VISUALIZAÇÃO DE DADOS DO SISTEMA ELÉTRICO UTILIZANDO COMPUTAÇÃO EM NUVEM

Luís Renato Azevedo de Araujo Silva

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenharia Elétrica, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia Elétrica.

Orientador: Aloysio de Castro Pinto Pedroza

Rio de Janeiro
Setembro de 2014

VISUALIZAÇÃO DE DADOS DO SISTEMA ELÉTRICO UTILIZANDO COMPUTAÇÃO EM NUVEM

Luís Renato Azevedo de Araujo Silva

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA ELÉTRICA.

Examinada por:

_____

Prof. Aloysio de Castro Pinto Pedroza, Dr.

_____

Prof. Jorge Lopes de Souza Leão, Dr.Ing.

_____

Prof. Marcelo Gonçalves Rubinstein, D.Sc.

RIO DE JANEIRO, RJ - BRASIL
SETEMBRO DE 2014

Silva, Luís Renato Azevedo de Araujo

Visualização de dados do sistema elétrico utilizando computação em nuvem / Luís Renato Azevedo de Araujo Silva – Rio de Janeiro: UFRJ/COPPE, 2014.

XV, 92 p.: il.; 29,7 cm.

Orientador: Aloysio de Castro Pinto Pedroza

Dissertação (mestrado) – UFRJ/COPPE/Programa de Engenharia Elétrica, 2014.

Referências Bibliográficas: p. 86 - 87.

1. Sistemas Elétricos. 2. Bancos de dados. 3. Computação em nuvem. I. Pedroza, Aloysio de Castro Pinto II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia Elétrica. III. Título

AGRADECIMENTOS

Primeiramente a Deus, pelas glórias que consegui na vida.

Aos pais, que me ensinaram a vivê-la com dignidade e honestidade e que se doaram por inteiro e renunciaram alguns de seus sonhos para que muitas vezes pudéssemos realizar os nossos.

Aos amigos e familiares que compartilharam e alimentaram meus ideais, compreendendo a falta de tempo, finais de semana de estudo e noites em claro, tendo sempre uma palavra e um sorriso a incentivar para que chegasse ao final de mais uma etapa.

A minha esposa Isabela Caetano Rodrigues, que sempre foi compreensiva e deu total apoio a minha longa jornada de estudos e trabalho.

Minha eterna e singela gratidão a todos os professores do PEE pelas lições e conhecimentos passados, em especial ao professor e orientador Aloysio de Castro Pinto Pedroza, pelo apoio e incentivo na construção desse trabalho.

Vocês fazem parte dessa vitória e da minha história.

"Se você quer saber como foi seu passado, olhe para quem você é hoje. Se quer saber como vai ser seu futuro, olhe para o que está fazendo hoje."

Provérbio chinês

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

VISUALIZAÇÃO DE DADOS DO SISTEMA ELÉTRICO UTILIZANDO COMPUTAÇÃO EM NUVEM

Luís Renato Azevedo de Araujo Silva

Setembro/2014

Orientador: Aloysio de Castro Pinto Pedroza

Programa: Engenharia Elétrica

A eletricidade é uma das principais formas de energia e atua como fator de integração e desenvolvimento de um país. O ambiente no qual engenheiros e técnicos operam os sistemas elétricos está mudando com muita rapidez. O sistema elétrico brasileiro cresce em ritmo acelerado e a tendência é em direção a uma maior complexidade, tamanho e níveis de potência. Apesar das atuais abordagens de análise atenderem suficientemente às necessidades em âmbito de operação, os dados estão disponíveis apenas a um grupo fechado, pequeno e local de controladores, operadores e engenheiros, limitando sua utilização e exploração. Este trabalho introduz uma forma de visualização de parte dos dados de sistemas gerenciadores de energia elétrica por meio da computação em nuvem, fazendo com que usuários menos integrados à complexidade computacional por trás da estrutura possam fazer uso das informações em qualquer lugar e a qualquer momento, podendo agregar valor às formas de utilização da energia.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.).

DATA VISUALIZATION OF THE ELECTRICAL POWER SYSTEM USING CLOUD COMPUTING

Luís Renato Azevedo de Araujo Silva

September/2014

Advisor: Aloysio de Castro Pinto Pedroza

Department: Electrical Engineering

Electricity is one of the main energy sources and acts as an integration and development factor of a country. The environment in which engineers and technicians operate the electrical systems is changing very quickly. The Brazilian electricity system grows at a rapid pace and the trend is toward greater complexity, size and power levels. Despite the current analysis approaches sufficiently meet the needs of the context operation, data are available only to a closed, small and local group of controllers, operators and engineers, limiting its use and exploitation. This dissertation introduces a form of data visualization for power management systems through cloud computing, making users less integrated into computational complexity behind the structure capable of making use of such information anywhere and anytime, adding aggregated value in the forms of energy use.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ANEEL – Agência Nacional de Energia Elétrica

