativa nos domínios tecnológicos concretos, interagindo diretamente com os mecanismos de controle disponíveis em cada ambiente. Dependendo do domínio, essa materialização pode envolver a geração de comandos de *traffic control* em sistemas Linux, a emissão de operações `<edit-config>` via NETCONF/YANG ou a inserção dinâmica de regras em *pipelines* programáveis baseados em P4. Em todos os casos, o AC preserva o desacoplamento entre o núcleo do *framework* e os detalhes específicos de cada tecnologia, permitindo que a mesma intenção seja realizada por mecanismos distintos sem alteração da especificação original.

O AC incorpora ainda mecanismos de observação e *feedback*, que permitem verificar a aplicação efetiva das configurações e coletar informações relevantes para adaptações

subsequentes. Esse ciclo estabelece as bases para extensões futuras, como otimização global ou adaptação preditiva, sem comprometer os princípios fundamentais do modelo.

3.3. Síntese da Proposta

Em conjunto, a L2i, o MAD e o AC formam um *framework* coerente para adaptação dinâmica na camada de enlace. A proposta distingue-se por tratar a L2 como um espaço semântico explícito, no qual intenções podem ser expressas, validadas e adaptadas de forma independente da tecnologia subjacente. Ao posicionar a L2i como a materialização concreta da camada declarativa do *framework*, este trabalho estabelece uma base sólida para integrar tecnologias legadas e programáveis, preservando a transparência para as camadas superiores e favorecendo a evolução incremental da infraestrutura. A próxima seção descreve a implementação da proposta e sua avaliação experimental em um ambiente de emulação realista, permitindo analisar empiricamente as propriedades arquiteturais discutidas até aqui.

4. Implementação e Avaliação Experimental

Esta seção descreve a implementação do *framework* proposto e o método de avaliação experimental adotado para analisar sua aplicabilidade em ambientes heterogêneos de camada de enlace. O foco não reside na otimização de desempenho isolado, mas na verificação empírica das propriedades arquiteturais discutidas ao longo do artigo, em particular o desacoplamento entre especificação declarativa e materialização tecnológica, a coerência multidomínio e o custo associado ao ciclo de adaptação dinâmica.

A avaliação foi conduzida em um *testbed* experimental realista e controlado, construído com primitivas nativas do Linux, sobre os quais foram instanciados domínios de enlace com mecanismos reais de controle e encaminhamento. Esse arranjo permite observar o comportamento efetivo de tecnologias empregadas em produção, fornecendo subsídios concretos para a análise apresentada na Seção 5.

4.1. Ciclo de Execução Orientado a Intenção

O ciclo de execução envolve a especificação da intenção, validação sintática e semântica, adaptação às capacidades dos domínios e materialização das configurações, correspondendo às etapas de *intent specification* até *intent realization*, com verificação contínua de conformidade.

Aplicações ou protocolos das camadas superiores expressam requisitos na CED por meio da linguagem L2i. Cada especificação define garantias de comunicação na camada de enlace, como limites percentílicos de latência, banda mínima, prioridades relativas ou escopo *multicast*, sendo previamente validada para assegurar consistência semântica e evitar configurações inválidas.

O MAD interpreta a intenção e a traduz para uma representação intermediária independente de tecnologia, considerando os requisitos e as capacidades de cada domínio. Em seguida, o Aplicador de Configurações materializa a intenção nos domínios envolvidos. O fluxo é ilustrado na Figura 2, evidenciando a separação entre a especificação declarativa, a adaptação semântica e a execução tecnológica.

