EX-JUD: Uma Plataforma Blockchain para Automação, Auditoria e Rastreabilidade de Serviços Extrajudiciais no Tribunal de Justiça de Mato Grosso do Sul

Renato Gil Arruda Vieira
Tribunal de Justiça de Mato Grosso do Sul
Campo Grande, Brasil
renato.vieira@tjms.jus.br

Dr. Marcelo Augusto Santos Turine
FACOM/Universidade Federal de Mato Grosso do Sul
Campo Grande, Brasil
marcelo.turine@ufms.br

Abstract

Os cartórios extrajudiciais frequentemente operam com sistemas legados e centralizados, o que compromete a transparência dos serviços e dificulta significativamente a auditoria dos repasses de emolumentos institucionais. A adoção de arquiteturas descentralizadas e imutáveis destaca-se como o principal meio para lidar com esses problemas. Este artigo apresenta a ferramenta EX-JUD, uma plataforma baseada em *blockchain* permissionada, amparada por microsserviços para certificação e rastreabilidade de atos cartorários extrajudiciais, visando à governança digital. Descrevemos a arquitetura da solução, que protege a privacidade dos dados sensíveis por meio de coleções privadas (aderência à LGPD), previne o gasto duplo de selos digitais e fornece uma infraestrutura tolerante a falhas bizantinas. Apresentamos o cenário de uso da ferramenta em serventias físicas, comparativos com sistemas existentes e o impacto da plataforma como instrumento de apoio à implantação e validação de processos de Engenharia de Software aplicados à governança eletrônica.

Keywords

Blockchain, Hyperledger Fabric, Microsserviços, Poder Judiciário, Cartórios Extrajudiciais

1 Introdução e Motivação

A transformação digital das estruturas estatais no Brasil, impulsionada no ecossistema do Poder Judiciário por políticas institucionais como o Programa Justiça 4.0 [6] e a Plataforma Digital do Poder Judiciário Brasileiro (PDPJ-Br) [7], modificou radicalmente o desenvolvimento de sistemas públicos. Contudo, enquanto o fluxo processual avança em direção à interoperabilidade nativa, o universo dos serviços extrajudiciais — delegados a serventias notariais e registrais — ainda opera de forma isolada, dependendo de arquiteturas legadas que comprometem a fiscalização em tempo real.

Nesse cenário, as Corregedorias-Gerais de Justiça (CGJ) dos Tribunais Estaduais exercem um papel institucional crítico e complexo. Compete exclusivamente às Corregedorias a fiscalização, regulação e controle de todos os atos praticados pelos cartórios [2, 3].

1.1 O Gargalo da Auditabilidade Centralizada

Apesar da relevância socioeconômica dessa fiscalização, a infraestrutura tecnológica disponível nas últimas décadas tem se mostrado insuficiente. O gerenciamento de selos de fiscalização eletrônicos apoia-se, historicamente, em plataformas monolíticas dependentes de bancos de dados relacionais centralizados (como o sistema legado SIG-EX do TJMS [12]).

- **Silos de Confiança e Vulnerabilidade Administrativa:** Em bancos relacionais tradicionais, a segurança lógica e a imutabilidade dependem da integridade de administradores centrais. A ausência de um mecanismo de consenso distribuído impede que o Tribunal, os Cartórios e as demais entidades de controle fiscal (como o Ministério Público e a Defensoria Pública) compartilhem uma visão unificada e sincronizada dos ativos emitidos e consumidos.
- **Infiltração de Fraudes Documentais:** Sistemas legados validam o selo de forma isolada, mas não blindam o conteúdo do ato notarial. A desconexão entre os sistemas cartorários e os de automação judicial processual (como o SAJ ou eProc) facilita a introdução de procurações falsas e certidões adulteradas em processos judiciais. Essa prática gera insegurança jurídica e expressivos prejuízos financeiros aos cidadãos e ao erário público.
- **O Paradoxo da Transparência vs. Privacidade:** Para auditar com rigor, o analista necessita confrontar os dados que geraram o ato. Todavia, armazenar dados sensíveis abertamente para fins de auditoria viola as premissas de minimização e proteção integral estabelecidas pela Lei Geral de Proteção de Dados (LGPD).

