GRUPO DE ESTUDO DE OPERAÇÃO DE SISTEMAS ELÉTRICOS - GOP

APLICAÇÃO DE NOSQL E BIG DATA NA INTEGRAÇÃO EM TEMPO REAL DE SISTEMAS DE SUPERVISÃO, CORPORATIVOS E IOT

RICARDO LASTRA OLSEN(1)
CEEE-T(1)

RESUMO

Este trabalho descreve a utilização de ferramentas de NoSQL e Big Data com o objetivo de integrar sistemas supervisórios, IIoT e corporativos em interface operativa unificada. MongoDB serviu como núcleo de processamento e persistência dos dados de tempo real. Foram criados drivers para protocolos de comunicação diversos para coletar dados. Uma interface web foi desenvolvida para apresentação de telas, alarmes, eventos, dados históricos, relatórios e aplicativos diversos. A integração com IoT e sistemas corporativos é via MQTT Broker, permitindo o desacoplamento de sistemas. Os resultados foram satisfatórios, o sistema é usado em produção em dois centros de controle com mais de 70 mil pontos supervisionados.

PALAVRAS-CHAVE

NoSQL, BIG DATA, TI/TO, INTEGRAÇÃO, IoT, IIoT, SCADA, MONGODB, INDÚSTRIA 4.0.

1.0 - INTRODUÇÃO

Na última década observa-se uma evolução muito significativa em sistemas de processamento de grandes quantidades dados (*Big Data*), tais como Computação em Nuvem com escalabilidade horizontal, IoT/IIoT (*Internet of Things* / *Industrial-IoT*), Aprendizagem de Máquina, Bancos de dados NoSQL, entre outros. A integração crescente entre TI (Tecnologia da Informação) e TO (Tecnologia Operacional) nas indústrias está acompanhando este processo (1), especialmente na implantação de projetos IIoT (2) e na aplicação dos conceitos de Indústria 4.0, os quais implicam na disponibilização dos dados de todos os níveis da hierarquia de sistemas corporativos e industriais para a aplicação de técnicas de *Big Data, Analytics* e Inteligência Artificial (3). Adicionalmente, os requisitos de Consciência Situacional para a efetiva operação dos sistemas industriais, cada vez mais, requerem a integração das informações dos sistemas de diversos níveis em uma interface humano-máquina unificada.

Em trabalho de conclusão de curso de Pós-Graduação em *Big Data* e *Data Science* (4), o autor procurou avaliar a hipótese de utilização de banco de dados do tipo documental NoSQL (MongoBD) para servir como núcleo de processamento em tempo real (*soft real time*) de dados de supervisão. Este estudo apresentou resultados promissores e motivou a implementação de um sistema integrador de informações multiníveis de tempo real que serve para diversos tipos de aplicação, entre elas: interface única operativa, integração em nuvem com visualização em interface *web*, plataforma para subestação 4.0 e plataforma de integração de dados para aprendizagem de máquina.

O presente artigo descreve como este sistema foi implementado na prática, já estando em produção em dois centros de controle, e quais as perspectivas de futuras aplicações possíveis para estas novas ferramentas e metodologias.

2.0 - INTEGRAÇÕES MULTINÍVEIS ENTRE SISTEMAS E DISPOSITIVOS

Conforme a norma ISA-95, os sistemas de informação industriais são subdivididos nos seguintes níveis hierárquicos (5):

- Nível 0– Maquinário: controle, sensoriamento e interface incorporados no equipamento.
- Nível 1– Controle Local: CLP's e IHM's.
- Nível 2– Controle de Processos: Sistemas SCADA.
- Nível 3– Execução de Manufatura: MES (*Manufacturing Execution System*).

- Nível 4– Planejamento Corporativo: ERP (*Enterprise Resource Planning*)

Tradicionalmente, estes níveis são representados na forma de pirâmide, onde o nível menor (nível 0) fica na base e o maior (nível 4) aparece no topo. Modernamente, muitas vezes se identifica um sexto nível nesta hierarquia que seria o de nuvem (*cloud*, BI e *Analytics*). IoT/IIoT é um outro componente moderno adicional que se encaixa de forma difusa nestes níveis, sendo pervasivo nas organizações.

