UNIVERSIDADE FEDERAL RURAL DO SEMIÁRIDO - UFERSA
CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO
Trabalho de Conclusão de Curso (2023).

EVOLUÇÃO DA ARQUITETURA DE PROCESSAMENTO ASSÍNCRONO PARA REGISTRO DE DOCUMENTOS DIGITAIS ACADÊMICOS EM BLOCKCHAIN

Mikéias G. M. Azevedo [1], Paulo G. G. Queiroz [2], Daniel F. L. de Souza [3]

Resumo: *A tecnologia blockchain tem sido amplamente utilizada em setores financeiros e não financeiros devido à sua capacidade de fornecer um registro transacional distribuído e transparente. No entanto, a sua adoção apresenta desafios decorrentes das suas características como sistema distribuído. Diante disso, o objetivo deste trabalho é aprimorar a arquitetura de uma aplicação responsável pelo registro de diplomas digitais e documentação acadêmica, tornando-a mais escalável, eficiente e resiliente. A arquitetura proposta possui módulos segregados para registro e revogação e utiliza-se do módulo Rap Broker como mecanismo de comunicação com os DLTs (Distributed Ledger Tecnologies) e de um message broker para comunicação assíncrona. Além disso, implementa o padrão de tarefas agendadas para otimização de processos internos e aplica os padrões de retentativas e enfileiramento de mensagens para resiliência e comunicação assíncrona. Com a sua implantação, foi possível obter uma aplicação capaz de atender a 78 IES (Instituições de Ensino Superior) e registrar picos de 10.000 documentos digitais diários.*
Palavras-chave: blockchain, documentação acadêmica digital, diploma digital, padrões de design.

1 INTRODUÇÃO

Os fundamentos da tecnologia *blockchain* e as estratégias para viabilizar as operações transacionais em cryptomoedas foram popularizados por Satoshi Nakamoto no artigo “Bitcoin: A Peer-to-Per Eletronic Cash System” (NAKAMOTO, 2008). Embora a verdadeira identidade de Nakamoto não tenha sido revelada, tal tecnologia tem ganhado espaço no mercado e vem sendo aplicada tanto em domínios financeiros como não financeiros.

Os cenários de utilização no setor de finanças podem ser exemplificados com os próprios criptoativos (*Bitcoin*, *Litecoin*, *Ether*) e com o registro de propriedades e verificação de histórico por empresas de seguros. Já nas aplicações não financeiras, há o uso dessa tecnologia para armazenamento descentralizado de documentos, persistência de registros de comunicação entre dispositivos inteligentes em IoT (*Internet of Things*) e verificação de autenticidade de produtos entre fornecedores e clientes. Essas aplicações estão listadas em (NOFER *et al.*, 2017) e (ZILE; STRAZDINA, 2018).

O conceito de *blockchain* está intrinsecamente ligado ao conceito de DLT (*Distributed Ledger Technologies*) ou livros razão distribuídos, uma vez que ela pode ser vista como uma materialização da ideia da DLT por méio do encadeamento de blocos de dados cronologicamente ordenados (ZILE; STRAZDINA, 2018). Sendo assim, a tecnologia possui, por natureza, mecanismos que garantem integridade e redundância dos dados. Na prática, os blocos armazenam dados com o seu horário de processamento através da criação de transações. Cada bloco recebe um valor de *hash*, que é calculado usando o *hash* anterior (PIERRO, 2017), e que é propagado quando é gerado. Por isso, o processo de alteração de blocos em *blockchain* é extremamente custoso computacionalmente.

A *blockchain* utiliza algoritmos de consenso para garantir que os blocos estejam distribuídos pelos nós em rede e que possam ser verificados utilizando os *hashes*

[1] Discente do Bacharelado em Ciência da Computação da UFERSA - Campus Mossoró
[2] Professor Adjunto IV do Departamento de Computação da UFERSA - Campus Mossoró
[3] Professor Adjunto IV do Departamento de Ciências Exatas da UFPB - Campus Rio Tinto

anteriormente citados. Isso faz com que múltiplas entidades tenham a responsabilidade de manter essas transações, evitando centralizações (NOFER *et al.*, 2017). Dessa forma, o conjunto de recursos da *blockchain* a tornam extremamente eficiente para impedir fraudes em diversos contextos de registro de dados e transações.

Em 2018, o grupo de pesquisa LAVID, da UFPB, através do financiamento da RNP, propôs o uso combinado da certificação digital, preservação digital e *blockchain* para registro e autenticação de documentos digitais acadêmicos, aproveitando-se das características transacionais descentralizadas para combater fraudes na emissão desses documentos no Brasil (COSTA *et al.*, 2018).

Como uma frente de trabalho paralela também em 2018, o mesmo grupo de pesquisa LAVID desenvolveu uma arquitetura baseada em *brokers* para registro em múltiplos DLTs (PIRES *et al.*, 2018). Na arquitetura proposta, o *broker* é um componente arquitetural que opera abstraindo os detalhes de implementação da comunicação com cada *blockchain* (dado que cada uma delas possui especificidades e aplicações diferentes) e que também faz o monitoramento do estado da transação, isto é, se ela foi confirmada ou não.

Com base nessa arquitetura, foi criada uma prova de conceito denominada *RAP Registry*, um serviço responsável pela comunicação com os DLTs através dos *brokers* mencionados anteriormente. Esse serviço disponibiliza uma API para recuperação de transações e registros, servindo como um *gateway* para usuários externos. Entretanto, como uma versão inicial, ele apresenta algumas fragilidades, como baixa tolerância a falhas no processamento de registros de documentos e alto custo operacional.