Para fins metodológicos, a etapa de materialização pode operar em duas modalidades. Em uma, a realização é efetivamente aplicada ao plano de dados, de forma a permitir

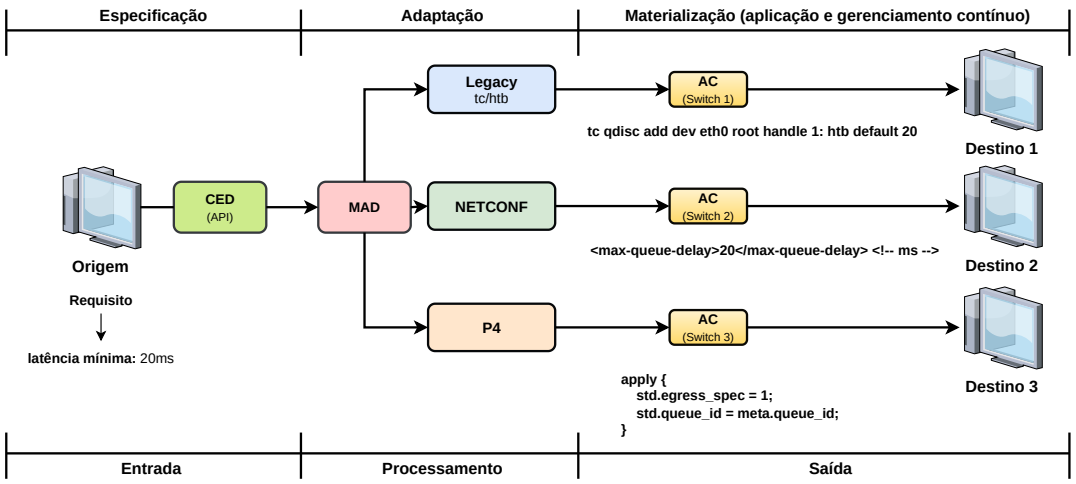


Figura 2. Fluxo de execução do *framework*, evidenciando a separação entre especificação declarativa (L2i), adaptação semântica (MAD) e materialização tecnológica (AC) em domínios Linux, NETCONF e P4, executados em ambiente de emulação realista.

observar os efeitos concretos da adaptação sobre mecanismos reais de controle e encaminhamento. Na outra, a intenção é validada e adaptada, mas a aplicação é suprimida, permitindo isolar a lógica de adaptação sem interferir no plano de dados. A formalização dessas modalidades, bem como sua combinação com os modos operacionais considerados na avaliação, é apresentada na Subseção 4.5.

4.2. Gramática, Identificação de Fluxos e Validação

A L2i é uma linguagem declarativa completa para a camada de enlace, cuja gramática, baseada em EBNF (*Extended Backus-Naur Form*), foi concebida com foco em expressividade controlada e verificabilidade. A linguagem organiza os requisitos em classes semânticas independentes, como latência, largura de banda, prioridade e *multicast*, permitindo sua composição em uma única intenção declarativa. A estrutura adota um formato JSON canônico, com tipagem explícita, unidades normalizadas e restrições bem definidas, o que viabiliza validação estática automatizada. Uma descrição melhor da L2i é disponibilizada no repositório público do projeto.

Cada intenção referencia um identificador lógico de fluxo, instanciado como um ULID. Essa escolha garante unicidade global, ordenação temporal e atomicidade durante operações de criação, atualização ou remoção de intenções, permitindo correlação precisa entre especificações declarativas, métricas observadas e artefatos de configuração gerados ao longo dos experimentos.

O processo de validação ocorre em duas etapas complementares. A validação sintática verifica a conformidade estrutural com a gramática e o esquema da linguagem. Em seguida, a validação semântica aplica regras globais, incluindo a consistência interna da intenção, a coerência do escopo *multicast*, faixas válidas de valores e regras de composição entre múltiplas intenções. Apenas intenções semanticamente válidas avançam para a fase de adaptação, o que é essencial para interpretar corretamente os resultados discutidos na Seção 5.