ANSI – *American National Standards Institute*

API – *Application Programming Interface*

ATR - Atributo

AWS – *Amazon Web Services*

BD – Bancos de Dados

CAG – Controle Automático de Geração

CEPEL – Centro de Pesquisa de Energia Elétrica

CI – *Continuous Integration*

CIO – *Chief Information Officer*

CHESF – Companhia Hidro Elétrica do São Francisco

COELBA – Companhia Elétrica do Estado da Bahia

CPU – *Central Processing Unit*

CSV – *Comma-separated Values*

CVS – *Concurrent Version System*

DMS – *Distribution Management System*

DSA – *Digital Signature Algorithm*

EMS – *Energy Management System*

EMR – *Elastic MapReduce*

ENT - Entidade

EQP – Equipamento

EUA – Estados Unidos da América

EVE – Evento

GBH – Gerenciador da Base Histórica

HA – *High Availability*

HD – *Hard Drive*

HPC – *High Performance Computing*

HTTP – *Hypertext Transfer Protocol*

IAAS – *Infrastructure As A Service*

ID – Identificador

IED – *Intelligent Electronic Device*

IHM – Interface Homem Máquina

IA – Inteligência Artificial

IDE – *Integrated Development Environment*

IP – *Internet Protocol*

IT – *Information Technology*

LAN – *Local Area Network*

LIA – Ligação de Aquisição

MCD – Memória Compartilhada e Distribuída

MIT – *Massachusetts Institute of Technology*

MRNN – Modelo Relacional Não Normalizado

N1NF – *Non First Normal Form*

NIST – *National Institute of Standards and Technology*

NOSQL – *Not Only SQL*

ONS – Operador Nacional do Sistema Elétrico

OSI – *Open Systems Interconnection*

PAAS – *Plataform As A Service*

PAS – Ponto Analógico do SAGE

PC – *Personal Computer*

PL – *Procedural Language*

PDS – Ponto Digital do SAGE

PHP – *Hypertext Preprocessor*

PMU – *Phasor Measurement Unit*

RAM – *Random Access Memory*

RHEL – *Red Hat Enterprise Linux*

RSA – *Ron Rivest, Adi Shamir and Leonard Adleman Algorithm*

SAAS – *Software As A Service*

SAGE – Sistema Aberto de Gerenciamento de Energia

SCADA – *Supervisory Control and Data Acquisition*

SCD – Subsistema de Comunicação de Dados

SCM – *Supply Chain Management*

SDK – *Software Development Kit*

SE – Subestação

SEB – Sistema Elétrico Brasileiro

SELINUX – *Security-Enhanced Linux*

SEP – Sistema Elétrico de Potência

SFTP – *SSH File Transfer Protocol*

SGBD – Sistema Gerenciador de Banco de Dados

SGBDOR – Sistema Gerenciador de Banco de Dados Objeto Relacional

SIN – Sistema Interligado Nacional

SO – Sistema Operacional

SOA – *Service Oriented Architecture*

SQL – *Structured Query Language*

SSH – *Secure Shell*

TCP – *Transmission Control Protocol*

TI – Tecnologia da Informação

TR – Tempo Real

UTR – Unidade Terminal Remota

VPC – *Virtual Private Cloud*

XDR – *External Data Representation*

Capítulo I: Introdução

A eletricidade desempenha um papel fundamental na sociedade moderna, por ser uma das principais fontes de energia, atuando amplamente como um fator de integração e desenvolvimento de um país.

A energia elétrica deve chegar aos consumidores dentro de determinados padrões de continuidade e qualidade de suprimento, obtidas à custa de certo investimento no sistema. Se, por um lado, investimentos insuficientes implicarão na perda de qualidade do produto, por outro lado, o excesso de investimentos resultará em um produto final com custo muito elevado, o que irá desestimular o consumo. Assim, planejar e operar adequadamente um sistema de energia elétrica significa chegar a uma solução de compromisso entre a minimização dos custos de investimentos e operação e o atendimento de padrões pré-estabelecidos de qualidade do produto final. O conceito de qualidade do produto de energia elétrica está usualmente associado à continuidade do suprimento e ao atendimento de padrões de regulação de frequência e tensão.

Soma-se a essas exigências o fato de que o ambiente no qual os engenheiros e técnicos que operam os sistemas elétricos atualmente está mudando com muita rapidez. O sistema elétrico brasileiro cresce em um ritmo acelerado, e a tendência é em direção a uma maior complexidade, tamanho e níveis de potência envolvidos na geração, transmissão e distribuição de energia.

Apesar das atuais abordagens de análise atenderem suficientemente às necessidades a âmbito de operação, os dados estão disponíveis a um grupo pequeno e local de controladores, operadores e engenheiros, limitando sua utilização e exploração.

Devido à grande importância desses dados, é visível a necessidade de se concentrar esforços em soluções que possam expandir a capacidade de contextualização e visualização dessas informações.

Esse trabalho pretende introduzir uma forma genérica de visualização de dados do sistema elétrico, de forma que qualquer usuário possa ter fácil acesso à informação sem necessariamente ser especialista ou ter qualquer entendimento sobre a complexidade computacional por trás da infraestrutura de acesso, por meio da utilização de computação em nuvem.

As principais características da computação em nuvem são a agilidade, a escalabilidade, e o acesso por qualquer local e por diferentes aparelhos (telefones celulares e laptops). Permite também o compartilhamento de recursos por um grande grupo de usuários, além de ser um serviço de fácil utilização, por meio de configuração de componentes disponíveis ao invés de instalações.

Neste capítulo serão apresentadas a justificativa e a motivação para o desenvolvimento deste trabalho, assim como os objetivos e contribuições e, ao final do capítulo, será descrito como está organizado o restante deste documento.

I.1 Motivação

Cezar Taurion, em seu livro **[1]**, afirma que à medida que as empresas se interconectam, em redes de valor colaborativas, trabalhando sob demanda em "organizações virtuais", mais e mais infraestrutura computacional também deve operar sob demanda. Alega que o mercado que precisa responder rapidamente às mudanças no contexto de negócios, seja para mais ou para menos, não pode ficar dependente de uma infraestrutura computacional rígida como a imposta nos dias de hoje.

Conforme a computação vai se tornando cada vez mais onipresente, o volume de dados que irá trafegar pelas empresas e que precisará ser manuseado em tempo real será absurdamente maior que o atual. A imprevisibilidade da demanda aumentará também de forma exponencial e será impossível implementar sistemas pelo tradicional método de dimensionamento de recursos pelo consumo em alguns momentos, pois os custos será alto.

Juntando a necessidade de acompanhar a flutuação de demandas de mercado com o crescimento do volume e serviços prestados, e com a subutilização dos recursos computacionais hoje disponíveis nas empresas, é possível chegar a constatação de que é necessário um novo modelo computacional, mais flexível e adaptado à velocidade das mudanças.

A computação em nuvem surge como alternativa e segundo Carr **[2]**, a sua proposta básica é que a provisão dos recursos computacionais fique sob a responsabilidade de empresas especializadas ou que seu fornecimento seja abstraído em

níveis que apenas especialistas tenham capacidade e conhecimento para gerenciá-los e mantê-los, e que também sejam disponibilizados como serviços.