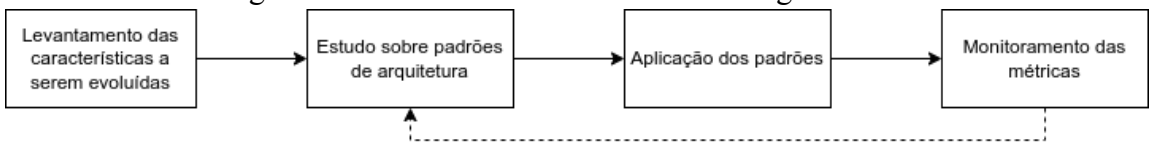
Diante disso, este trabalho tem como objetivo aprimorar o *RAP Registry*, buscando criar e aplicar uma arquitetura de software eficiente para lidar com o processamento assíncrono de registros em *blockchain* de maneira escalável, para assegurar a integridade do processo e reduzir os custos transacionais.

O restante deste artigo está organizado da seguinte maneira: na seção 2, descreve-se a metodologia utilizada para a construção da arquitetura proposta. Em seguida, na seção 3, apresentam-se os trabalhos relacionados, indicando suas semelhanças e limitações em relação ao presente trabalho. Na seção 4, expõe-se a proposta de arquitetura para a evolução do sistema *RAP Registry* e os mecanismos empregados para alcançar os objetivos de confiabilidade e escalabilidade. Na seção 5, são apresentados os resultados obtidos com a utilização do sistema na prática, processando o registro de documentos digitais acadêmicos em *blockchain*. Por fim, na seção 6, são expostas as considerações finais deste trabalho, destacando-se suas contribuições, dificuldades encontradas e sugestões para trabalhos futuros.

2 METODOLOGIA

Dada a complexidade do trabalho, a execução foi dividida em 4 atividades que serão enunciadas a seguir e que podem ser vistas na figura 1. As atividades são representadas pelos retângulos, e o fluxo de execução é direcionado pelas setas. É importante destacar que a seta pontilhada ligando as atividades 2 e 4 indica que esse fluxo pode ser repetido, dependendo dos resultados obtidos em relação ao monitoramento das métricas.

Figura 1. Fluxo das atividades da metodologia adotada.



Fonte: autoria própria (2023).

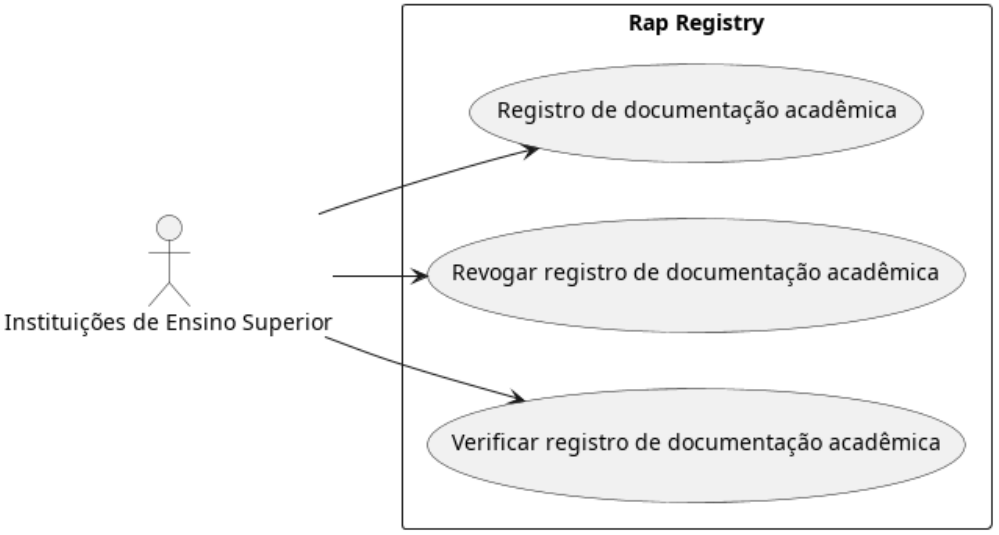
2.1 Levantamento das características de evolução

Inicialmente, foi feito um mapeamento dos casos de uso, das características e dos requisitos não funcionais que demandam evolução na arquitetura atual do sistema.

Os casos de uso foram identificados a partir da estrutura preliminar do sistema e da documentação disponível nos repositórios de código do projeto. Adicionalmente, para cada caso de uso, observou-se as oportunidades de evolução nos componentes e módulos utilizados por meio das métricas de falhas e análise empírica do funcionamento da aplicação.

As subseções seguintes trarão de forma mais detalhada os casos de uso e oportunidades de melhoria percebidas.

Figura 2. Diagrama de casos de uso do *Rap Registry*.



Fonte: autoria própria (2023).

2.1.1 Registro de documentação acadêmica

Fluxo responsável por registrar os metadados da documentação acadêmica em *blockchain*, demandando uma carga de processamento de dezenas de milhares de documentos por dia com resiliência durante esse processo. É importante ressaltar que, embora não haja urgência em relação ao intervalo de tempo em que o registro é feito, é fundamental garantir que o processo seja executado de forma confiável e sem interrupções.

2.1.2 Revogar registro de documentação acadêmica

Este caso de uso tem como objetivo revogar os metadados de uma documentação acadêmica registrada em *blockchain*. O processamento da revogação deve ser resiliente e suportar milhares de revogações por dia. Assim como no registro, não há urgência em relação ao período de tempo em que a revogação é realizada, mas é essencial garantir a sua execução.

2.1.3 Verificar registro de documentação acadêmica

Essa funcionalidade descreve o cenário para obter os metadados do registro e verificar se ele é válido. Esse caso de uso é menos crítico e não possui requisitos a serem evoluídos.

2.2 Estudo sobre padrões de arquitetura

Com as características mapeadas, a próxima etapa consistiu em identificar padrões de processamento assíncrono que são utilizados comercialmente em aplicações orientadas à *blockchain* (ou não) e que auxiliassem a alcançar os objetivos de cada processo.