As integrações entre níveis normalmente são feitas verticalmente, entre os níveis adjacentes. Este método se torna muito complexo e custoso de ser implementado e mantido pois as integrações são geralmente específicas para as soluções de *software* adotadas, ou seja, não se aplica uma tecnologia padronizada para efetuar as integrações. Qualquer alteração nos dados pode requerer modificações nas integrações de todos os sistemas afetados. O explosivo aumento da quantidade de dados envolvidos, especialmente com a introdução de IoT, bem como a agilidade atualmente requerida para as implementações, torna este tipo de integração cada vez mais inconveniente.

A forma de integração entre sistemas e dispositivos nos diversos níveis escolhida foi através de MQTT *broker*, de maneira a permitir o desacoplamento entre produtores e consumidores de dados. Com isto, é possível a integração horizontal, concentrando dados de qualquer tipo em local único, usando uma tecnologia padronizada, eficiente, escalável e independente de fornecedor. MQTT funciona através de *publish*/*subscribe* sendo agnóstico ao conteúdo dos dados. A conexão se dá de forma segura no sentido do dispositivo ao *broker* com o uso de certificados para criptografia TLS. Não é necessário manter portas TCP abertas para conexão em dispositivos remotos. A implementação do software MQTT *broker* pode suportar o uso de ACLs (*Access Control Lists*) para melhor controle de acesso e proteção dos dados.

Dispositivos e sistemas que suportam MQTT podem se conectar diretamente ao *broker* centralizado ou por meio de *edge nodes*. Os elementos que não suportam MQTT necessitam de um *gateway* ou de um *software* dedicado que possa extrair dados para publicação no *broker* bem como para injetar dados assinados.

Sparkplug-B é um protocolo que roda sobre o MQTT, criado por Arlen Nipper, coinventor do MQTT. Este protocolo padroniza o formato de dados para sistemas SCADA, de forma a possibilitar a autodescoberta das informações disponíveis nos dispositivos. Futuramente, poderão ser obtidos grandes benefícios com a adoção deste protocolo para conectar os sistemas remotos aos centros de controle, tais como: desacoplamento de sistemas, melhor segurança, autodescoberta de dados (cadastro automático das bases de dados), otimização do uso de largura de banda na rede, transferência eficiente de dados históricos.

Para suportar as funcionalidades providas por MQTT e Sparkplug-B necessita-se de um sistema para operação em tempo real bastante flexível e dinâmico, que possa suportar dados numéricos, textuais (*strings*), estruturados (JSON) e ainda documentos ou arquivos binários.

3.0 - NÚCLEO DE PROCESSAMENTO E PERSISTÊNCIA DE DADOS

Foi adotado o banco de dados NoSQL MongoDB como núcleo de processamento e persistência de dados em tempo real (*soft real time*). Esta ferramenta possui as seguintes características:

- Desempenho adequado – capacidade de armazenar e processar grandes quantidades de dados e responder a uma elevada carga de consultas simultâneas.
- Ausência de esquemas (*schema-less*) – permite flexibilidade e expansões no modelo de dados.
- Suporte a vários tipos de dados – dados representados em formato BSON. Suporte a dados numéricos diversos: *strings*, arrays, objetos, estampas de tempo, objetos GeoJSON, código Javascript, dados binários e séries de tempo (somente a partir da versão 5.0). Suporte a armazenamento de grandes arquivos e documentos binários (GridFS).
- Escalabilidade horizontal (*sharding*) – permite distribuir o processamento e armazenamento com a utilização de mais servidores.
- Redundância – funcionalidade nativa de replicação (*replica-sets*), permite obter alta disponibilidade.
- Fluxos de Alterações – funcionalidade nativa para processamento de alterações de forma assíncrona (*Change Streams*). Com o uso de *change streams* é possível interceptar todas as alterações no banco de dados que se deseje observar. Evita a necessidade de se fazer *polling* (consultas cíclicas) no banco de dados.
- Autenticação e autorização – segurança, criptografia na rede e controle de acesso de usuários do banco de dados.
- Encriptação de dados – pode-se criptografar os dados armazenados no banco de dados (apenas na versão *Enterprise*).
- Linguagem de consulta e manipulação de dados – baseada em JSON, a *MongoDB Query Language* é bastante poderosa. Provê ordenação, utilização de indexação, agregações, operações matemáticas, expressões regulares e, operadores lógicos, comparativos, de *arrays*, de bits e geoespaciais.