Buyya **[3]**, por sua vez, afirma que a nuvem pode ser considerada uma metáfora para a internet, já que se baseia em abstrações que não revelam a complexidade das infraestruturas, e cada parte é disponibilizada como um serviço e hospedada em centros de dados, distintos ou não, que utilizam hardware compartilhado para computação e armazenamento.

Neste contexto, a provisão de recursos precisa ser vista em várias camadas, onde cada camada representa um gênero específico de recursos que podem ser providos de diferentes formas conforme mostrado na Figura I-1.

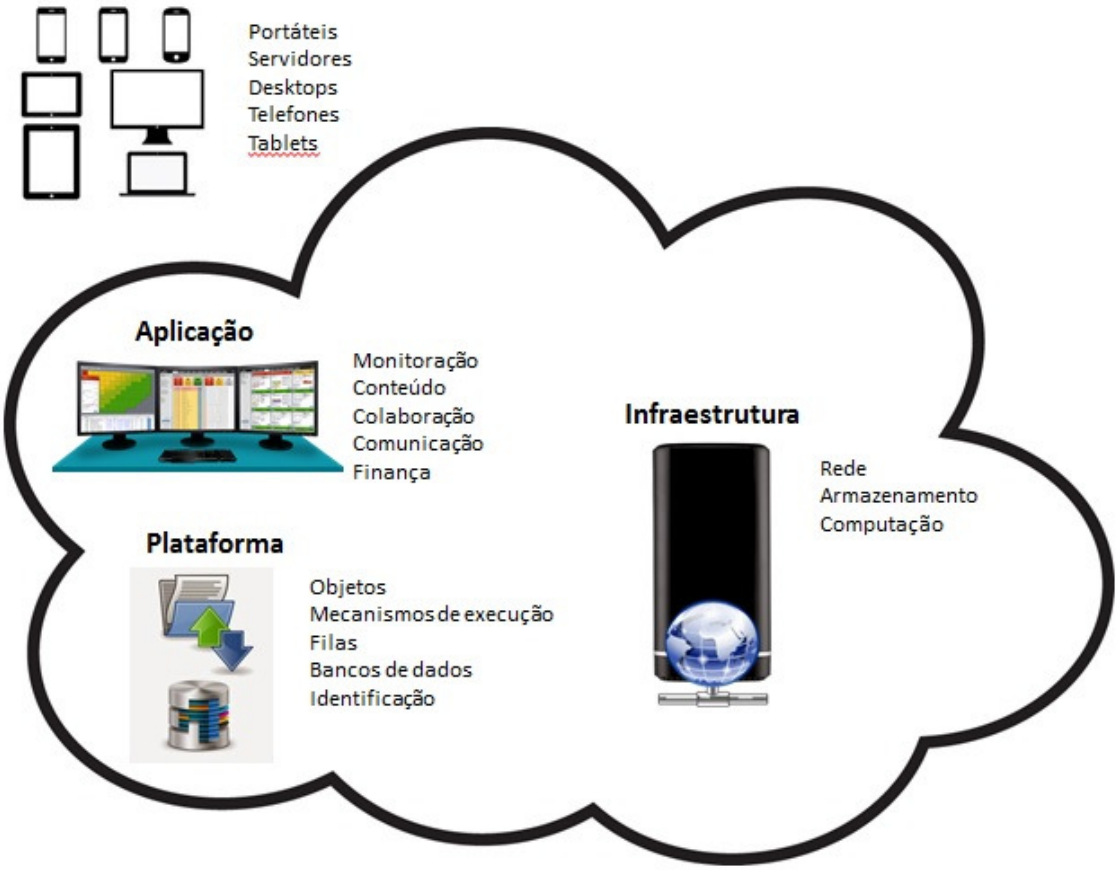


Figura I-1 - Provisão de recursos e serviços através da nuvem.

Esta proposta é extremamente desafiadora porque representa uma grande quebra de paradigma, principalmente se relacionada ao sistema elétrico, visto que até muito pouco tempo atrás as empresas utilizavam exclusivamente os recursos computacionais de forma proprietária, ou seja, os donos são responsáveis pela gestão, manutenção e atualização dos recursos computacionais que dispõem.

Com o advento da computação em nuvem surgem novos desafios e oportunidades de seu fornecimento e utilização e sua contribuição a torna útil para uma possível transição do modelo elétrico tradicional (acoplado aos centros de controle) a um modelo novo, desacoplado, permitindo inclusive o aproveitamento do sistema legado, porém incluindo novas vertentes a respeito do escalonamento de recursos e dinamismo de operação e configuração.

I.2 Objetivo e contribuição

Este trabalho apresenta um estudo sobre a computação em nuvem e o sistema elétrico, onde o objetivo principal é descrever uma estrutura que possa fazer uma interação entre computação em nuvem e sistemas gerenciadores de energia elétrica, abordando alguns conceitos e particularidades básicas sobre ambas as partes e ainda definindo papéis e possíveis cenários de utilização.

A principal contribuição deste trabalho é mostrar a viabilidade da utilização da computação em nuvem em uma arquitetura baseada em SGBDs centralizados e servidores de dados para acesso remoto às informações (descentralizado), tanto para estudos como para outros interesses relacionados ao gerenciamento de sistemas de potência. Dois tipos diferentes de SGBD (não relacional e relacional) são avaliados como parte dessa arquitetura para observar qual deles tem melhor comportamento em determinados tipos de consultas usualmente feitas durante a operação do sistema, seja ela realizada internamente, por código, ou externamente, pelo usuário, na tentativa de simular possíveis situações. Também são avaliados o módulo de acesso ao BD e o servidor de dados, elo de ligação entre o sistema local e a computação em nuvem. A latência de comunicação entre os módulos locais e remotos também é analisada e comparada ao modelo tradicional (totalmente centralizado).