Os padrões foram pesquisados em livros que compõem o estado da arte dos sistemas distribuídos e padrões de mercado para desenvolvimento de sistemas, como Tanenbaum e Steen (2023), Pinedo (2016) e Coulouris *et al.* (2011), que serão retomados na seção 4, conforme forem aplicados nos componentes da arquitetura.

2.3 Aplicação dos padrões

A partir dos padrões estudados no passo anterior, foi feita a adequação da arquitetura do *RAP Registry*. A decisão de aplicar ou não um padrão foi tomada com base na complexidade inerente à sua implantação na infraestrutura, isto é, se a arquitetura já possuía suporte técnico para utilizá-lo. Dessa forma, a implementação seria mais rápida e menos custosa.

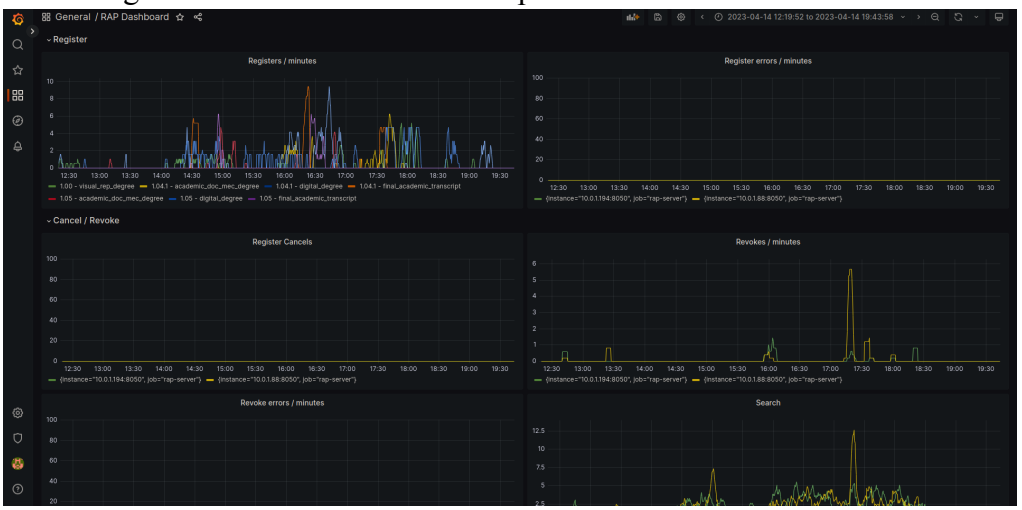
Em relação à decisão de quais componentes de software deveriam ser refatorados, levou-se em conta os serviços e fluxos de cada caso de uso, se eles poderiam ser convertidos em processos assíncronos, além das suas taxas de erros.

2.4 Monitoramento das métricas

Após a implementação e lançamento das alterações em ambientes de implantação, realizou-se o monitoramento das métricas e dos impactos das atualizações no negócio.

O projeto contou com o uso combinado das ferramentas Prometheus (PROMETHEUS, 2023) e Grafana (GRAFANA LABS, 2019) como infraestrutura de monitoramento. O Prometheus é um sistema que exporta diferentes tipos de métricas de aplicação. Enquanto isso, o Grafana possibilita a criação de painéis de monitoramento com gráficos que operam em tempo real, além de permitir a geração de alertas que são enviados, conforme critérios definidos são alcançados (o que é muito útil para o monitoramento de falhas). Ademais, o *Rap Registry* possui portais internos para coleta e apresentação de estatísticas específicas para o domínio, como número de registros por IES (Instituições de Ensino Superior), por exemplo.

Figura 3. Dashboard no Grafana para os fluxos do RAP.



Fonte: autoria própria (2023).

3 TRABALHOS RELACIONADOS

Os principais trabalhos que se relacionam com a presente publicação foram citados de forma inicial na seção 1, mas serão retomados nesta seção.

Em Costa *et al.* (2018), os autores focam na proposta de utilização das tecnologias de *blockchain* e certificação digital para inibir as fraudes no cenário da emissão de documentação acadêmica. Isso é feito trazendo luz ao impacto da adoção da documentação digital (no contexto do trabalho, o diploma digital e documentação acadêmica digital), expondo também os seus desafios em relação a todo o processo de geração, registro, autenticação e preservação. Além disso, também é apresentado um protótipo arquitetural que deu origem ao serviço que está sendo abordado no presente trabalho.

Já em Pires *et al.* (2018), o foco é direcionado à criação de uma estratégia de registro de transações em múltiplos livro-razão distribuídos (ou DLTs) baseada em *brokers*. A principal motivação da arquitetura proposta é lidar com as singularidades de cada *blockchain* de forma transparente, seja nas informações necessárias para realizar uma transação, no tamanho de armazenamento ou suporte aos contratos inteligentes. Por isso, a solução proposta consiste na criação de um ponto de acesso unificado denominado *Broker*, que provê *gateways* para cada tipo de DLT, abstraindo toda a complexidade do registro, da recuperação de dados da transação e da própria DLT. Internamente, cada *gateway* possui múltiplos provedores de comunicação com o DLT ao qual está associado, aumentando a resiliência na sua operação. Dito isso, a arquitetura apresentada aqui foi utilizada como ponto de comunicação com os DLTs no presente trabalho.