1.2 A Proposta da Plataforma EX-JUD

Para superar as debilidades dos sistemas monolíticos, desenvolvemos a plataforma **EX-JUD**. A ferramenta instrumentaliza as Corregedorias por meio de um ecossistema em consórcio blockchain baseado em *Distributed Ledger Technology* (DLT) [11], utilizando a rede permissionada *Hyperledger Fabric 3.0* [1].

A EX-JUD introduz a transição de um modelo de confiança baseado em uma autoridade central única para um modelo de confiança distribuída amparado por provas criptográficas imutáveis. Os contratos inteligentes (*smart contracts*) gerenciam o ciclo de vida dos selos notariais como tokens digitais, prevenindo o ataque de gasto duplo e assegurando a conformidade legal.

2 Arquitetura e Componentes da Ferramenta

A ferramenta EX-JUD afasta-se de soluções monolíticas ao implantar uma arquitetura de microsserviços altamente desacoplada, separando de forma estrita as responsabilidades do órgão regulador e das serventias delegadas.

Como ilustrado na Figura 1, o ecossistema é regido pelo Consórcio *Blockchain* sob a governança da Corregedoria-Geral de Justiça (TJMS). A topologia da rede acomoda os nós das serventias (Cartórios Extrajudiciais) e o nó ordenador central. Para garantir o isolamento

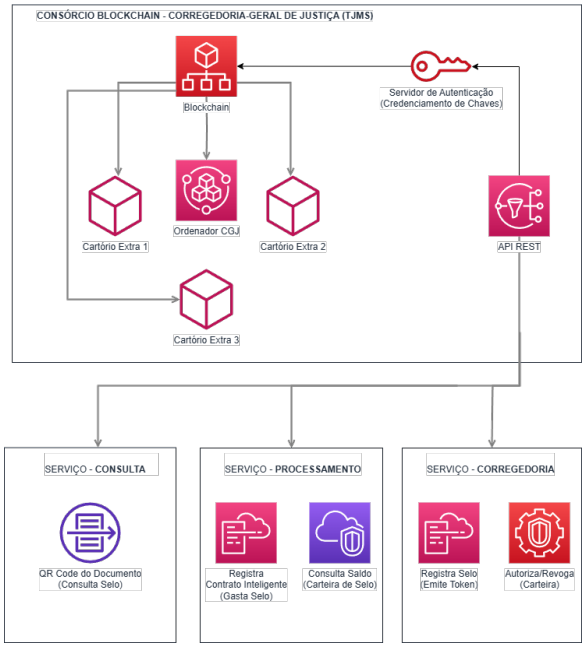

Figure 1: Visão dos microsserviços e do consórcio blockchain

e a segurança das operações, a comunicação com a rede distribuída ocorre intermediada por uma API REST e por um Servidor de Autenticação, responsável pelo credenciamento das chaves criptográficas.

A partir dessa camada de integração, a plataforma disponibiliza três verticais de serviços distintos: o **Serviço de Consulta**, voltado à verificação pública da autenticidade documental pelo cidadão via QR Code; o **Serviço de Processamento**, restrito aos cartórios para a execução de contratos inteligentes (gasto de selos) e consulta de saldo nas carteiras; e o **Serviço da Corregedoria**, de uso exclusivo do órgão regulador para a emissão de novos *tokens* (selos) e a autorização ou revogação de carteiras digitais.

Para suportar essa divisão lógica de serviços, a infraestrutura-base foi projetada em contêineres e é gerenciada via IaC (*Infrastructure as Code*). O aprofundamento técnico dessa rede, detalhando o fluxo transacional e a segregação de dados, é apresentado na Figura 2, que serve de base para a arquitetura em camadas descrita nas subseções a seguir.