4.3. Ambiente de Emulação e Instrumentação

Os experimentos foram executados em um *testbed* experimental controlado, no qual cada domínio de enlace é instanciado com recursos nativos do Linux, incluindo *namespaces* de rede e enlaces virtuais (*veth*). Esse arranjo permite definir topologias multidomínio, parametrizar enlaces com largura de banda e atraso, e instrumentar fluxos de tráfego com precisão. Por utilizar mecanismos reais do sistema operacional, o *testbed* possibilita avaliar o comportamento efetivo de políticas de controle e encaminhamento.

A implementação do *framework* foi realizada no Ubuntu Server 24.04 LTS, com *kernel* 6.8.0-87. O domínio Linux utiliza *Linux traffic control* para controle de filas e políticas de QoS. O domínio baseado em gerenciamento declarativo é implementado por meio de uma pilha NETCONF/YANG composta por `sysrepo`, `Netopeer2` e `libnetconf2`. O domínio de plano de dados programável baseia-se no *behavioral model* (bmv2), com suporte a *P4Runtime* para inserção dinâmica de regras.

Para a instrumentação experimental, foram utilizadas ferramentas amplamente consolidadas. O `fping` foi empregado para a coleta de RTTs percentílicos, permitindo a análise do comportamento de cauda, enquanto o `iperf3` foi utilizado para a medição de vazão sustentada sob diferentes níveis de contenção. A avaliação privilegia métricas percentílicas porque garantias determinísticas não são factíveis em sistemas reais sob concorrência e reconfiguração dinâmica. Estas métricas fornecem subsídios diretos para avaliar previsibilidade, isolamento entre fluxos e estabilidade.

4.4. Cenários Experimentais

A avaliação foi estruturada em dois cenários complementares, projetados para explorar dimensões distintas da proposta: comunicação *unicast* multidomínio com requisitos explícitos de QoS (cenário S1) e comunicação *multicast* orientada à origem em ambiente multidomínio (cenário S2).

No cenário S1, modela-se a comunicação *unicast* atravessando três domínios administrativos interconectados sequencialmente. O domínio intermediário é configurado como um gargalo natural, de modo a induzir uma contenção controlada. A intenção declarativa expressa requisitos de latência percentílica, banda mínima e prioridade elevada.

O cenário S2 estende a avaliação para comunicação *multicast* orientada à origem. Nesse caso, a complexidade da infraestrutura intermediária é intencionalmente abstraída para enfatizar o controle exercido no domínio emissor e a capacidade do *framework* de preservar a semântica da intenção sob eventos dinâmicos, como *join* e *leave*. A especificação declarativa ativa explicitamente o *multicast* e define requisitos mínimos de comunicação, sem impor algoritmos específicos de construção de árvores.

As topologias lógicas utilizadas nos dois cenários são apresentadas na Figura 3. As especificações utilizadas, bem como exemplos, arquivos correspondentes e detalhes adicionais de implementação, estão disponíveis no repositório público do projeto.

4.5. Modos de Operação e Tipos de *Backend*

A avaliação considera duas dimensões complementares. A primeira distingue o comportamento do sistema quando não há adaptação e quando o ciclo declarativo é efetivamente