Os estudos mencionados acima contribuem para o desenvolvimento deste artigo ao fornecerem um arcabouço teórico e prático envolvendo microsserviços, *blockchain* e a aplicação aqui discutida. O trabalho de Costa *et al.* (2018) fornece a arquitetura inicial na qual o *Rap Registry* está inserido, focando em um contexto acerca da motivação da sua criação e em uma visão mais abstrata da aplicação, sem abordar os desafios técnicos referentes à resiliência na sua implementação. Por fim, Pires *et al.* (2018) apresenta o desenvolvimento de uma poderosa abstração para operar sobre diferentes DLTs de forma transparente. Embora forneça uma visão mais pragmática, também existem lacunas em relação às técnicas utilizadas para garantir a confiabilidade em seu funcionamento

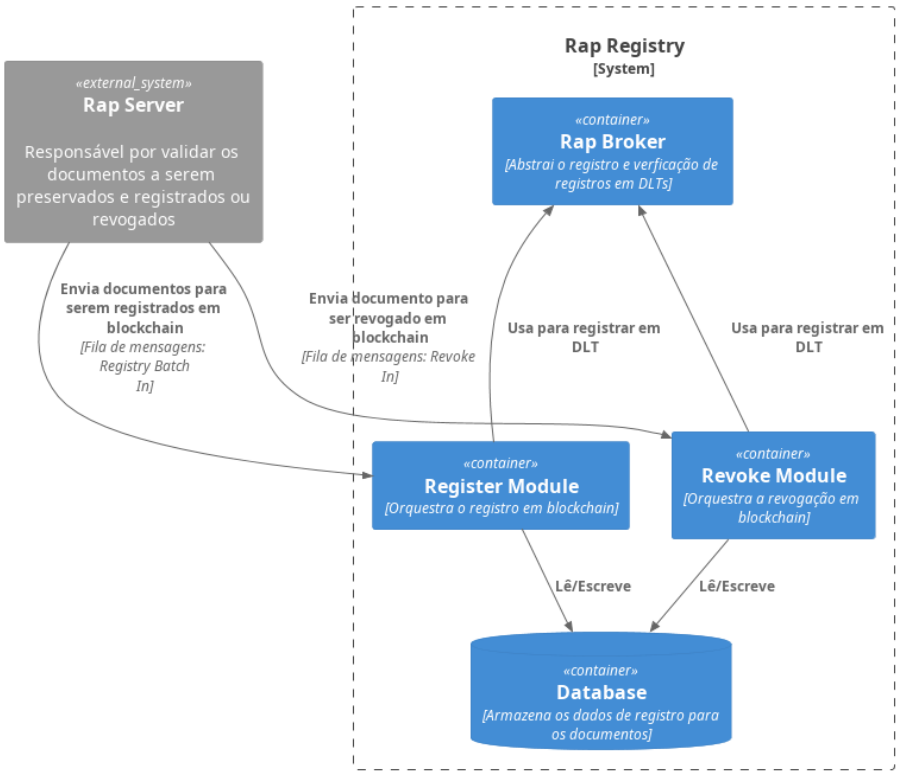
4 ARQUITETURA PROPOSTA PARA O RAP REGISTRY

Na subseção 4.1 e subseção 4.2, apresenta-se, respectivamente, os módulos de registro e revogação do *Rap Registry* vistos na figura 4. Em cada subseção, é posta a visão da arquitetura legado, para fins de comparação e, em seguida, é exposta a proposta de arquitetura desenvolvida e as decisões de design envolvidas na sua concepção.

Os diagramas apresentados foram modelados seguindo o padrão C4 (BROWN, 2006) para modelos de arquitetura. Esse modelo se baseia em diagramas de arquitetura com diferentes níveis de abstração, permitindo uma visão geral das responsabilidades do sistema até diagramas a nível de código.

Por convenção, como estereótipos adotados nesse trabalho, os retângulos brancos indicam componentes que atuam como *listeners* ou *workers* das mensagens recebidas das filas de mensagens aos quais estão associados. Já as elipses de cor cinza são tarefas agendadas que executam em intervalos configuráveis.

Figura 4. Diagrama para os módulos internos do *RAP Registry* no nível de container.



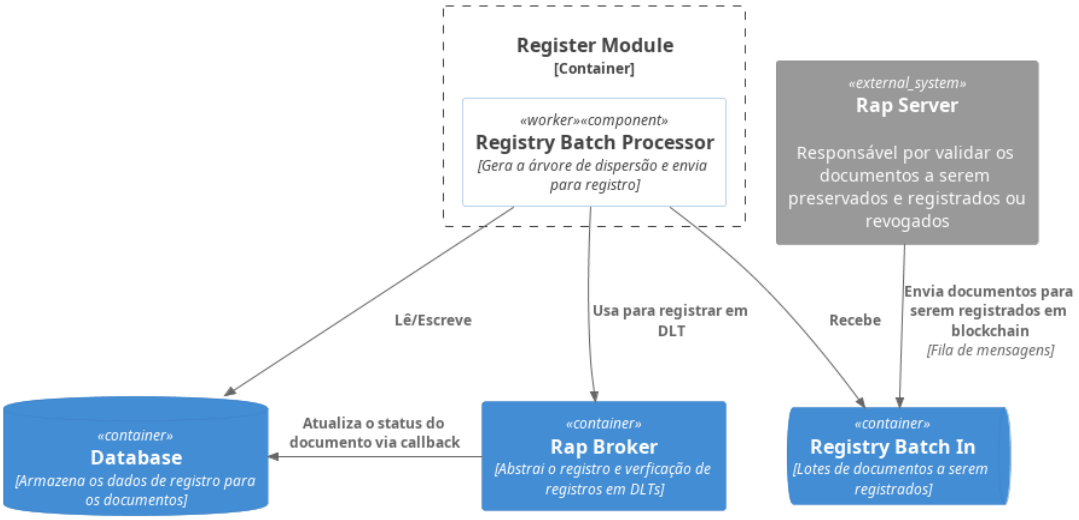
Fonte: autoria própria (2023).

4.1 Módulo de Registro

4.1.1 Arquitetura legado

Os componentes do módulo de registro da arquitetura legado, representados na figura 5, são detalhados a seguir.

Figura 5. Diagrama para a arquitetura legado do módulo de registro do *RAP Registry* no nível de componentes.



Fonte: autoria própria (2023).