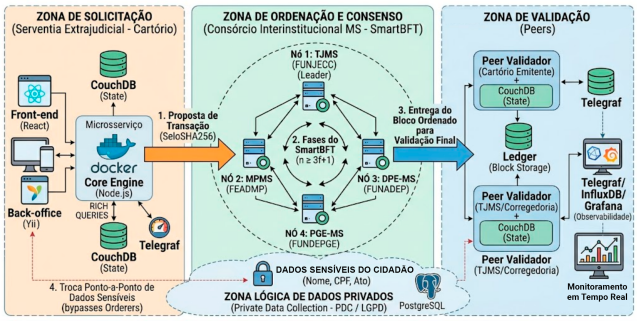

Figure 2: Arquitetura lógica e topologia da ferramenta

Como ilustrado na Figura 2, a arquitetura de rede é segmentada em três zonas principais: a **Zona de Solicitação** (onde operam as serventias), a **Zona de Ordenação e Consenso** (consórcio interinstitucional) e a **Zona de Validação** (*peers* e *ledger*). As aplicações *front-end* (Portal Cartório) e *back-office* (Portal Tribunal) se comunicam com a API Core Engine em Node.js. Essa API atua como o principal barramento do sistema, gerenciando as conexões gRPC com a rede e orquestrando o ciclo de vida das transações e validações de estado para manter o ecossistema escalável.

2.1 Camada Institucional (Domínio do Tribunal)

Gerido no âmbito da Corregedoria-Geral de Justiça (CGJ), o componente **Identity Manager & Back-office** foi desenvolvido em PHP 8.5 sobre o *framework* Yii 3.0. Para suportar altas cargas de requisições simultâneas e chamadas de API assíncronas, o ambiente utiliza o servidor de aplicação FrankenPHP 1.11 em Modo *Worker*.

Este módulo interage diretamente com um banco relacional PostgreSQL 18 para manter o controle de logs administrativos, dados de conciliação bancária dos emolumentos e o cadastro das serventias. No fluxo operacional, além de centralizar o **Serviço da Corregedoria** e encapsular as credenciais da Autoridade Certificadora (*Fabric CA*) para emissão de certificados X.509 e controle de acesso (*Access Control Lists*), o domínio do Tribunal abriga os nós endossantes do órgão regulador (*TJMS-ORG*).

Conforme a política de endosso da rede, o *peer* do TJMS atua ativamente na etapa de execução distribuída das transações. Ele recebe as propostas iniciadas pelas serventias, simula a execução do *chaincode* localmente e, caso as regras de negócio sejam respeitadas, aplica sua assinatura criptográfica (*Sign_TJMS*). Essa validação descentralizada garante que nenhum ato seja processado sem o aval criptográfico prévio da Corregedoria.

2.2 Camada Operacional (Domínio dos Cartórios)

Instalado localmente ou em nuvem privada em cada uma das serventias extrajudiciais cadastradas, o **Core Engine** atua como a API de orquestração do módulo cartório. Desenvolvido em Node.js 25 (LTS), este microsserviço utiliza chamadas assíncronas estritas para evitar o travamento da interface do escrevente no momento do atendimento aos jurisdiscionados.

O fluxo transacional na plataforma EX-JUD baseia-se em uma rigorosa política de endosso, definida pela regra lógica `AND(CARTÓRIO-MSP.peer, TJMS-MSP.peer)`. Quando a aplicação cliente solicita um ato, o *peer* do cartório atua como nó originador. Ele invoca o *chaincode* de transação — por exemplo, a função `ConsumirSelo` atrelada ao identificador do selo (`ID_Selo`) e à integridade do documento (`Hash_Doc`) —, executa a simulação local da **Proposta de Transação** e aplica sua própria assinatura digital (*Sign_Cartório*). Em seguida, a proposta é enviada aos *peers* endossantes do Tribunal para a coleta da segunda assinatura exigida pela política.

Conforme se observa na Figura 3, apenas após a obtenção do endosso completo (*Sign_Cartório* `AND` *Sign_TJMS*), comprovando a concordância bilateral, a transação consolidada é enviada à *Zona de Ordenação e Consenso*. Nesta zona central, o protocolo SmartBFT é executado pelo consórcio interinstitucional (TJMS, MPMS, DPE-MS

Figure 3: Processo de Assinatura das Transações

e PGE-MS) para determinar a ordem cronológica, amparado pela tolerância a falhas bizantinas [9, 10].