- Ferramentas de Desenvolvimento, Linguagens de Programação e Conectores – suporta oficialmente as principais linguagens de programação, tais como C++, C#, GO, Java, Node.js, PHP e Python. Possui amplo ecossistema de ferramentas GUI, geradores de relatório, conector para Apache Spark, etc.
- *Open Source*, com suporte comercial e comunitário – suportado comercialmente por MongoDB Inc. e Percona LLC, ou informalmente por ampla comunidade de usuários.

Os servidores MongoDB podem ser arquitetados de diversas formas proporcionando grande flexibilidade: com ou sem particionamento (*sharding*); usando armazenamento em memória nos servidores principais (*in-memory engine*) para a minimização da latência; instalando diretamente no hardware, com Docker, VMs, Kubernetes ou mesmo contratando serviço gerenciado em nuvem.

Graças a todas estas capacidades, a utilização de MongoDB possibilitou uma enorme redução de complexidade na elaboração do sistema de processamento de informações em tempo real, algo que não era possível há alguns anos quando seria necessário desenvolver um complexo subsistema dedicado para esta função.

4.0 - ARQUITETURA DO SISTEMA

Tendo como núcleo de persistência e processamento de tempo real o servidor MongoDB, ao seu redor foram construídos ou integrados os seguintes módulos (ver Figura 1) que compõem o projeto *open-source* batizado de JSON-SCADA (6) (este nome foi escolhido pois a representação JSON é utilizada em todos os níveis do sistema):

- **Base de dados em MongoDB** – Núcleo de processamento e persistência de dados com esquema extensível.
- **PostgreSQL/TimescaleDB** – Banco de dados historiador (TimescaleDB é uma extensão para séries de tempo para o PostgreSQL).
- **Drivers de Protocolos de Comunicação** – Executam a aquisição e distribuição de dados de/para dispositivos e sistemas.
- **Driver MQTT/Sparkplug-B** – Assina e publica tópicos no MQTT *broker*. Este é o método principal para a troca de dados desacoplada entre sistemas e dispositivos.
- **Processador de *Change-Streams*** – Este módulo processa as escritas brutas dos *drivers* de protocolos, aplicando fatores de conversão, determinando alarmes, direcionando dados ao módulo historiador, entre outros processamentos diversos.
- **Processador de *Change-Streams* dedicado** – Processa ações customizadas para alterações no banco de dados. Ex. automatismos e cálculos dinâmicos de baixa latência.
- **Processador de Cálculos** – Executa o processamento cíclico de cálculos compilados.
- **Módulo de *Front-end*** – Fornece a interface *web* operativa, gerencia a autenticação de usuários (através de JWT), verifica direitos e registra as ações de usuários, provê a interface administrativa.
- **Grafana** – Software *open-source* para a criação de painéis de monitoração (*dashboards*). Conecta ao PostgreSQL para consultar dados históricos.
- **Nginx** – Software servidor *web open-source*, permite servir HTTPS com a utilização de certificados de cliente para uma comunicação segura com os clientes em navegador *web*.
- **MQTT *Broker*** – Elemento externo que permite desacoplar sistemas e dispositivos (consumidores e produtores de dados). Pode ser uma solução *open-source* como Eclipse Mosquitto ou VerneMQ.
- **Inkscape (customizado)** – Editor de telas no padrão SVG, estendido para marcação de animações tipo SCADA.