4.1.1.1 Registry Batch Processor

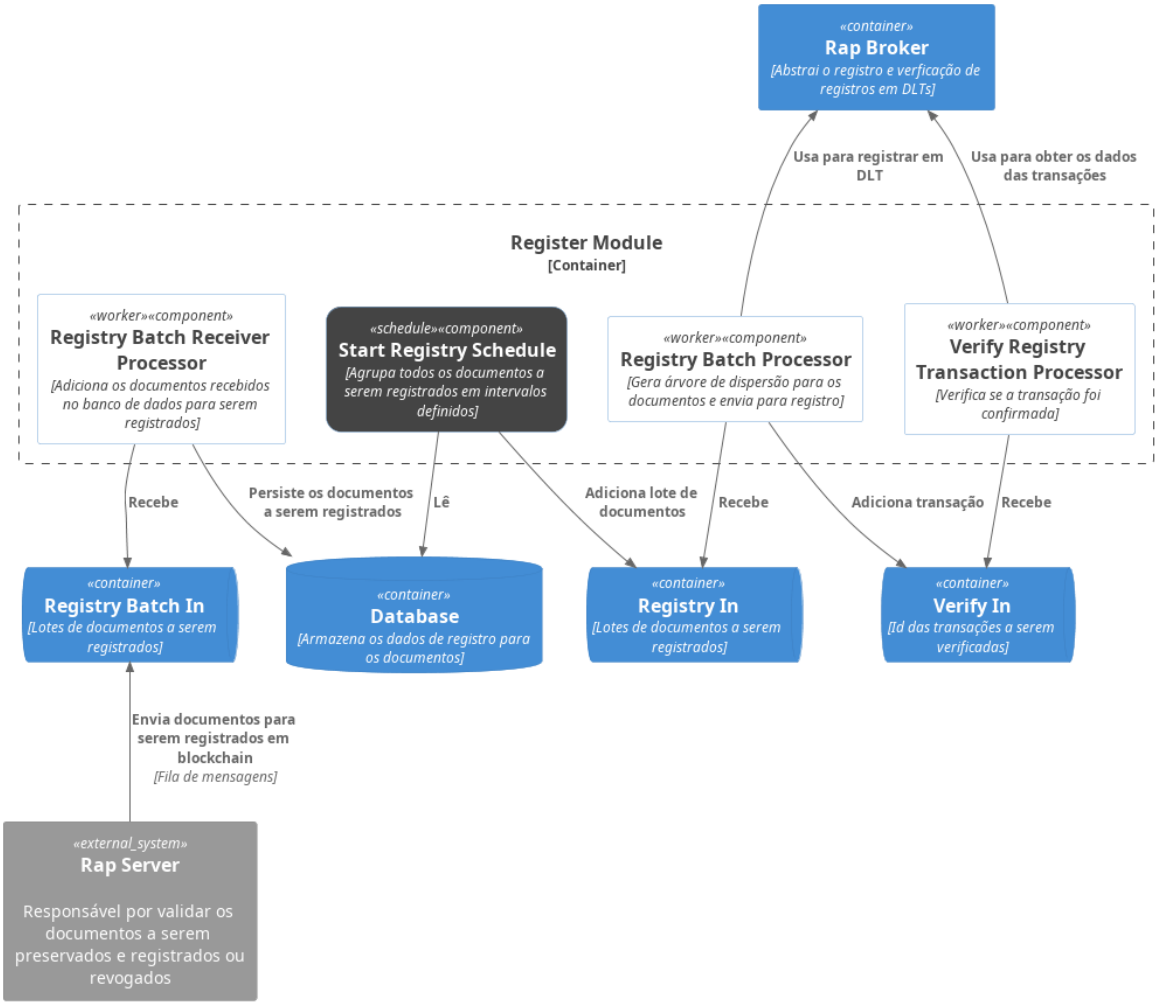
O *Registry Batch Processor* é um consumidor que recebe um lote de documentos a serem registrados da fila *Registry Batch In*. A cada lote de documentos é gerada a árvore de dispersão para eles. Em seguida, a árvore gerada é enviada para o registro por meio do *Rap Broker* com um *callback,* que é executado caso o registro seja confirmado, atualizando o estado do documento no banco de dados.

Observando as métricas desse fluxo, o envio para registro tinha uma baixa taxa de falhas. Entretanto, o processo de verificação das confirmações do registro muitas vezes falhava, e os dados no banco de dados ficavam em estado de inconsistência. Além disso, para cada registro é cobrada uma taxa de processamento pela rede *blockchain*, o que fazia com que o registro conforme a demanda de pequenos lotes fosse ineficiente em relação a custos. Por isso, esse foi um dos componentes repensados na nova arquitetura.

4.1.2 Arquitetura proposta

Os componentes do módulo de registro representados na figura 6 são detalhados a seguir.

Figura 6. Diagrama para a arquitetura proposta para o módulo de registro do *RAP Registry* no nível de componentes.



Fonte: autoria própria (2023).

4.1.2.1 Registry Batch Receiver Processor

O *Registry Batch Receiver Processor* consome os documentos a serem registrados da fila *Registry Batch In,* cujo envio é feito pelo *Rap Server*, persistindo-os no banco de dados com um estado de "aguardando registro".

4.1.2.2 Start Registry Schedule

Novo processo que executa uma tarefa agendada de agrupamento dos documentos que estão prontos para serem registrados, persistidos pelo *Registry Batch Receiver Processor*, e os envia para a fila de mensagens *Registry In*. A ideia de agrupar os documentos surge para reduzir custos, uma vez que cada transação possui um valor a ser pago para que seja adicionada à *blockchain*.

A abordagem de tarefas agendadas é amplamente conhecida, desempenhando um papel fundamental em sistemas e processos de grandes organizações, como apresentado por Pinedo (2016). Por isso, ela foi adotada no contexto deste trabalho.