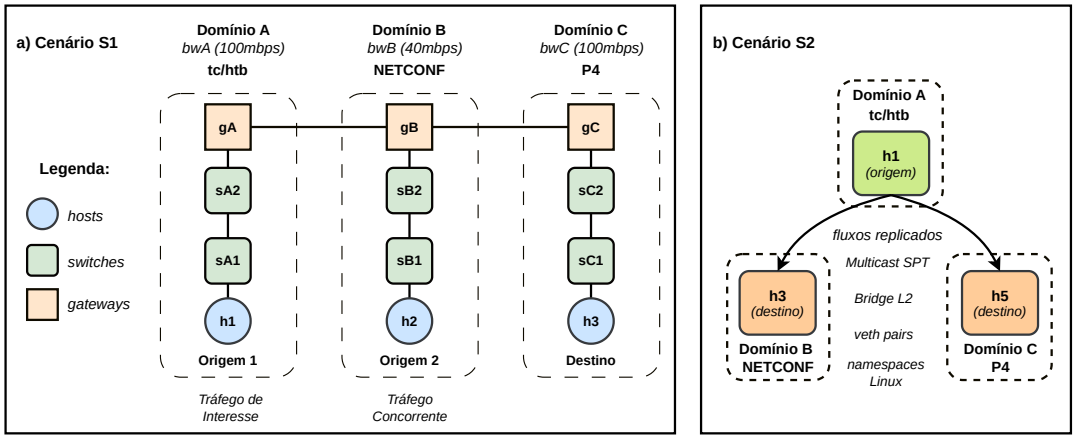


Figura 3. Topologias experimentais dos cenários S1 e S2. Em S1, três domínios administrativos interconectados evidenciam heterogeneidade de capacidade e gargalo intermediário. Em S2, um ambiente multicast orientado à origem abstrai a infraestrutura intermediária para enfatizar controle e adaptação no domínio emissor.

executado. A segunda distingue se a fase de aplicação altera, ou não, o plano de dados. Essa separação é necessária para interpretar os resultados de forma causal, evitando atribuir à infraestrutura efeitos que decorrem da lógica de adaptação, ou vice-versa.

A avaliação considera dois modos operacionais. No modo *baseline*, o ambiente emulado opera com configurações estáticas e comportamento de melhor esforço, sem intervenção do *framework*. No modo *adapt*, o *framework* processa as intenções expressas em L2i e realiza a adaptação dinâmica, permitindo avaliar os efeitos da separação entre especificação e execução.

Adicionalmente, a implementação suporta dois tipos de *backend*. O *backend mock* valida e adapta semanticamente as intenções sem alterar o plano de dados, sendo útil para verificação funcional. O *backend real* materializa efetivamente as configurações nos domínios Linux, NETCONF e P4, permitindo observar os efeitos concretos da adaptação em mecanismos reais. Essa distinção é fundamental para interpretar os resultados apresentados na Seção 5.

4.6. Reprodutibilidade

Todos os experimentos são integralmente reprodutíveis. O repositório público disponibiliza o código-fonte do *framework*, os esquemas da linguagem L2i, as especificações utilizadas, os *scripts* de execução e os artefatos gerados. Essa organização permite a reprodução fiel dos resultados apresentados, bem como a validação independente e a extensão da proposta para novos cenários, tecnologias e requisitos declarativos.

5. Resultados e Análise

Esta seção analisa os resultados obtidos a partir da execução dos cenários experimentais descritos na Seção 4. A análise é conduzida à luz do ciclo de execução orientado a intenção e das dimensões metodológicas definidas, em particular os modos de operação (*baseline* e *adapt*) e a materialização efetiva das intenções no plano de dados.

Diferentemente de avaliações centradas em métricas isoladas de desempenho, os resultados são interpretados com foco nas propriedades estruturais da proposta. Em especial, investigam-se a preservação da semântica da intenção sob contenção, a estabilidade do comportamento diante de eventos dinâmicos, a coerência da adaptação em ambientes multidomínio heterogêneos e o custo operacional associado ao ciclo declarativo.

As evidências são organizadas de forma incremental e complementar, conforme sintetizado nas Figuras 1 e 2 e nas Tabelas 1 e 2. Em conjunto, os resultados sustentam a tese central do trabalho: abordagens declarativas na camada de enlace são tecnicamente viáveis, introduzem previsibilidade e isolamento entre fluxos e impõem um *overhead* limitado e previsível, compatível com redes modernas e com a evolução incremental da infraestrutura.

5.1. Conformidade da Intenção sob Contenção Multidomínio

A primeira evidência empírica avalia a capacidade do *framework* de preservar a conformidade das intenções declaradas sob diferentes níveis de contenção no cenário S1. A Tabela 1 sintetiza os resultados obtidos para combinações representativas de condições de carga no domínio intermediário, comparando os modos *baseline–real* e *adapt–real*.