Todos os módulos são portáveis, podendo executar no mesmo servidor ou de forma distribuída. São suportadas as arquiteturas x86/64 e ARM64, Linux em diversas distribuições e Windows 10 ou Server em 64 bits.

O servidor ou *cluster* MongoDB necessita ser configurado no modo de *replica-set* para habilitar a funcionalidade de *change-streams*. A configuração tradicionalmente recomendada para alta disponibilidade de MongoDB é a constituída por três membros. Não é usual a configuração com dois servidores, pois em caso de perda de um deles seria necessário executar uma intervenção manual para ativar o servidor que ficou disponível. Assim ocorre devido ao algoritmo de eleição requerer a maioria de votos dos membros do cluster para eleger o servidor primário, isto é, no cluster com apenas dois servidores, um deles não representa maioria.

A utilização de conexões criptografadas aos servidores MongoDB e PostgreSQL é opcional, todavia é fundamental quando o tráfego de dados ocorre em redes não segregadas.

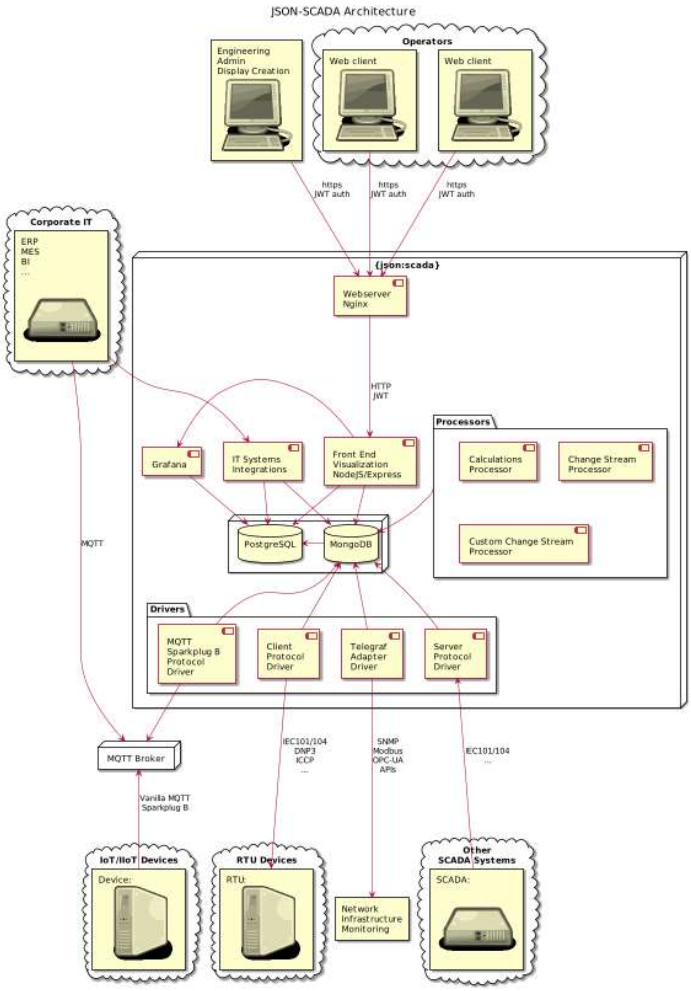

FIGURA 1 – Arquitetura do sistema JSON-SCADA.

5.0 - *DRIVERS* DE COMUNICAÇÃO

Utilizando bibliotecas de código aberto, foram desenvolvidos diversos *drivers* de protocolos de comunicação para permitir o uso do sistema tanto em centros de controle, como servidor de portal de Intranet ou nuvem, bem como nas próprias subestações.

Até o momento foram disponibilizados os seguintes *drivers* de protocolos:

- IEC 60870-5-104 Cliente e servidor (suporta TLS).
- IEC 60870-5-101 Cliente e servidor (serial e IEC101 sobre TCP).
- ICCP Cliente.
- MQTT / Sparkplug B – Publicador e assinante (suporta TLS).
- DNP3 Cliente serial, UDP e TCP (suporta TLS).