O grande desafio de implementação foi lidar com os aspectos inerentes à escalabilidade horizontal, uma vez que é possível existir múltiplas instâncias do *RAP Registry* em execução, e isso demanda algum tipo de sincronização para evitar estados de inconsistência entre as entidades no banco de dados e reprocessamento de registros. Para solucionar esse problema, foram utilizados os mecanismos de transação e bloqueio suportados pelo banco de dados relacional

4.1.2.3 Registry Batch Processor

Este componente atua como um consumidor, recebendo os documentos enviados para a fila *Registry In*. Logo após, agrupa os *hash* dos documentos em uma árvore de dispersão e os envia para o registro em DLT através do *Rap Broker*.

Apesar de ser semelhante ao apresentado na subseção 4.1.1.1, especializa a responsabilidade do consumidor para o registro dos documentos, dado que os processos de registro e confirmação, que antes eram executados pelo mesmo componente, foram separados. O padrão aplicado nessa mudança foi o de enfileiramento de mensagens (COULOURIS *et al.*, 2011) por meio de um *message broker*. Essa técnica permite o desacoplamento de cada uma das etapas de processamento dos registros, além de possibilitar a aplicação de padrões de retentativa para garantir que, em caso de falha, o processo possa se recuperar.

4.1.2.4 Verify Registry Transaction Processor

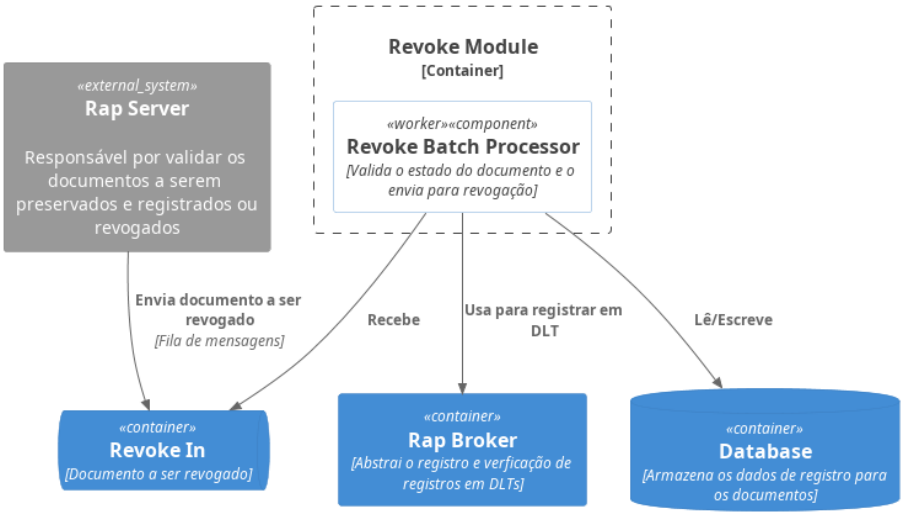
Esse componente também opera como consumidor de mensagens, mas tratando da verificação de confirmação dos registros feitos pelo *Registry Batch Processor*. Isso é feito consumindo as mensagens da fila *Verify In*. Ao recebê-las, verifica as confirmações das transações através do *Rap Broker*, executando até que confirmações ou falhas sejam recebidas. Caso o limite de tentativas seja alcançado, o lote é enviado para reprocessamento pelo *Start Registry Schedule*.

4.2 Módulo de Revogação

4.2.1 Arquitetura legado

Os componentes do módulo de revogação da arquitetura legado representados na figura 7 são detalhados a seguir.

Figura 7. Diagrama para a arquitetura legado do módulo de revogação do *RAP Registry* no nível de componentes.



Fonte: autoria própria (2023).

4.2.1.1 Revoke Batch Processor

O *Revoke Batch Processor* também atua como consumidor de mensagens da fila *Revoke In*, validando o documento a ser revogado e o enviando para revogação através da API do RAP Broker. Após a chamada, ele atualiza o estado do documento no banco de dados e salva os metadados da revogação.

4.2.2 Arquitetura proposta

Os componentes do módulo de revogação representados na figura 8 são detalhados a seguir.