Na tabela, `bwB` (Mbps) representa a largura de banda disponível no domínio B, configurado como gargalo da topologia. O parâmetro `BE` (Mbps) indica a taxa de tráfego concorrente de melhor esforço injetada nesse domínio, modelando diferentes níveis de contenção. As colunas *Baseline* e *Adapt* indicam se a intenção declarada foi satisfeita ao final da execução, considerando exclusivamente o plano de dados real.

Tabela 1. Conformidade da intenção no cenário S1 sob diferentes níveis de contenção.

bwB (Mbps)	BE (Mbps)	Baseline	Adapt
10	0	Sim	Sim
10	20	Não	Sim
25	0	Sim	Sim
25	50	Não	Sim

Observa-se que, na ausência de contenção significativa, ambos os modos preservam a conformidade da intenção. Contudo, à medida que o tráfego concorrente de melhor esforço se aproxima ou excede a capacidade do domínio gargalo, o modo *baseline* passa a violar os requisitos declarados, enquanto o modo *adapt* mantém a conformidade plena em todos os cenários avaliados.

Esse comportamento não decorre de otimizações pontuais de desempenho, mas da introdução de mecanismos explícitos de coordenação e isolamento na camada de enlace, acionados a partir da interpretação da intenção declarativa. A Tabela 1 evidencia, assim, que o valor central da proposta reside na previsibilidade do comportamento sob contenção, e não na maximização de métricas isoladas.

5.2. Estabilidade, Recuperação e Contenção sob Eventos Dinâmicos de *Multicast*

A segunda evidência experimental avalia o comportamento do *framework* sob eventos dinâmicos no cenário S2, de comunicação *multicast* orientada à origem em ambiente multi-

Tabela 2. Estatísticas de dispersão (min, mediana, max e IQR) para os tempos observados no cenário S1. A duração total é apresentada em segundos (baseline vs adapt) e os tempos por domínio em milissegundos (modo adapt).

Métrica	Unid.	Min	Mediana	Max	IQR
Duração total (baseline)	s	69.282	69.432	69.845	0.164
Duração total (adapt)	s	70.260	70.628	71.364	0.190
tc/htb (adapt)	ms	102.9	115.6	119.5	14.0
NETCONF (adapt)	ms	939.3	1297.7	1796.2	469.2
P4 (adapt)	ms	54.8	67.3	162.1	55.4

domínio de camada de enlace. Diferentemente do cenário S1, o foco não é no desempenho absoluto, mas na capacidade de manter e restabelecer a conformidade semântica diante de perturbações, especialmente eventos de *join multicast*. A análise inclui tráfego concorrente de melhor esforço (BE), em Mbps, utilizado como contenção controlada. Esse tráfego compartilha os mesmos domínios de enlace e permite observar simultaneamente a robustez do ciclo declarativo e seus efeitos sobre o restante da rede. Define-se o tempo de recuperação como o *tempo até conformidade estável*, isto é, o intervalo entre o término do *join* e o primeiro instante em que a intenção passa a ser satisfeita de forma sustentada por múltiplas janelas de observação.

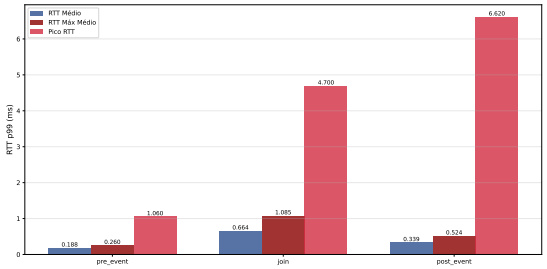
A Figura 4a mostra esse tempo em função da intensidade do BE: quanto maior a contenção, maior o tempo de recuperação, embora sempre claramente delimitado, sem instabilidade prolongada. A Figura 4b apresenta as fases *pre-event*, *join* e *post-event*, fornecendo o contexto temporal para separar regime estável, perturbação e recuperação. Por fim, a Figura 4c mostra a latência do fluxo sob intenção (RTT p99) em janelas de 500 ms. Os picos coincidem com o *join* e maior contenção BE, seguidos de rápida convergência aos valores esperados. A adaptação não elimina a contenção, mas a absorve de forma controlada, preservando a semântica do fluxo prioritário.