Alguns outros protocolos (Modbus, OPC-UA e SNMP) estão disponíveis apenas para monitoramento através do módulo adaptador da ferramenta Telegraf. Esta é uma ferramenta *open-source*, bastante útil também para efetuar a monitoração da infraestrutura de TI.

O driver MQTT/Sparkplug B é um diferencial importante em relação a muitos dos supervisórios tradicionais pois permite executar as seguintes tarefas:

- Desacoplamento entre dispositivos e sistemas.
- Recebimento e publicação de arquivos binários.
- Publicação dos dados supervisórios em árvore de tópicos no MQTT broker.
- Recebimento e interpretação de dados estruturados (JSON).
- Troca de informações textuais, como por exemplo anotações de segurança.

- Autodescoberta de informações e cadastro automático de pontos supervisionados.

6.0 - INTERFACE DE USUÁRIO

A interface de usuário é disponibilizada em navegador. As telas são construídas através de uma versão alterada do *software* de código aberto Inkscape que permite embutir marcação de animações relacionadas aos dados de tempo real em arquivos no padrão SVG.

Os seguintes visores e aplicativos foram desenvolvidos ou integrados:

- **Visor de Telas** – Visualização de telas mímicas, unifilares, etc. Ver figura 2.
- **Visor de Alarmes** – Apresenta os alarmes correntes com diversas opções de filtragem.
- **Visor de Eventos** – Visualiza a sequência de eventos (SOE), nos modos tempo real e histórico.
- **Visor de Curvas (Grafana)** – Software de monitoração em painéis (*dashboards*) Conecta-se ao PostgreSQL.
- **Visor Tabular** – Visualização em tempo real de listas de dados, em forma de tabela.
- **Visor de Desligamentos** – Permite observar todas as funções de transmissão recentemente desligadas. Muito útil em caso de grandes ocorrências.

Uma interface administrativa, também *web*, permite aos usuários configurar as conexões e instâncias de protocolos, os pontos supervisionados e os usuários e seus papéis (RBAC-*Role Based Access Control*). Ver figura 3.

A segurança do sistema é garantida pela utilização de HTTPS, certificados de cliente e uso JWT (*JSON Web Token*) para autenticação e autorização de usuários.