I.3 Estrutura do trabalho

Os próximos capítulos deste trabalho estão estruturados da seguinte forma:

- Capítulo II: apresenta algumas definições disponíveis na literatura sobre os sistemas elétricos, ONS e SCADA (*Supervisory Control and Data Acquisition*) bem como características básicas da computação em nuvem, como os modelos de serviços, papéis de atuação e cenários do ambiente;

- Capítulo III: apresenta a proposta do trabalho, as ferramentas de infraestrutura utilizadas - SAGE e Openshift – e a arquitetura do sistema, destacando os módulos desenvolvidos/utilizados;

- Capítulo IV: define e discute sobre algumas experiências feitas a partir do modelo;

- Capítulo V: apresenta as conclusões do estudo e abre novas propostas a possíveis trabalhos futuros;

Capítulo II: Conceitos

A visualização de grandezas físicas e estados de operação de equipamentos da rede elétrica é feita através do SCADA, algumas vezes também chamado de software supervisório, que através de toda uma estrutura de telemedição, são capazes de coletar dados da rede elétrica em tempo real.

Já a computação em nuvem (em inglês, *cloud computing*) refere-se à utilização da memória e das capacidades de armazenamento e cálculo de computadores e servidores compartilhados e interligados por meio da *Internet*, seguindo o princípio da computação em grade. O armazenamento de dados é feito em serviços que poderão ser acessados de qualquer lugar do mundo, a qualquer hora, não havendo necessidade de instalação de programas ou de armazenar dados. O acesso a programas, serviços e arquivos é remoto, através da *Internet* – daí a alusão à nuvem. O uso desse modelo (ambiente) é mais viável do que o uso de unidades físicas.

Este capítulo resume os dois conteúdos como forma de introdução ao tema proposto.

II.1 Sistema Elétrico Brasileiro

Com mais de um século de utilização e com uma grande capacidade instalada, o setor de energia elétrica no Brasil possui a responsabilidade de agir dentro de padrões técnicos elevados, compatíveis com a tecnologia de nossos dias. O crescimento e a modernização da economia brasileira nas últimas décadas geraram um enorme e sistemático aumento da demanda, principalmente nas grandes áreas urbanas e em regiões predominantemente industriais. Para suprir tal demanda, veio também a necessidade de se aumentar a capacidade de geração do sistema que, por sua vez, requer uma rede de transmissão complexa e que transporta um crescente fluxo de potência.

Os sistemas elétricos de potência podem ser definidos como um conjunto de equipamentos físicos e elementos de circuitos elétricos conectados, que atuam de modo coordenado para gerar, transmitir e distribuir energia elétrica aos consumidores, onde:

- A geração corresponde a tarefa de converter alguma forma de energia (hidráulica, térmica, entre outras) em energia elétrica;

- A transmissão é responsável pelo transporte de energia elétrica dos centros de produção aos centros de consumo, ou até outros sistemas elétricos, interligando-os;

- A distribuição, como o próprio nome sugere, realiza a distribuição da energia elétrica recebida do sistema de transmissão aos grandes, médios e pequenos consumidores.

A Figura II-1 - Rede elétrica desde a geração, transmissão e distribuição até os consumidores finais expõe resumidamente as etapas da rede elétrica desde a geração até os consumidores.

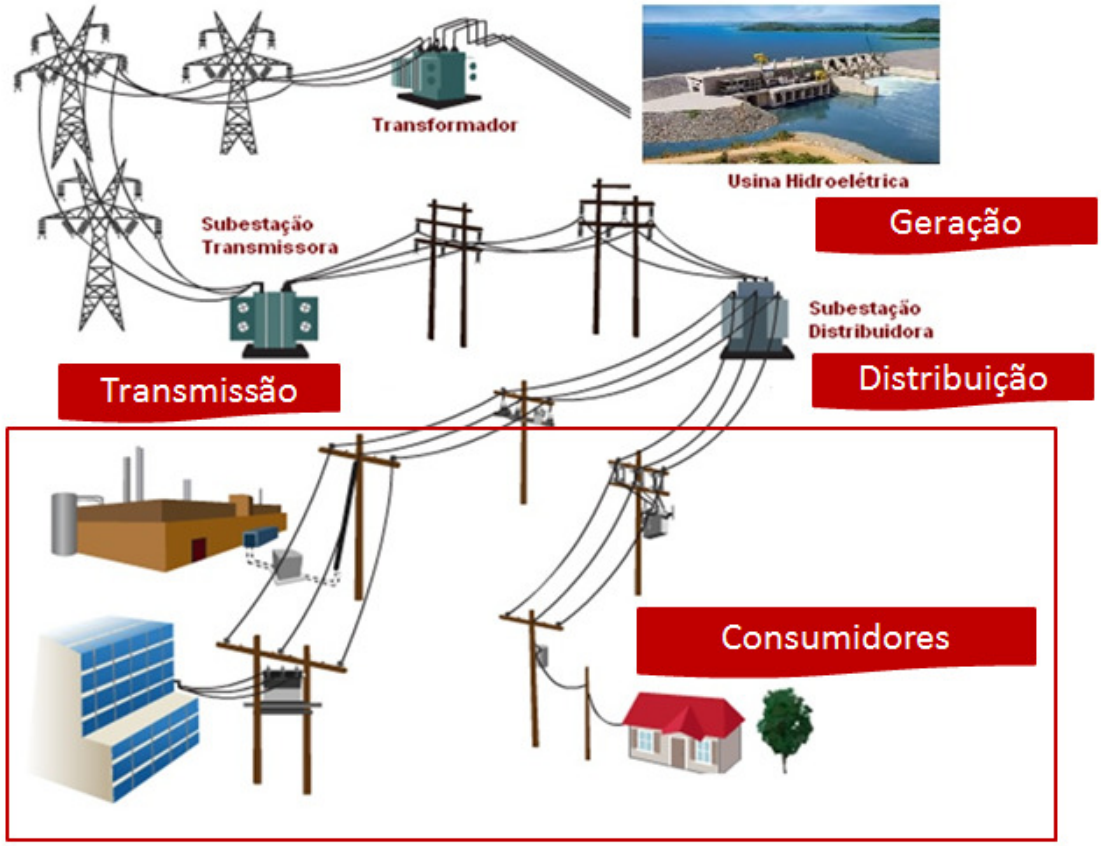


Figura II-1 - Rede elétrica desde a geração, transmissão e distribuição até os consumidores finais

A gestão de energia e operação de sistemas elétricos é uma tarefa difícil e complexa, exigindo um planejamento prévio bem elaborado em paralelo com a análise

de uma grande quantidade de informações técnicas e econômico-financeiras. Além disso, armazenar uma grande quantidade de energia na forma elétrica é inviável.