Em conjunto, os resultados indicam que o ciclo declarativo preserva a intenção em regime dinâmico e sob alta contenção. A recuperação rápida e mensurável reforça que a separação entre especificação declarativa e materialização tecnológica favorece a estabilidade e a previsibilidade em ambientes multidomínio de camada de enlace.

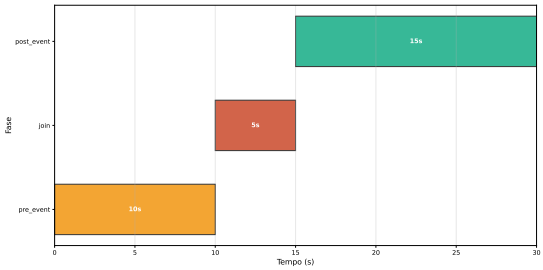
5.3. *Overhead* do Ciclo Declarativo

Uma crítica recorrente às abordagens declarativas refere-se ao custo potencial associado à interpretação e adaptação das intenções. Para analisar esse aspecto, a Tabela 2 apresenta estatísticas de dispersão do tempo total de execução no cenário S1, comparando os modos *baseline* e *adapt*, bem como os tempos associados à atuação do plano de controle nos domínios tecnológicos, considerados apenas no modo *adapt*.

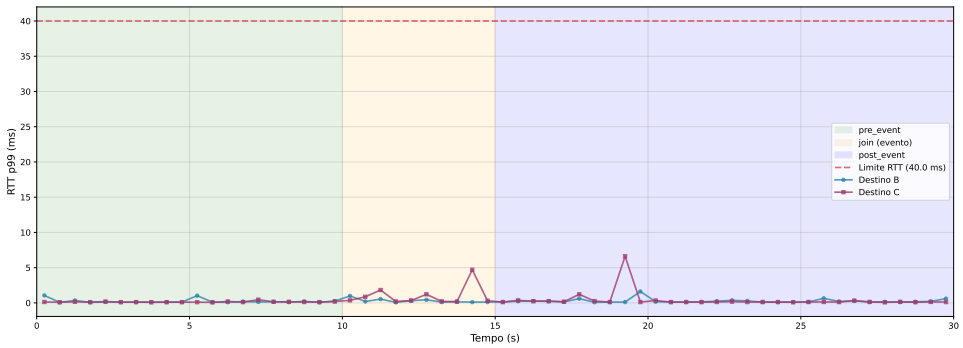
O tempo total no modo *baseline* reflete exclusivamente a execução do experimento sem intervenção do *framework*. Por essa razão, não há decomposição por domínio nesse modo, uma vez que não ocorre atuação explícita do plano de controle. No modo *adapt*, o aumento temporal observado decorre do ciclo declarativo completo, que inclui a interpretação da especificação L2i, a adaptação semântica das intenções e a aplicação coordenada das configurações nos domínios de enlace.



(a) Tempo até conformidade estável em função do tráfego concorrente de melhor esforço (BE).



(b) Linha do tempo do experimento S2, destacando as fases *pre-event*, *join* e *post-event*.



(c) Evolução temporal do RTT p99 do fluxo sob intenção ao longo do experimento, evidenciando os efeitos da contenção do tráfego concorrente e o processo de recuperação após o evento de *join multicast*.

Figura 4. Estabilidade, recuperação e contenção sob eventos dinâmicos de *join multicast* no cenário S2.