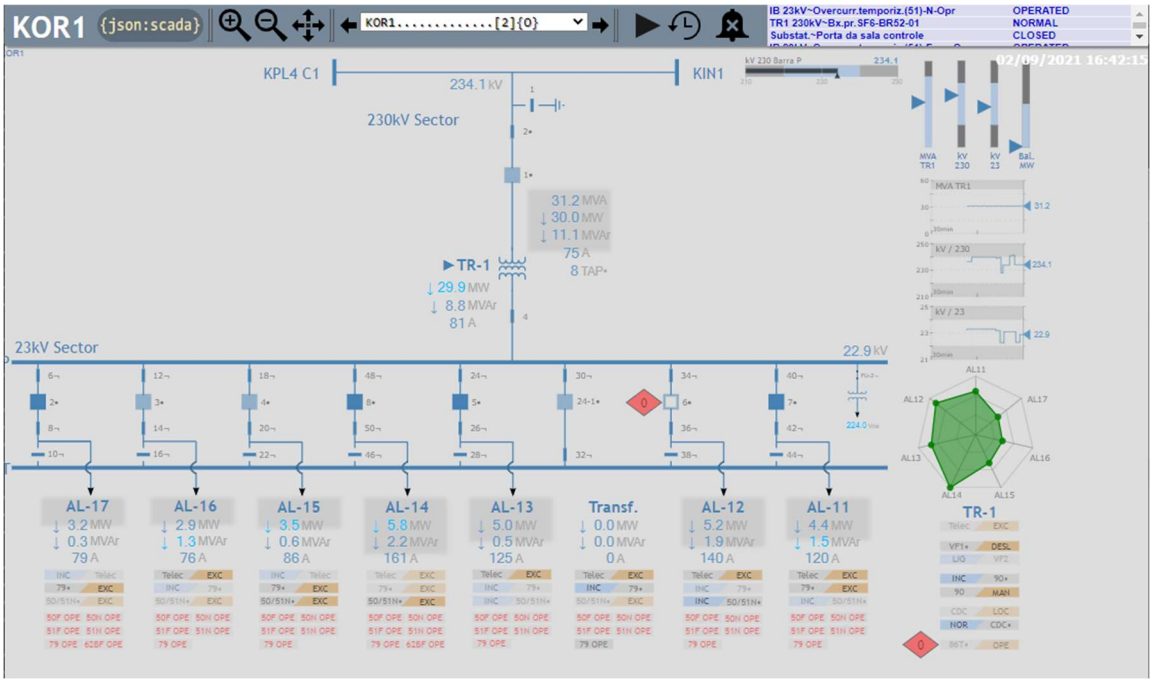

FIGURA 2 – Tela unifilar no padrão de alto desempenho.

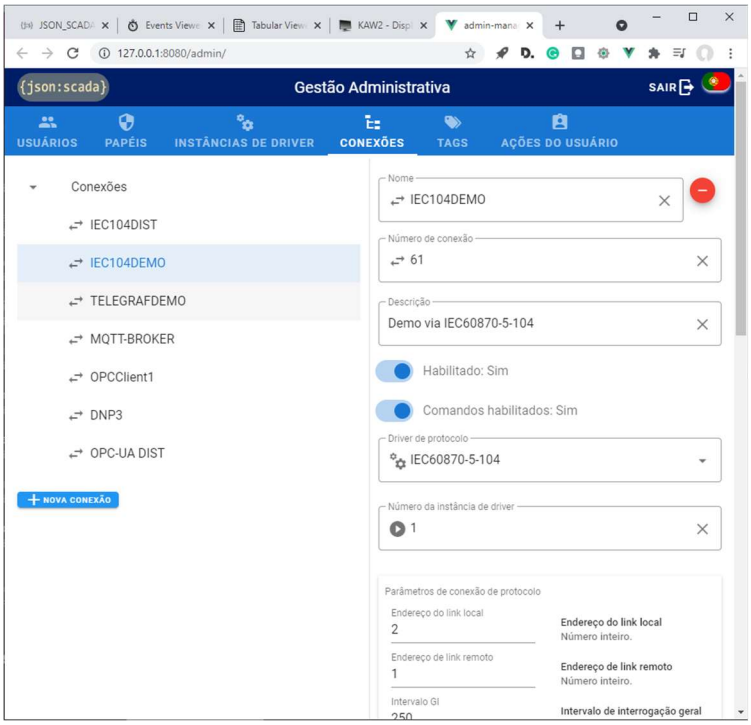

FIGURA 3 – Interface de configuração administrativa.

7.0 - IMPLEMENTAÇÃO EM CENTROS DE CONTROLE

Para os Centros de Controle foi adotada uma arquitetura como mostrada na Figura 4. Existem dois centros independentes (Principal e Reserva). Em cada centro há servidores SAGE que executam a varredura dos sistemas das subestações via protocolos IEC60870-5-101/104. Os sistemas SAGE disponibilizam os dados de supervisão via protocolo ICCP aos sistemas JSON-SCADA. Os sistemas JSON-SCADA fornecem a interface operativa, estando ainda ligados aos MQTT Brokers para integrações adicionais.

Os operadores podem se conectar tanto ao sistema principal como ao reserva. A redundância de centros permite garantir a teleassistência nos seguintes casos de falhas:

- Falha simples de hardware, exemplo servidores ou máquinas clientes.
- Falha de um sistema completo, exemplo SAGE ou JSON-SCADA, em apenas um dos centros.
- Falha de telecomunicação entre subestação e apenas um dos Centros.