É possível afirmar que a gestão de energia e a operação desses sistemas tem como função principal o suprimento do seu mercado, levando-se em conta continuidade, qualidade e economia.

Em decorrência do grande desafio e da grande quantidade de informações a serem analisadas e processadas, tornou-se imprescindível automatizar a operação e o gerenciamento, e isso só foi possível graças a implantação de dois tipos de Centros de Controle:

- Um responsável pela gestão dos Sistemas de Geração e Transmissão, denominado Sistema de Gerenciamento de Energia ou EMS (*Energy Management System*);

- Outro responsável pela gestão dos Sistemas de Distribuição de Energia, denominado Sistema de Gerenciamento de Distribuição ou DMS (*Distribution Management System*).

A dificuldade ou até mesmo a complexidade da gestão de energia aumenta à medida que mais agentes (geradores, transmissores ou consumidores), novas tecnologias e forças socioeconômicas são incluídas.

Até meados do século XX os sistemas eram isolados e o gerenciamento de energia elétrica era local. Entretanto, devido à necessidade de um maior aproveitamento energético, e à expansão e interligação dos sistemas elétricos, tornou-se imperativo uma gestão ou controle central para todo sistema, sob jurisdição de uma mesma empresa. Desse modo, os controles locais passaram a compartilhar seus dados com um controle central e assim integrar o nível mais baixo na hierarquia de gestão e controle.

Os controles centrais ficaram responsáveis pela coordenação geral dos sistemas, isto é, passaram a ocupar o nível mais alto na hierarquia.

Os primeiros centros de controle dispunham de dois sistemas independentes:

- Sistema Supervisório (SCADA), responsável pela aquisição e processamento de dados através de unidades terminais remotas (UTRs), pela representação dos dados aos operadores via interfaces homem-máquina e pelo controle de

abertura e fechamento de disjuntores e de dispositivos reguladores de níveis de tensão (reatores, capacitores, taps de transformadores, e outros).

- Controle de Geração, responsável por controlar a geração das usinas do sistema com intuito de estabilizar a frequência, devido às alterações nas cargas demandadas. Derivam desse modelo o Controle Automático de Geração (CAG) e o Despacho Econômico (ou ótimo).

Essa estratégia, de dois sistemas independentes, continuou até o final da década de 1960, quando pesquisadores começaram a analisar o problema de um ponto de vista sistêmico, tendo como motivação diversos fatores, entre eles blecautes frequentes que ocorriam nas redes elétricas em todos os países (principalmente no Nordeste dos EUA), o avanço tecnológico nas áreas de computação e telecomunicação e a evidente necessidade de se dispor de estratégias de controle e operação mais rápidas e efetivas, em resposta à fragilidade dos sistemas elétricos face às mudanças dos estados operativos.

A partir daí surgiu o conceito de segurança e de controle de segurança em tempo-real definido como um sistema integrado de controles manuais e automáticos, responsáveis pela permanente operação do sistema elétrico frente às várias condições de operações.

O Sistema Elétrico Brasileiro tem características próprias, o que o torna singular entre outros países. A matriz energética é baseada principalmente em usinas hidráulicas. A geração hidráulica, baseada em grandes reservatórios de água, depende da ocorrência de chuvas nas regiões das barragens e, se não chover o suficiente, os reservatórios esvaziam e as usinas tem que diminuir ou até parar a geração de energia elétrica.

Por esta razão, a quantidade de energia elétrica que cada usina vai gerar deve ser planejada, calculada levando-se em conta fatores sazonais, históricos de chuvas, e previsão do consumo de energia elétrica.

No Brasil pode-se pensar no sistema elétrico como quatro grandes subsistemas de energia: Norte, Nordeste, Sul e Sudeste (a região central fica dividida entre o Sul e o Sudeste). Estes quatro subsistemas são interligados através de linhas de transmissão de longa distância e alta capacidade de transmissão, formando o Sistema Interligado Nacional (SIN), oferecendo ao sistema a possibilidade de se levar grandes blocos de

energia de um subsistema a outro. Esta interligação garante uma melhor gerência dos recursos energéticos do país. Por exemplo, quando há uma temporada de seca na região norte e há risco de redução dos níveis dos reservatórios do norte, pode-se exportar energia da região Sul, onde seria época de chuvas e os reservatórios estariam em sua capacidade máxima de armazenamento, evitando-se ao mesmo tempo, o vertimento da água excedente na região Sul.

Siqueira **[4]** detalha parte da operação do sistema elétrico brasileiro e alega que forma interligada proporciona uma série de vantagens, tais como:

- Ganho em energia firme (energia mínima que permite uma operação contínua das plantas hidrelétricas num período de tempo);

- Minimização de riscos de interrupção no suprimento de energia, devido ao fato da conservação de reservas energéticas para suportar períodos de baixa hidrologia;

- Manutenção de níveis adequados de confiabilidade da rede elétrica;

- Utilização de energia hidráulica disponível em outros pontos do sistema, de maneira a diminuir os custos operativos e reduzir os preços da energia elétrica para os consumidores.

- Uma adequada reprogramação da geração, ajustando-se a mesma às condições verificadas de demanda e hidrologia.

- Outros usos dos reservatórios: navegabilidade, controle de cheias, irrigação, etc.

Apenas 3,4% da capacidade de produção de eletricidade do país encontram-se fora do SIN, em pequenos sistemas isolados, localizados principalmente na região amazônica. Mais de 80% da capacidade de energia elétrica no Brasil são hidrelétricas localizadas em diferentes bacias hidrográficas. A matriz energética se completa com as usinas termelétricas convencionais e nucleares.