Os tempos associados a *tc/htb*, NETCONF e P4 correspondem à atuação do plano de controle em cada domínio. No domínio P4, esse tempo inclui a serialização das mensagens P4Runtime, a comunicação via gRPC, o processamento no servidor P4Runtime do BMv2, a escrita de tabelas, contadores e medidores, bem como o recebimento da confirmação de aplicação. Trata-se, portanto, do intervalo entre o envio do comando e a confirmação de sua aplicação no *switch*. O menor tempo observado nesse domínio reflete o caráter altamente otimizado do pipeline P4 no plano de dados, cujo processamento ocorre de forma determinística e com baixa latência.

Os valores associados ao domínio NETCONF refletem a natureza transacional e orientada a estado do protocolo, que envolve validação, persistência e sincronização de configurações. Ainda assim, os tempos permanecem estáveis e compatíveis com ambientes reais. No conjunto, o aumento introduzido pelo ciclo declarativo é da ordem de poucos segundos, apresentando baixa variabilidade entre execuções e comportamento previsível.

Esse *overhead* não caracteriza uma degradação indesejável do sistema, mas o custo esperado da mediação semântica necessária para viabilizar o desacoplamento tecnológico, a preservação de conformidade e a estabilidade sob contenção. A Tabela 2 sustenta, assim, a hipótese de que o *framework* proposto impõe um custo operacional controlado e compatível com ambientes reais, neutralizando a crítica de inviabilidade prática frequentemente associada a abordagens declarativas na camada de enlace.

5.4. Síntese dos Resultados

Em conjunto, os resultados indicam que a proposta introduz maior previsibilidade, isolamento entre fluxos e estabilidade sob contenção, ao custo de um *overhead* operacional limitado e previsível. Esses achados demonstram que abordagens declarativas na camada de enlace são tecnicamente viáveis, cientificamente relevantes e compatíveis com a evolução incremental de infraestruturas heterogêneas. Mais do que ganhos pontuais de desempenho, os experimentos evidenciam propriedades estruturais emergentes da separação entre intenção e execução, posicionando o *framework* proposto como uma base sólida para a adaptação dinâmica da camada de enlace em redes modernas.

6. Conclusão e Trabalhos Futuros

Os resultados obtidos nos experimentos demonstram a preservação da previsibilidade, isolamento entre fluxos e a estabilidade sob contenção e reconfiguração dinâmica. A linguagem declarativa L2i mostrou-se adequada para capturar requisitos como latência percentílica, banda mínima, prioridade e escopo *multicast*. O ciclo de adaptação preservou semanticamente essas intenções em ambientes multidomínio heterogêneos. A avaliação em emulação realista indica que os benefícios decorrem da separação entre especificação e execução, com custo limitado e previsível, compatível com cenários reais. Do ponto de vista estrutural, o *framework* não impõe dependência de tecnologias, algoritmos ou mecanismos específicos, permitindo diferentes estratégias de materialização sem violar a intenção declarada.

Como trabalhos futuros, destacam-se a implantação em infraestruturas físicas, a inclusão de métricas como *jitter* e confiabilidade, e mecanismos formais para tratar conflitos entre intenções concorrentes, ampliando a autonomia do modelo sem comprometer seus princípios de desacoplamento.

Disponibilidade de Artefatos

Todo o material associado a este trabalho está disponível em repositório público[1]. Abrangendo código-fonte, especificações da L2i, *scripts* de execução, cenários experimentais e artefatos gerados. O projeto é disponibilizado sob licença Apache 2.0, permitindo uso, modificação e redistribuição. Contribuições externas são incentivadas, com o objetivo de ampliar a cobertura de cenários, tecnologias e requisitos suportados.

Agradecimentos

Os autores agradecem o apoio do Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) e da Fundação de Amparo à Pesquisa do Estado da Bahia (FAPESB). Este material é baseado em trabalho apoiado pelo Escritório de Pesquisa Básica da Força Aérea (*Air Force Office of Scientific Research*) dos EUA sob o número de concessão FA9550-23-1-0631.