Para garantir a privacidade exigida pela Lei Geral de Proteção de Dados (LGPD), o fluxo utiliza uma abordagem híbrida. Os dados sensíveis do cidadão (como Nome, CPF e teor do ato) sofrem *bypass* na rede de ordenadores por meio de *Private Data Collections* (PDC). Essas informações trafegam via troca ponto a ponto criptografada exclusivamente entre os *peers* envolvidos, sendo armazenadas em instâncias apartadas na base do validador.

Por fim, o banco de dados que apoia localmente o nó do cartório é o CouchDB 3.5.1, operando como o *State Database* (*World State*) do Fabric. O CouchDB permite a execução de *Rich Queries* estruturadas em formato JSON, garantindo que o cartório consulte seu estoque de selos, execute baixas e visualize o histórico local instantaneamente em seu domínio de **Processamento e Consulta**, eliminando qualquer dependência dos nós de ordenação central para operações.

2.3 Smart Contracts e o Modelo CCaaS

O núcleo lógico de validação da ferramenta reside no contrato inteligente (*smart contract*) [13], implementado na linguagem Go (Golang 1.26) por meio da *Fabric Contract API*. Rompendo com o padrão de versões anteriores do Fabric, que exigiam o acoplamento do Docker Daemon ao nó *peer* para compilação local (Docker-in-Docker), a EX-JUD migrou para a arquitetura **Chaincode-as-a-Service (CCaaS)** utilizando *External Builders*, alinhando-se às diretrizes evolutivas do Fabric-X [8].

Sob o modelo CCaaS, o contrato roda como um microsserviço isolado em seu próprio contêiner Golang. A comunicação entre o *peer* e o contrato inteligente ocorre via canal gRPC bidirecional seguro, mapeado através de identificadores únicos denominados `CC_PACKAGE_ID`. O ciclo de vida do "Selo-Token" é regido por funções criptográficas expostas pelo contrato, destacando-se:

- `MintSelo()`: Executado apenas pela identidade do Tribunal, emite o lote de selos atestando o lastro financeiro.
- `ConsumirSelo()`: Vincula o identificador alfanumérico do selo ao *hash* SHA-256 do documento gerado no balcão, alterando o estado do ativo de forma definitiva no livro-razão e impedindo o ataque de gasto duplo.
- `RevogarSelo()`: Função de controle exclusivo da Corregedoria para invalidar atos específicos no *World State* por razões judiciais ou administrativas.

Para garantir a conformidade com a LGPD, a modelagem de dados no *smart contract* segrega rigorosamente as informações públicas das informações sensíveis do cidadão. A Listagem 1 demonstra as estruturas de dados (*structs*) utilizadas no código-fonte, evidenciando como os dados sensíveis são isolados na estrutura `SeloDataPriv`, a qual é persistida exclusivamente nas coleções privadas (*Private Data Collections* - PDC).

```
// Selo descreve os metadados do ativo digital no World
    State

type Selo struct {
    IDSelo      string  `json:"id_selo"`
    CartorioID  string  `json:"cartorio_id"`
    HashDoc     string  `json:"hash_doc"`
    Status      string  `json:"status"` // "EMITIDO", "
        CONSUMIDO" ou "REVOGADO"
    Timestamp   string  `json:"timestamp"`
    Valor       float64 `json:"valor"`
}

// SeloDataPriv representa os dados sens veis salvos
    APENAS na PDC

type SeloDataPriv struct {
    IDSelo     string `json:"id_selo"`
    CidadaoID  string `json:"cidadaoId"`
    Nome       string `json:"nome"`
    Detalhes   string `json:"detalhes"`
}
```

Listing 1: Contrato Inteligente em Go para segregação de informações sensíveis (LGPD)

2.4 Privacy by Design e Consenso SmartBFT

Para garantir a conformidade com a LGPD, os dados sensíveis sigilosos dos cidadãos (como nomes e CPFs) nunca são expostos na estrutura pública replicada da blockchain. A ferramenta implementa as **Private Data Collections (PDC)** em estrita observância ao princípio de *Privacy by Design* [5]. Os dados textuais confidenciais são transmitidos de forma criptografada via dados transientes (*Transient Data*) e armazenados em tabelas laterais privadas (*SideDBs*) protegidas pelo CouchDB dos nós envolvidos.