O Sistema elétrico brasileiro é formado por empresas de capital privado e empresas estatais, que operam sob concessão, autorização ou permissão do estado. Porém, por suas características sinérgicas, o sistema elétrico tem que ser operado de

forma monopolista. O ONS **[5]**, Operador Nacional do Sistema, tem este papel no setor elétrico brasileiro, como agência reguladora, com o objetivo de garantir a confiabilidade e eficiência da operação. O ONS, entre outras funções, determina a geração de cada usina e estipula pesadas multas para o caso do não cumprimento de suas decisões.

De forma geral, os grandes usuários de energia elétrica, principalmente aqueles com uma alta demanda contratada, localizados em grandes áreas industriais e cada vez em maior número, também demandam melhores condições operacionais das companhias geradoras ou distribuidoras de energia elétrica. Visando prevenir indesejáveis e frequentes quedas, interrupções não programadas, oscilações abruptas de carga e tensão, variações de frequência, ou falhas nas redes de transmissão e distribuição, índices e padrões para o controle de qualidade são estabelecidos por leis regulamentares do setor. Simultaneamente, há um aumento na complexidade do planejamento, administração, supervisão e controle necessários para atender esta demanda.

Como consequência deste crescimento, os sistemas de energia podem ser operados próximos a seus limites, demandando assim um planejamento mais cuidadoso para que sejam evitados maiores problemas. Grandes racionamentos já marcaram a história, inclusive recente, do Brasil, com reflexos desastrosos na vida urbana e na produção industrial. Na década de 1950, entre as medidas inicialmente tomadas para permitir o desenvolvimento dos sistemas de geração estava a realização do inventário dos recursos elétricos. Em meados da década de 1960, técnicos estrangeiros foram contratados para realizarem estudos, o que representou o marco da introdução do planejamento de sistemas de energia elétrica no Brasil. Desde então, a crescente complexidade dos sistemas de energia elétrica tem sido acompanhada por um notável desenvolvimento tecnológico. Procura-se, até hoje, aliar experiências e avanços tecnológicos internacionais com a real necessidade de solução dos problemas brasileiros. As características físicas e socioeconômicas do país impõem a formulação de soluções específicas para a nossa realidade e não disponíveis em outros países.

II.2 SCADA

A operação do sistema elétrico de potência depende totalmente das informações de estados, da análise dos dados e da rapidez na tomada de decisões das intervenções dos operadores.

Segundo o Schweitzer Engineering Laboratories **[6]**, a operação manual, baseada em percepções e ações de operadores, envolve riscos elevados para o sistema e para os consumidores, que por sua vez são minimizados com a utilização da tecnologia digital em sistemas de supervisão e controle. O atual porte, importância e dependência social e econômica da vida moderna não permitem, na operação dos sistemas elétricos, a utilização de métodos operacionais ultrapassados. Ainda, o estabelecimento de rígidas regras para os serviços de suprimento de energia elétrica estimula cada vez mais as concessionárias a investirem em sofisticadas ferramentas para aumentar a visibilidade, a agilidade e a precisão para operar o sistema elétrico como um todo.

Os sistemas de gerenciamento de processos tornaram-se popular com a evolução da tecnologia digital, intensificada a partir da década de 1960. Com o intuito de aumentar a velocidade da operação, melhorar a confiabilidade, a segurança **[7]** e a qualidade do suprimento de energia elétrica, a partir desta década, os primeiros projetos SCADAs passaram a fazer parte de concessionárias de energia elétrica em todo o mundo.

Com base em tecnologia “mainframe”, os sistemas eram de custo elevado e operação muito complexa. Exigiam muita interação humana com o processo e, gradativamente, suas funções de operação foram evoluindo em função do avanço tecnológico, permitindo sistemas mais complexos no automatismo. A disponibilização de plataformas computacionais de menor porte (PC) tornou o sistema mais simples e acessível em termos de hardware.

Em algumas empresas, os primeiros SCADAs foram aplicados somente nas subestações e na malha principal da rede em função da dificuldade de se obter o retorno financeiro dos investimentos necessários para sua implantação. Atualmente, devido à necessidade de atendimento aos requisitos rígidos de suprimento, à disponibilidade de tecnologia e aos custos de implantação e operação, o emprego do sistema foi estimulado, desde a geração à distribuição de energia elétrica.

A operação do sistema elétrico, normalmente feita por concessionárias de energia elétrica, está baseada na coleta, processamento, análise e armazenamento de dados analógicos e digitais adquiridos em subestações. Estes dados são medidos por

meio de sensores instalados nos equipamentos das respectivas plantas, que podem ser usinas de geração de energia, subestações de transmissão, subestações de distribuição, redes de transmissão/distribuição. Após a obtenção remota dos dados, estes são transmitidos para um centro computacional de gerenciamento.

Os sistemas são compostos por várias camadas, chamadas de níveis hierárquicos. Cada um é responsável por uma parte do processo de aquisição e controle. O SCADA é formado basicamente por módulos digitais de aquisição e controle (entradas e saídas digitais), por módulos de aquisição de entradas e saídas analógicas, por processadores de lógicas, por rede de comunicação, por interfaces com usuários, por equipamentos de comunicação e softwares para diversas aplicações.

O exemplo da Tabela II-1 - Exemplo de estrutura de hierarquia SCADA apresenta uma estrutura de hierarquia para SCADAs aplicada a sistemas de energia elétrica. Quanto maior a quantidade de níveis, maior é a complexidade no tráfego de informações e maior o investimento necessário para implantação e manutenção do sistema.