Caso haja a perda total do Centro Principal (ex.: falha geral de alimentação), é então necessário deslocar os operadores para o Centro Reserva.

A integração entre sistemas e dispositivos via MQTT *broker* possibilita:

- Sincronização de cartões de segurança de bloqueio de comando nos dois centros e nas IHMs locais das subestações.
- Distribuição automatizada de documentos operativos para os centros e subestações.
- Implantação automatizada de telas e bases de dados nos sistemas JSON-SCADA dos centros e das subestações.
- Integração com os sistemas técnicos corporativos, como por exemplo Gestão de Intervenções.
- Integração com dispositivos de IoT/IIoT.
- Monitoração da infraestrutura de rede e sistemas, sem ocupar as UTRs e sistemas SAGE.
- Desacoplamento entre dispositivos e sistemas.

As funcionalidades providas através de MQTT Broker não são essenciais para a operação remota das subestações.

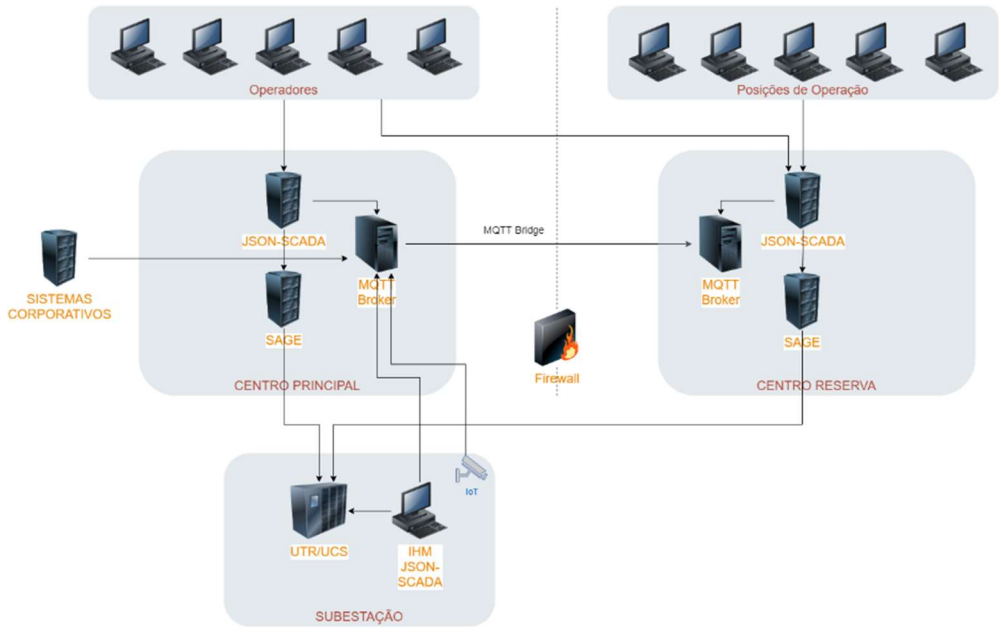

FIGURA 4 – Arquitetura de Centros de Controle Redundantes.

8.0 - CONCLUSÕES E PERSPECTIVAS FUTURAS

O desenvolvimento de um novo sistema com tecnologia de Big Data/NoSQL foi executado com sucesso. A implantação em produção em dois Centros de Operação de subestações de transmissão mostrou que o funcionamento é bastante estável, atendendo de forma eficiente as necessidades operativas, mesmo com uma grande quantidade de dados (mais de 70 mil pontos supervisionados) e muitos usuários (mais de 20 máquinas clientes).

A metodologia de integração horizontal, utilizando MQTT Broker, possibilitou a redução da complexidade nas conexões através do desacoplamento entre sistemas e dispositivos. Funcionalidades antes de difícil implementação se tornaram viáveis, tal como a sincronização dos cartões de segurança entre centros e subestações.