A segurança e a imutabilidade contra ataques cibernéticos ou falhas de nós validadores maliciosos baseiam-se na adoção do protocolo de consenso **SmartBFT**. Diferente do algoritmo Raft, que tolera apenas falhas de queda (*Crash Fault Tolerant*), o SmartBFT implementa uma variação moderna de tolerância a falhas bizantinas (PBFT) [4]. A rede sustenta a integridade determinística das transações desde que o número total de nós validadores obedeça à equação clássica $n \geq 3f + 1$, onde f representa o limite de nós corrompidos [10].

3 Principais Funcionalidades e Usuários

A ferramenta EX-JUD foi concebida sob uma arquitetura de governança em camadas, alinhando o fluxo de trabalho extrajudicial

aos diferentes níveis de responsabilidade do ecossistema jurídico. Como ilustrado no Diagrama de Casos de Uso (Figura 4), o sistema interage com quatro perfis de atores bem delimitados:

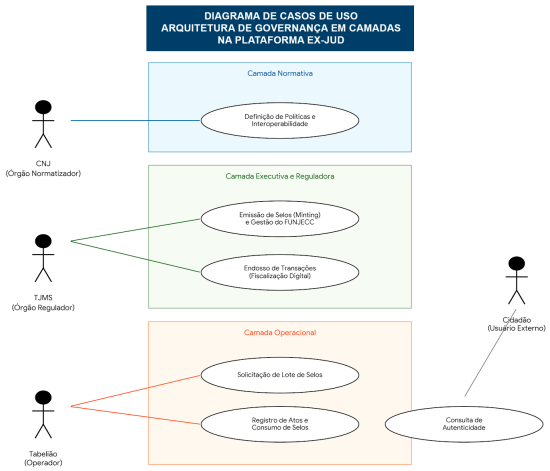

Figure 4: Diagrama de Casos de Uso organizado pelas camadas de governança da plataforma EX-JUD

(1) **CNJ (Órgão Normatizador - Camada Normativa):** Atua no nível mais alto da governança do sistema, sendo responsável pela definição das políticas gerais e por garantir as diretrizes de interoperabilidade nacional entre as soluções do Judiciário.
(2) **TJMS / Corregedoria (Órgão Regulador - Camada Executiva e Reguladora):** Exerce a administração correicional ativa. Suas funcionalidades centrais incluem a emissão de selos digitais (*minting*), a gestão de repasses financeiros para fundos institucionais e o endosso de transações para fiscalização digital contínua. Além disso, detém ferramentas de segurança para revogação criptográfica em lote; caso um cartório relate invasão de credenciais ou extravio de mídias, o Tribunal dispara uma transação que invalida os *tokens* daquela serventia de forma imediata e imutável.
(3) **Tabelião / Escrevente Notarial (Operador - Camada Operacional):** Atua diretamente na ponta de atendimento. Esse perfil possui acesso a um painel de controle para solicitação de novos lotes de selos, gerenciamento de estoque com alertas de nível mínimo de ativos e um formulário automatizado para o registro de atos e consumo vinculativo dos selos.
(4) **Cidadão (Usuário Externo):** Representa o consumidor final do serviço notarial. Utiliza um portal de consulta externa para realizar a validação de autenticidade dos documentos recebidos, garantindo segurança jurídica sem a necessidade de possuir chaves privadas ou cadastros prévios no ecossistema.

4 Interfaces da Ferramenta

A plataforma EX-JUD disponibiliza interfaces dedicadas para os três níveis de usuários do ecossistema, garantindo usabilidade e transparência operacional. Para atender aos rigorosos requisitos de governança digital corporativa do Judiciário, o sistema adota um design claro, responsivo e focado na facilidade das operações.