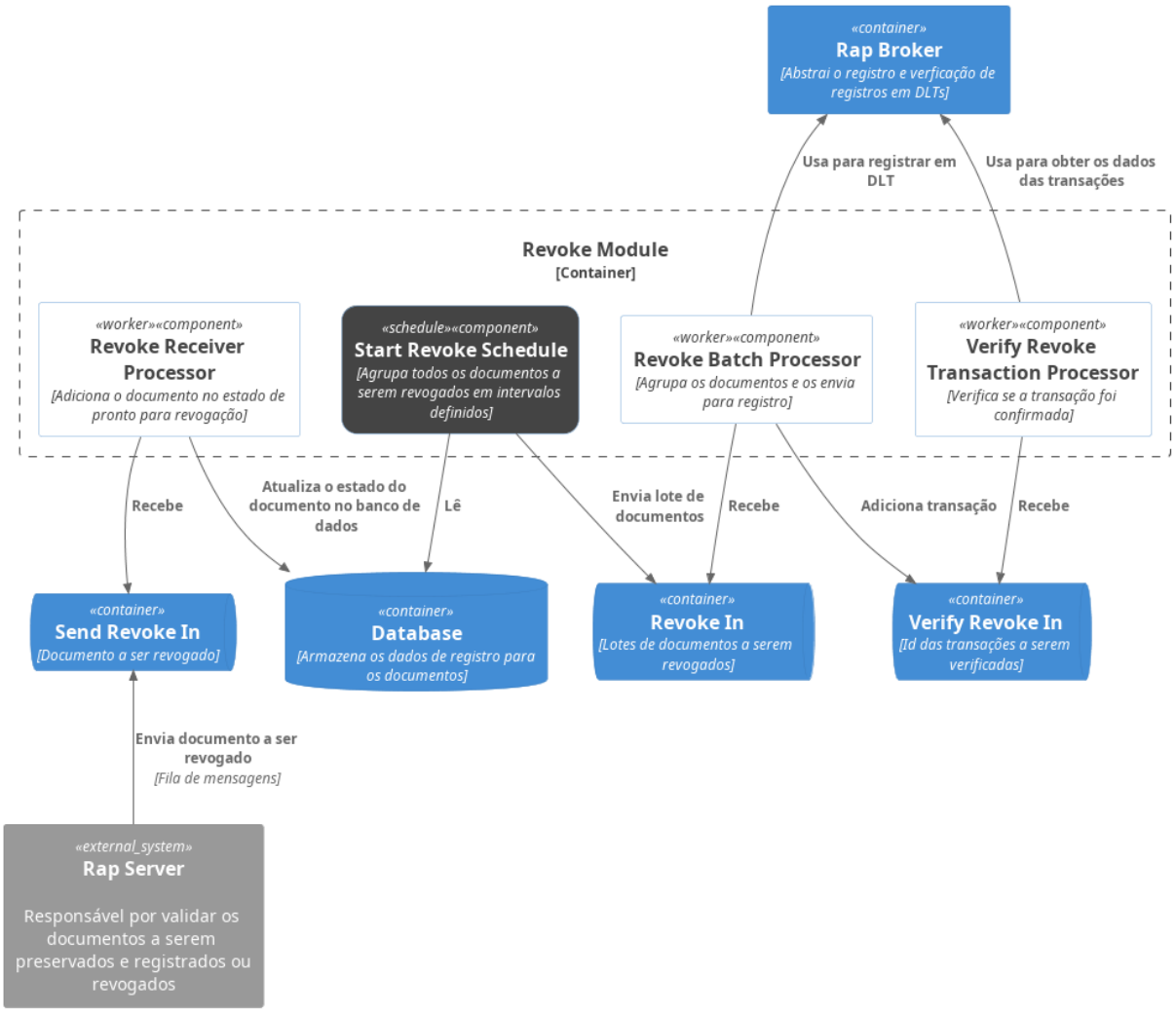
4.2.2.1 Revoke Receiver Processor

O *Revoke Receiver Processor* atua semelhantemente ao visto na subseção 4.1.2.1. Ele consome os documentos a serem revogados, que são enviados pelo *Rap Server* à fila *Send Revoke In*, validando os dados e atualizando o estado para "aguardando revogação".

4.2.2.2 Start Revoke Schedule

Análogo ao componente apresentado na subseção 4.1.2.2, mas que é usado para selecionar documentos prontos para revogação, os quais foram atualizados pelo *Revoke Receiver Processor*, e enviá-los para a fila *Revoke In*. No contexto do *RAP Registry*, uma revogação significa invalidar um registro já confirmado. Isso é feito adicionando-o em um registro em blockchain com um rótulo de revogação em um endereço específico.

Figura 8. Diagrama da arquitetura proposta para o módulo de revogação do *RAP Registry* no nível de componentes.



Fonte: autoria própria (2023).

4.2.2.3 Revoke Batch Processor

Componente similar ao exposto na subseção 4.1.2.3 aplicado ao contexto de revogação. Responsável por agrupar os documentos em lotes com rótulo específico e enviá-los para registro na DLT através do *Rap Broker*.

4.2.2.4 Verify Revoke Transaction Processor

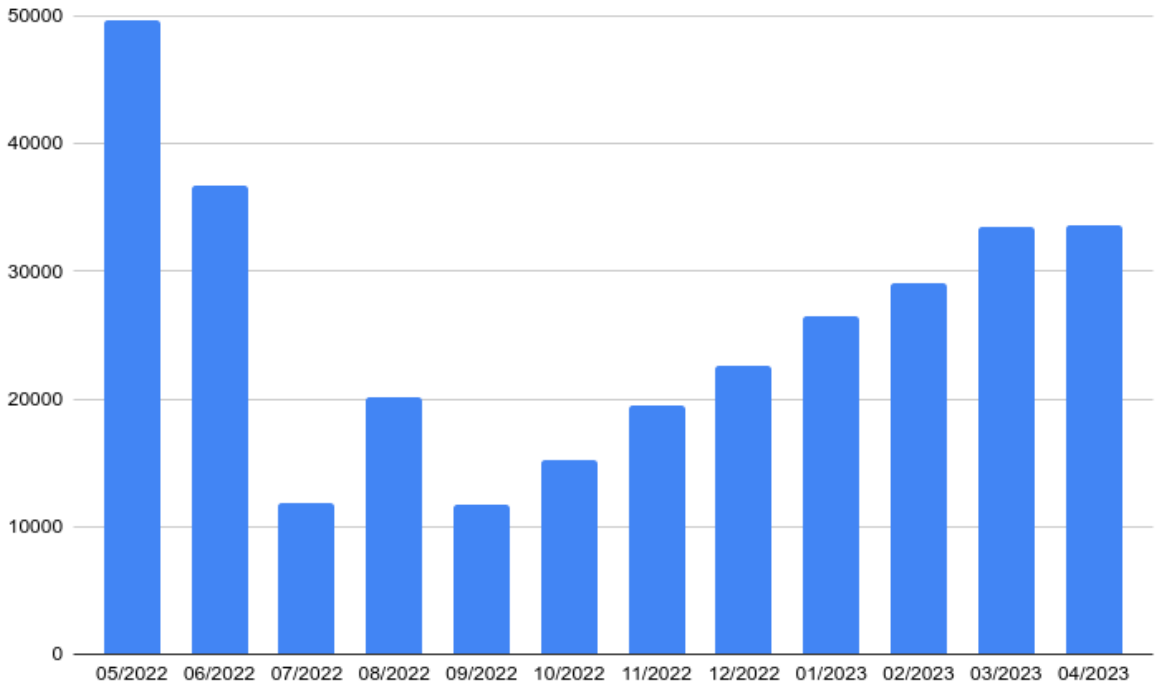
Componente com responsabilidade semelhante ao representado na subseção 4.1.2.4. Utilizado para verificar as confirmações da transação de revogação recebidas da fila *Verify Revoke In*.