Entre os benefícios alcançados pode-se ainda citar:

- **Facilidade de desenvolvimento de aplicações**: o uso de ferramentas comuns de informática comercial faz com que o desenvolvimento de aplicações customizadas seja muito mais ágil (ex. *Stacks* MEVN/MERN). A extensibilidade do modelo de dados da base de tempo real facilita a integração de dados importados de sistemas externos.
- **Tamanhos das bases de dados e quantidades de conexões**: o uso de MongoDB, o qual permite a escalabilidade horizontal, conjugado ao emprego de MQTT *broker* torna possível armazenar e processar quantidades massivas de dados, bem como conectar um número enorme de dispositivos.
- **Segurança**: a comunicação via MQTT pode ser feita de forma mais segura, sendo que o dispositivo conecta ao broker usando certificados para autenticação e criptografia TLS. Não é necessário deixar portas TCP abertas nos dispositivos remotos.
- **Interface única**: informações oriundas de sistemas diversos são apresentadas em uma única interface operativa, aumentando a consciência situacional.

Como perspectivas futuras salientamos:

- **Agregação de dados para processamento**: o processamento de aprendizagem de máquina pode se beneficiar da disponibilidade de grandes massas de dados em local único.
- **Uso de protocolo Sparkplug B**: a adoção deste protocolo para enviar os dados da subestação para os centros em substituição aos protocolos tradicionais como IEC60870-5-104 poderá trazer diversas vantagens: maior eficiência e segurança; autodescoberta de listas de informações disponíveis; maior semântica na modelagem dos dados através do uso de *templates*; acréscimo de atributos dedicados na comunicação; permitir a conexão simultânea (desacoplada por MQTT *broker*) a mais de um sistema supervisório.

9.0 - REFERÊNCIAS BIBLIOGRÁFICAS

(1) Shilenge, M. Telukdarie A. 4IR Integration of Information Technology Best Practice Framework in Operational Technology. In:2021 Journal of Industrial Engineering and Management. Vol 14 No 3.
< http://dx.doi.org/10.3926/jiem.3429 > (Acesso em 7 Set 2021).

(2) Garimella, P. K. IT-OT integration challenges in utilities. In:2018 IEEE 3rd International Conference on Computing, Communication and Security (ICCCS). p. 199–204.

(3) Lipnicki, P. et al. Future of IoTSP – IT and OT integration. In:2018 IEEE 6th International Conference on Future Internet of Things and Cloud (FiCloud). p.203–207.

(4) OLSEN, R.L. Estudo da Aplicabilidade de Banco de Dados MongoDB como Núcleo de um Sistema SCADA. Monografia de conclusão Pós-Graduação em Big Data e Data Science – UFRGS – 2019.
< https://bit.ly/3t7hpdv > (acesso em 28 Ago 2021).

(5) Brandl, D. What is ISA-95? Industrial best practices of manufacturing information technologies with isa-95 models. 2008. < https://apsom.org/docs/T061_isa95-04.pdf > (acesso em 28 Ago 2021).

(6) OLSEN, R.L. Repositório do projeto JSON-SCADA. < https://github.com/riclolsen/json-scada > (Acesso em 7 Set 2021).

DADOS BIOGRÁFICOS

Ricardo Lastra Olsen, MEng.Nascimento: Porto Alegre – RS, 1965.Graduação em Engenharia Elétrica – UFRGS – Porto Alegre, 1990.Mestrado em Instrumentação Eletroeletrônica – UFRGS – Porto Alegre, 1992.Pós-Graduação em Big Data e Data Science – UFRGS – Porto Alegre, 2019.Trabalha na CEEE-T, na área de Engenharia de Supervisão desde 1998.Áreas de atuação: Sistemas SCADA/EMS, Big Data, IoT, Gráficos de Alto Desempenho, Automatismos em Centros de Controle, Interfaces IHM para controle local e remoto de subestações, Consciência Situacional, sistemas SCADA na nuvem, historiadores de dados, visualização de informações, integração de dados de supervisão aos sistemas corporativos.