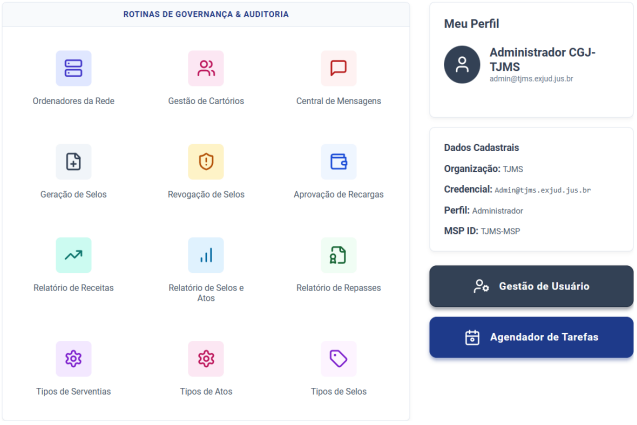

Figure 5: Dashboard Administrativo da Corregedoria-Geral de Justiça (TJMS)

A Figura 5 ilustra a visão do **Administrador Correicional**. Neste painel central, o servidor autorizado do Tribunal possui acesso direto à identificação de suas credenciais criptográficas (*Chave Pública*) e a um menu focado em **Rotinas Administrativas**. Através desta interface, a CGJ executa funções críticas para a saúde da rede, acessando módulos como o **Credenciamento de Cartórios**, **Estoque de Selos**, **Controle de Nós Validadores** e, em cenários de auditoria, a **Revogação de Selos**.

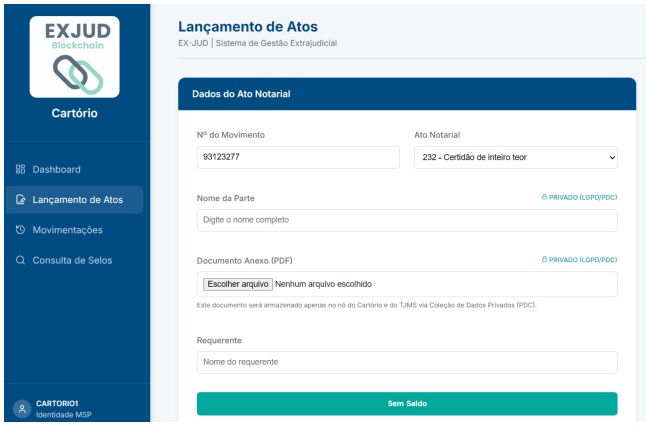

Figure 6: Interface operacional do Cartório evidenciando os campos protegidos pela LGPD via Private Data Collection

O fluxo transacional diário ocorre no ambiente delegado às serventias (Figura 6). A interface do **Cartório** apresenta um menu lateral com as rotinas de balcão. O formulário de **Lançamento de Atos** é projetado para guiar o escrevente notarial. Destaca-se visualmente nesta tela a marcação indicativa `PRIVADO (LGPD/PDC)` anexada aos campos sensíveis, como o **Nome da Parte** e o **Documento Anexo (PDF)**. Essa sinalização confirma ao operador a

arquitetura de *bypass* de dados sensíveis: o arquivo selecionado não é transmitido ao livro-razão público, sendo armazenado de forma cifrada apenas nos nós do Cartório e do TJMS.

O fluxo de processamento neste balcão segue os seguintes passos:

(1) O operador seleciona o tipo de ato notarial (ex: *232 - Certidão de inteiro teor*), preenche o formulário com dados sensíveis e anexa o arquivo PDF finalizado.
(2) A aplicação aciona localmente a rotina de cômputo da assinatura digital SHA-256 do arquivo no navegador do cliente (*Web Crypto API*).
(3) O sistema valida instantaneamente o saldo da carteira do cartório (sinalizado pelos botões de ação na parte inferior da tela).
(4) Ao confirmar o lançamento, o *frontend* encapsula os dados sensíveis na estrutura volátil de *Transient Data* e envia o *hash* via requisição REST para o *Core Engine*, que orquestra a coleta de endossos e a submissão ao *cluster* SmartBFT.