5 RESULTADOS

Como resultado desse trabalho, foi desenvolvida uma arquitetura funcional que atende a 78 Instituições de Ensino Superior em todo o Brasil e que já realizou o número de registros e revogações por mês apontados na figura 9, desde o início de sua execução em maio de 2022.

Em média, são realizados 987 registros e revogações por dia, com picos de cerca de 10.000 processamentos, sem falhas fatais até a data de escrita deste trabalho.

Figura 9. Gráfico com número de registros por mês.



Fonte: autoria própria (2023).

Os resultados expressivos obtidos foram alcançados graças à robustez adquirida com a separação das etapas de registro e confirmação, por meio da aplicação da tecnologia de fila de mensagens em conjunto com a estratégia de retentativas. Essas medidas tornaram o fluxo de processamento muito mais confiável e escalável. Além disso, ao agrupar os documentos a serem processados em tarefas agendadas por dia, foi possível reduzir o custo das transações, uma vez que as taxas de registro passaram a ser pagas em lotes relativamente grandes.

Pontua-se que os que detalhes de implementação e as tecnologias adotadas foram omitidos no decorrer do trabalho por questões de confidencialidade, já que o sistema em questão é proprietário da RNP (Rede Nacional de Pesquisa).

6 CONSIDERAÇÕES FINAIS

Esse trabalho se propôs a evoluir a arquitetura do *RAP Registry*, sistema responsável pelo registro de documentação digital acadêmica em *blockchain*. O foco dessa evolução foi aumentar a escalabilidade do sistema, reduzindo os custos com transações e o tornando mais tolerante a falhas.

Isso foi feito identificando as características a serem evoluídas, pesquisando e incorporando padrões de design e mecanismos conhecidos pelo mercado ao projeto, como filas de mensagens, tarefas agendadas e retentativas. Por fim, para verificar a efetividade das adequações, foi feito o monitoramento da aplicação e o levantamento de métricas.

Apesar de empregada no contexto de documentação acadêmica digital, é interessante pontuar que essa arquitetura pode ser efetivamente extrapolada para documentos digitais de naturezas diversas, como uma forma descentralizada de registro e autenticação.

Como indicações para trabalhos futuros, sugere-se uma proposta de desacoplamento dos módulos que compõem a aplicação em serviços totalmente independentes, implementando o padrão arquitetural de microsserviços, considerando os benefícios em relação à resiliência, escalabilidade e manutenibilidade. Além disso, também é recomendado o estudo da aplicabilidade dos contratos inteligentes, *scripts* que são executados em redes *blockchain*, dentro do contexto do serviço aqui trabalhado, levantando os desafios e benefícios da sua utilização. Por fim, considerando a sazonalidade gerada pela diferença entre os calendários das IES, sugere-se o estudo de viabilidade de migração do sistema aqui apresentado para uma arquitetura *serverless*, isto é, cujo provisionamento e gestão de infraestrutura é mantido por um provedor em nuvem, o que pode possibilitar uma economia razoável em relação a custos com infraestrutura.

REFERÊNCIAS

BROWN, S. **The C4 model for Visualising Software Architecture**. 2006. Disponível em: https://c4model.com. Acesso em: 18 Mai. 2023.

COSTA, R. *et al*. **Uso não financeiro de blockchain: Um estudo de caso sobre o registro, autenticação e preservação de documentos digitais acadêmicos**. Workshop em Blockchain: Teoria, Tecnologias e Aplicações (WBlockchain), 2018. Disponível em: https://sol.sbc.org.br/index.php/wblockchain/article/view/2356. Acesso em: 18 Mai. 2023.

COULOURIS, G. F. *et al*. **Distributed Systems**. 5. ed. Upper Saddle River, NJ: Pearson, 2011.

GRAFANA LABS. **Grafana - The open platform for analytics and monitoring**. Disponível em: https://grafana.com/. Acesso em: 18 Mai. 2023.

NAKAMOTO, S. **Bitcoin: A Peer-to-Peer Electronic Cash System**. 2008. Disponível em: https://bitcoin.org/bitcoin.pdf. Acesso em: 18 Mai. 2023.

NOFER, M. *et al*. **Blockchain**. Springer Science and Business Media LLC, 2017. 183–187 p. Disponível em: http://dx.doi.org/10.1007/s12599-017-0467-3. Acesso em: 18 Mai. 2023.

PIERRO, M. D. **What is the blockchain? Computing in Science Engineering**, v. 19, n. 5, p. 92–95, 2017.

PINEDO, M. L. **Scheduling**. Springer International Publishing, 2016. Disponível em: http://dx.doi.org/10.1007/978-3-319-26580-3. Acesso em: 18 Mai. 2023.

PIRES, M. *et al*. **Uma abordagem baseada em brokers para registro de transações em múltiplos livros razão distribuídos**. Workshop em Blockchain: Teoria, Tecnologias e Aplicações (WBlockchain), 2018. Disponível em: https://sol.sbc.org.br/index.php/wblockchain/article/view/2353. Acesso em: 18 Mai. 2023.

PROMETHEUS. **Prometheus - Monitoring system & time series database**. 2023. Disponível em: https://prometheus.io/. Acesso em: 18 Mai. 2023.

TANENBAUM, A. S.; STEEN, M. V. **Distributed Systems**. [S.l.]: Maarten Van Steen, 2023.

ZILE, K.; STRAZDINA, R. **Blockchain use cases and their feasibility**. Applied Computer Systems, v. 23, n. 1, p. 12–20, 2018. Disponível em: https://doi.org/10.2478/acss-2018-0002. Acesso em: 18 Mai. 2023.