Tabela II-1 - Exemplo de estrutura de hierarquia SCADA

Centro de operação do sistema Plataformas computacionais – SCADA remoto	Nível 5
Centro de operação regional Plataformas computacionais – SCADA remoto	Nível 4
Interface homem-máquina local Plataforma computacional – SCADA local	Nível 3
Unidade central de subestação Processadores, gateways de comunicação, IEDs (*Intelligent Electronic Device*)	Nível 2
Relés de proteção, unidades de controle de Bay, medidores de equipamento	Nível 1
Disjuntores, seccionadoras, controles de Tap, reguladores de carga e frequência	Nível 0

A principal função do SCADA, a obtenção de dados, a partir da qual todo o sistema se torna possível, é realizada pelos equipamentos de proteção, controle e medição, baseados em processadores digitais, chamados de *Intelligent Electronic Devices* (IEDs).

Os IEDs são responsáveis pela interface para obtenção dos dados que alimentam a base do SCADA com:

- Medidas analógicas como tensões, correntes, grandezas elétricas derivadas (potência e energia), frequência, temperaturas, velocidades etc. – grandezas medidas em números reais diretamente de instrumentos de medidas (transformadores de corrente, transformadores de potencial, transdutores etc.);

- Medidas digitais como estados físicos de equipamentos de manobras (a exemplo de disjuntor aberto/fechado, seccionadora aberta/fechada), estado de saúde de equipamentos (como retificador normal/falha, relé normal/falha), leituras de posição (posição de TAP de transformador medida por meio de pontes de diodos) – grandezas medidas digitalmente por estado 0 ou 1 (verdadeiro/falso) fornecidas diretamente pelos equipamentos elétricos, por relés auxiliares ou por medidores de posição;

- Controle ou envio de ordens para mudanças de estados de equipamentos – são os comandos para mudanças de estados, que são encaminhados aos equipamentos elétricos via interface física (contatos). A origem de uma ordem de controle é decorrente de uma necessidade operativa ou o resultado de um processamento lógico (automatismo). O início dos processos de mudanças de estados dos equipamentos deve ser possível nos diversos níveis hierárquicos, com a possibilidade de configurações de restrições;

- Gerenciamento de comunicação – transformação dos dados medidos e/ou calculados em informações inteligíveis para o nível hierárquico superior. As informações são transmitidas em linguagem estruturada, ou seja, via protocolos de comunicação.

A unidade central de processamento da subestação exerce a função de concentração de dados, gerenciamento de local de dados, gerenciamento de comunicação, processamento de lógicas no nível da subestação e permite a execução de todas as funções de supervisão e controle remoto da instalação com velocidade, segurança e a menor intervenção humana possível.

A habilidade para controlar o sistema elétrico em tempo real é uma função-chave, que depende fundamentalmente dos equipamentos de aquisição e das facilidades de comunicação entre as diversas camadas.

A disponibilidade de todos os equipamentos que compõem os SCADAs deve ser uma preocupação constante das concessionárias. A responsabilidade pela qualidade do suprimento não é mais tão fortemente dependente de ações humanas e muito mais dependente do bom funcionamento do sistema. Quanto menores as taxas de falhas dos equipamentos utilizados, melhor será a percepção do cliente para a qualidade do suprimento.

A comunicação é feita por diversos tipos de meios físicos e de forma estruturada, preferencialmente padronizada por normas e em múltiplos arranjos de arquitetura de rede de comunicação utilizados para a troca de informações entre os níveis hierárquicos de SCADAs.

O SCADA é um sistema de múltiplos dados usualmente chamados de pontos. Os pontos de um sistema de supervisão e controle podem ser do tipo físico (medido ou monitorado) ou ponto lógico, que são obtidos por meio de cálculos internos nos IED ou por processamento de softwares em plataformas computacionais. O conjunto dos pontos físicos e lógicos compõe a base de dados do sistema que é utilizada para a operação local, operação à distância, registros de eventos, históricos e automatismos no sistema elétrico.

As interfaces com os operadores, em que os dados coletados e processados estarão disponíveis para serem monitorados, supervisionados e controlados são as chamadas Interfaces Homem Máquinas (IHMs ou do Inglês HMIs – *Human Machine Interfaces*).

As plataformas computacionais com softwares específicos para a função SCADA são utilizadas como IHM. Os dados de operação do sistema são apresentados de forma padronizada independentemente dos equipamentos utilizados para a coleta e processamento das informações em cada planta e normalmente são também as plataformas onde são verificados os passos e resultados dos automatismos realizados na operação do sistema.

A massa de dados coletada e a capacidade de processamento dos diversos níveis são usadas para o desenvolvimento de automatismos, com amplo uso na operação do sistema elétrico.

II.3 Computação em Nuvem

A denominação *cloud computing* foi popularizada por volta de 2008. Também conhecida no Brasil como computação nas nuvens ou computação em nuvem, *cloud computing* se refere, essencialmente, à ideia de utilizar-se, em qualquer lugar físico e independente de plataforma computacional, as mais variadas aplicações por meio da *internet* com a mesma facilidade de tê-las instaladas localmente no computador.

Este modelo é um sonho de longa data da computação como um serviço público e tem o potencial de transformar uma grande parte da indústria de TI, tornando o software ainda mais atraente como um serviço e moldando a forma como o hardware de TI é projetado e comprado. Desenvolvedores com idéias inovadoras para novos serviços de Internet já não requerem grandes custos em hardware para implantar o seu serviço ou mão-de-obra para operá-lo. Eles não precisam se preocupar sobre o excesso de provisionamento de um serviço cuja popularidade não satisfaz as suas previsões, desperdiçando assim recursos onerosos, ou pelo pouco provisionamento de recursos para aqueles serviços que se tornam muito populares, faltando assim potenciais clientes e receitas. Além disso, empresas com grandes tarefas orientadas a lotes podem obter resultados tão rapidamente quanto a sua programação puder escalar, já que utilizando 1.000 servidores em uma hora não custa mais do que usar um servidor durante 1000 horas. Esta elasticidade de recursos, sem ter que se pagar um preço elevado para o largo escalonamento, é sem precedentes na história da TI.

II.3.1 Principais Características

É habitual o armazenamento de arquivos e dados dos mais variados tipos e a utilização de aplicações de maneira *on premise*, isto é, instaladas em computadores pessoais. No ambiente corporativo, este cenário é apenas um pouco diferente, já que nele é mais fácil encontrar aplicações disponíveis em servidores, onde estas aplicações podem ser acessadas por qualquer terminal autorizado por meio de uma rede.