Figure 7: Portal de Consulta Pública de Selos Digitais

Por fim, a validação da autenticidade pelo **Cidadão** (ou usuário externo) é realizada por meio do portal de consulta (Figura 7). Projetada com uma abordagem *mobile-first*, a interface permite que qualquer indivíduo informe o identificador único do selo alfanumérico impresso no documento físico ou utilize a câmera do dispositivo móvel através do **Leitor QR**. O portal confronta os dados diretamente com o livro-razão distribuído, atestando a integridade criptográfica do ato de maneira instantânea e dispensando cadastros.

5 Avaliação de Carga e Resultados

Para validar a resiliência e a estabilidade da arquitetura de *software* proposta, o ambiente composto por quatro nós de ordenação SmartBFT e cinco nós *peers* validadores foi submetido a testes rigorosos de estresse. O foco foi mensurar o impacto da orquestração CCaaS (*Chaincode as a Service*) e as correções de compilação aplicadas no ecossistema Go 1.26.

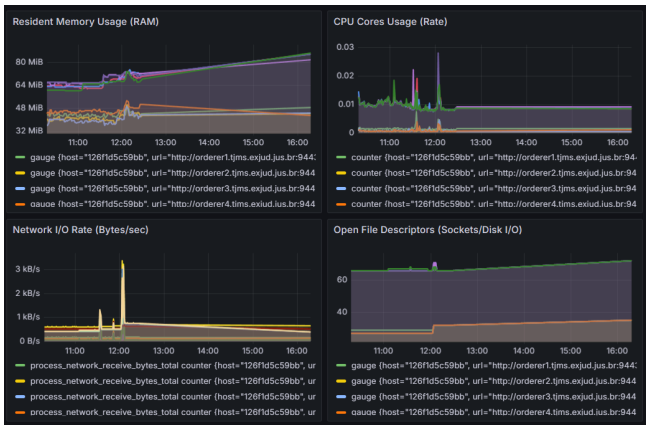

Figure 8: Métricas de consumo de recursos (RAM, CPU, Rede e I/O) dos nós ordenadores durante os testes de estresse

Como ilustrado pelos painéis de telemetria na Figura 8, a infraestrutura demonstrou alta eficiência operacional. Sob uma carga controlada e contínua de 50 transações por segundo (TPS), simulando a operação nominal simultânea de dezenas de serventias de grande porte, a latência média situou-se em 1,2 segundos. Os gráficos refletem o momento exato das janelas de estresse agressivo (picos observados próximos às 12:00), com rajadas atingindo 250 TPS.

Durante esses picos extremos, a plataforma manteve a estabilidade sem descartar requisições, apresentando um consumo de hardware notavelmente otimizado, com o uso de memória RAM estabilizado abaixo de 85 MiB e frações residuais de processamento (CPU). Devido às fases de votação e assinaturas múltiplas intrínsecas ao quórum PBFT do algoritmo SmartBFT, a latência média na janela de estresse elevou-se para 4,8 segundos. Ainda assim, as filas de processamento e de descritores de arquivos (*sockets*/I/O) mantiveram-se consistentes.

Esses resultados confirmam que a topologia adotada e o uso de microsserviços em Node.js sobre gRPC atendem rigorosamente aos requisitos de concorrência, viabilizando a substituição dos sistemas monolíticos legados sem introduzir atrasos perceptíveis no atendimento de balcão ao cidadão.

6 Conclusão e Trabalhos Futuros

A plataforma EX-JUD consolida-se como uma solução madura e inovadora voltada à Governança Digital do Poder Judiciário. Ao aliar microsserviços desacoplados, contratos inteligentes em modelo CCaaS e privacidade nativa via Private Data Collections, a ferramenta resolve de forma concomitante o desafio da auditabilidade correicional dos fundos de repasse institucionais e a conformidade estrita com a LGPD. Como perspectivas para trabalhos futuros, a equipe planeja expandir as rotas de comunicação assíncronas da API Core Engine para acoplá-las diretamente às rotinas de distribuição processual do sistema de automação de processos (como o eProc), viabilizando a validação automatizada de procurações e escrituras.