Referências

Birrittella, M. S., Debbage, M., Huggahalli, R., Kunz, J., Lovett, T., Rimmer, T., Underwood, K. D., and Zak, R. C. (2016). Enabling scalable high-performance systems with the intel omni-path architecture. *IEEE Micro*, 36(4):38–47.

[1] https://github.com/cleberaraujo/link_layer_intent

Dominicini, C., Mafioletti, D., Locateli, A. C., Villaca, R., Martinello, M., Ribeiro, M., and Gorodnik, A. (2020). Polka: Polynomial key-based architecture for source routing in network fabrics. In *2020 6th IEEE Conference on Network Softwarization (NetSoft)*, pages 326–334.

Gatouillat, A., Badr, Y., and Massot, B. (2018). Qos-driven self-adaptation for critical iot-based systems. In *Service-Oriented Computing – ICSOC 2017 Workshops: ASOCA, ISyCC, WESOACS, and Satellite Events, Málaga, Spain, November 13–16, 2017, Revised Selected Papers*, page 93–105, Berlin, Heidelberg. Springer-Verlag.

Gonzalez, L. F., Vidal, I., Valera, F., Martin, R., and Artalejo, D. (2023). A link-layer virtual networking solution for cloud-native network function virtualisation ecosystems: L2s-m. *Future Internet*, 15(8).

Han, L. and Li, R. (2023). On the study of internet ossification, impacts, and solutions. In *International Journal on Advances in Networks and Services, vol 16 no 3 & 4, year 2023*, pages 53–62, Berlin, Heidelberg. Springer-Verlag.

Huang, J., Chen, Z., Wang, Y., Li, H., Li, Z., Wang, Q., Li, S., He, Z., and Jiang, W. (2024). Achieving high efficiency for datacenter multicast using skewed bloom filter. In *Proceedings of the 53rd International Conference on Parallel Processing*, ICPP '24, page 1227–1236, New York, NY, USA. Association for Computing Machinery.

Iyengar, J. and Thomson, M. (2021). QUIC: A UDP-Based Multiplexed and Secure Transport. RFC 9000.

Jacobs, A. S., Pfitscher, R. J., Ribeiro, R. H., Ferreira, R. A., Granville, L. Z., Willinger, W., and Rao, S. G. (2021). Hey, lumi! using natural language for Intent-Based network management. In *2021 USENIX Annual Technical Conference (USENIX ATC 21)*, pages 625–639. USENIX Association.

Maurya, S., Tiwari, N. K., and Gupta, S. C. (2018). Layered software defined networking. In Janyani, V., Tiwari, M., Singh, G., and Minzioni, P., editors, *Optical and Wireless Technologies*, pages 351–362, Singapore. Springer Singapore.

Mekrache, A. and Ksentini, A. (2024). Llm-enabled intent-driven service configuration for next generation networks. In *2024 IEEE 10th International Conference on Network Softwarization (NetSoft)*, pages 253–257.

Patetta, M., Secci, S., and Taktak, S. (2022). A lightweight southbound interface for standalone p4-netfpga smartnics. In *2022 1st International Conference on 6G Networking (6GNet)*, pages 1–4.

Sampaio, L., Freitas, A., Brito, I., Araújo, F., and Ribeiro, A. (2021). Revisitando as ICNs: Mobilidade, Segurança e Aplicações Distribuídas através das Redes de Dados Nomeados. *Minicursos do XXXIX Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*, pages 51–100.

Xue, X., Pan, B., Agraz, F., Pagès, A., Guo, X., Yan, F., Spadaro, S., and Calabretta, N. (2020). Sdn-enabled reconfigurable optical data center network with automatic network slicing to provision dynamic qos. In *2020 European Conference on Optical Communications (ECOC)*, pages 1–4.