References

[1] Elli Androulaki, Artem Barger, Vita Bortnikov, Christian Cachin, Konstantinos Christidis, Angelo De Caro, David Enyeart, Christopher Ferris, Gennady Laventman, Yacov Manevich, Srinivasan Muralidharan, Chet Murthy, Binh Nguyen, Manish Sethi, Gari Singh, Keith Smith, Alessandro Sorniotti, Chrysoula Stathakopoulou, Marko Tamayo, Gari Vadamoole, Jason Weaver, and Wo Yellick. 2018. Hyperledger Fabric: A Distributed Operating System for Permissioned Blockchains. In *Proceedings of the Thirteenth EuroSys Conference* (Porto, Portugal) *(EuroSys '18)*. Association for Computing Machinery, New York, NY, USA, Article 30, 15 pages. doi:10.1145/3190508.3190538

[2] Marcelo Benacchio. 2022. *Direito Notarial e Registral Contemporâneo: Governança, Compliance e Fiscalização* (2 ed.). Revista dos Tribunais, São Paulo.

[3] Brasil. 1994. Lei nº 8.935, de 18 de novembro de 1994. Regulamenta o art. 236 da Constituição Federal, dispondo sobre serviços notariais e de registro. (Lei dos Cartórios). http://www.planalto.gov.br/ccivil_03/leis/l8935.htm

[4] Miguel Castro and Barbara Liskov. 1999. Practical Byzantine Fault Tolerance. In *Proceedings of the Third Symposium on Operating Systems Design and Implementation (OSDI)*. USENIX Association, New Orleans, USA, 173–186.

[5] Ann Cavoukian. 2009. *Privacy by Design: The 7 Foundational Principles*. Technical Report. Information and Privacy Commissioner of Ontario, Canada. https://student.cs.uwaterloo.ca/~cs492/papers/7foundationalprinciples_longer.pdf

[6] CNJ. 2024. *Programa Justiça 4.0: conheça a iniciativa que vem mudando o Poder Judiciário brasileiro pela transformação digital e ampliação do acesso à justiça cidadã*. Technical Report. CNJ, Brasília, Brasil. Disponível em https://bibliotecadigital.cnj.jus.br/handle/123456789/869.

[7] Conselho Nacional de Justiça. 2024. *Plataforma Digital do Poder Judiciário Brasileiro (PDPJ-Br)*. Technical Report. Brasília, Brasil. Disponível em https://www.cnj.jus.br/tecnologia-da-informacao-e-comunicacao/plataforma-digital-do-poder-judiciario-brasileiro-pdpj-br/.

[8] Hyperledger Foundation. 2025. Hyperledger Fabric-x: The Future of Hyperledger Fabric Architecture. https://github.com/hyperledger/fabric-x Acesso em: 12 dez. 2025.

[9] S. Gao, T. Yu, J. Zhu, and W. Cai. 2019. T-PBFT: An EigenTrust-based practical Byzantine fault tolerance consensus algorithm. *China Communications* 16, 12 (2019), 111–123. doi:10.23919/JCC.2019.12.008

[10] Leslie Lamport, Robert Shostak, and Marshall Pease. 1982. The Byzantine Generals Problem. *ACM Transactions on Programming Languages and Systems (TOPLAS)* 4, 3 (1982), 382–401. doi:10.1145/357172.357176

[11] Jennifer Li and Mohamad Kassem. 2021. Applications of distributed ledger technology (DLT) and Blockchain-enabled smart contracts in construction. *Automation in construction* 132 (2021), 103955.

[12] STI/TJMS. 2023. *Manual do Sistema de Gerenciamento das Serventias Extrajudiciais (SIG-EX)*. Secretaria de Tecnologia da Informação, Campo Grande, MS. https://sig-ex.tjms.jus.br/publico/Manual_SIG-EX.pdf

[13] Nick Szabo. 1996. Smart Contracts: Building Blocks for Digital Markets. (1996). Disponível em http://www.fon.hum.uva.nl/rob/Courses/InformationInSpeech/CDROM/Literature/LOTwinterschool2006/szabo.best.vwh.net/smart_contracts_2.html.