Na abordagem *on premise*, a principal vantagem está no fato de ser possível, pelo menos na maioria das vezes, utilizar as aplicações mesmo sem acesso à *internet* ou à rede. Em outras palavras, é possível usar estes recursos de maneira *offline*. Entretanto, todos os dados gerados estão restritos a este computador, exceto quando compartilhados em rede, o que não é muito comum no ambiente doméstico. Mesmo no ambiente

corporativo, esta situação pode gerar algumas limitações, como a necessidade de se ter uma licença de um determinado *software* para cada computador, por exemplo.

A evolução constante da tecnologia computacional e das telecomunicações faz com que o acesso à *internet* torne-se cada vez mais amplo e cada vez mais rápido. Em países mais desenvolvidos, como Japão, Alemanha e Estados Unidos, é possível ter acesso rápido à internet pagando-se muito pouco. Esta tendência cria a condição perfeita para a popularização da computação em nuvem, fazendo com que o conceito se torne conhecido no mundo todo, inclusive no Brasil.

Segundo Armbrust **[8]** *Cloud Computing* refere-se tanto a aplicativos entregues como serviços através da *Internet* como aos hardwares e softwares de sistemas nos datacenters que fornecem esses serviços. O conjunto de hardware e software dos datacenteres é o que chamamos de uma nuvem. Para o NIST **[9]** a computação em nuvem é um modelo para permitir de maneira ubíqua, conveniente e sob-demanda o acesso a um conjunto configurável de recursos computacionais que podem ser rapidamente fornecidos e liberados com um mínimo de esforço de gerenciamento ou interação com o provedor de serviços.

Esse modelo é possível graças aos conceitos de virtualização que, segundo Vouk **[10]**, permite a abstração e isolamento de funcionalidades de baixo nível e de hardwares subjacentes. Isto faz com que funções de alto nível e o compartilhamento e agrupamento de recursos físicos sejam portáveis. O conceito tem se aprimorado com o tempo e hoje em dia é aplicado em múltiplos aspectos da computação, como memória, armazenamento, processamento, software, redes e serviços.

Com a *cloud computing*, muitos aplicativos, assim como arquivos e outros dados relacionados, não precisam mais estar instalados ou armazenados no computador do usuário ou em um servidor próximo. Este conteúdo passa a ficar disponível nas “nuvens”. Ao fornecedor da aplicação cabem todas as tarefas de desenvolvimento, armazenamento, manutenção, atualização, backup e escalonamento. O usuário não precisa se preocupar com nenhum destes aspectos, apenas com acessar e utilizar. A Figura II-2 ilustra essa situação de maneira simplificada.

Figura II-2 - Plataforma, infraestrutura e aplicação podem ser disponibilizadas pela nuvem

Do ponto de vista do hardware, três aspectos são novos em *Cloud Computing*.

- A ilusão de infinitos recursos de computação disponíveis sob demanda, eliminando assim a necessidade dos usuários de planejar muito à frente de provisionamento;

- A eliminação de um compromisso up-front, permitindo assim que as empresas que tenho um aumento gradativo de recursos de hardware apenas quando há um aumento em suas necessidades;

- A capacidade de pagar por uso de recursos de computação em curto prazo, conforme necessário (por exemplo, processamento por hora e armazenamento por dia) e liberá-los quando necessário, portanto premiando a conservação à medida que vão liberando armazenamento quando não são mais úteis. Este modelo geralmente é utilizado quando se exigem grandes taxas de processamento e armazenamento.

Um exemplo prático desta nova realidade é o *Google Docs*, serviço onde os usuários podem editar textos, fazer planilhas, elaborar apresentações de slides, armazenar arquivos, entre outros, tudo através da internet, sem necessidade de se ter programas como *Microsoft Office* ou *OpenOffice* instalados em suas máquinas. O que o usuário precisa fazer é apenas abrir o navegador de internet e acessar o endereço do aplicativo para começar a trabalhar, não importando qual o sistema operacional ou o computador utilizado para este fim. Neste caso, o único cuidado que o usuário deve ter é o de utilizar um navegador de internet compatível, o que é o caso da maioria dos *browsers* da atualidade.

As vantagens da utilização da computação em nuvem são inúmeras. De acordo com Sousa, Moreira e Machado **[11]**, o que distingue essa solução das demais são:

- **Self-service sob demanda**: a capacidade unilateral do usuário de adquirir recursos computacionais conforme desejar. Hardware e Softwares podem ser reconfigurados e orquestrados de forma transparente e simples, ou seja, permitem uma personalização de ambiente;

- **Amplo acesso:** o acesso é feito por meio da rede e por mecanismos padronizados que permitem o uso de plataformas *thin* ou *thin client*, como celulares, *laptops, tablets*. A interface de acesso não condiciona mudanças no ambiente de trabalho, como SOs e linguagens de programação;

- **Pooling de recursos:** os recursos são organizados em conjuntos para servir múltiplos usuários, por meio de diferentes recursos físicos e virtuais, atribuídos dinamicamente sob-demanda. Não é necessário saber a localização física, podendo apenas especificar sua localização em níveis de abstração.

- **Elasticidade rápida:** recursos são adquiridos e liberados rapidamente, elasticamente e até automaticamente. A complexidade é invisível ao usuário e a virtualização ajuda nessa característica a medida que cria várias instâncias de recursos requisitados utilizando um único real **[12]** ;

- **Serviço medido:** a solução em nuvem controla e ao mesmo tempo aperfeiçoa o uso de recursos por medições. A automatização é feita segundo

algum nível de abstração mais indicado para o serviço, seja ele armazenamento, processamento, largura de banda ou contas ativas. Os recursos podem ser monitorados e controlados, oferecendo transparência ao provedor e ao usuário;

Independente da aplicação, com a *cloud computing* o usuário não necessita conhecer toda a estrutura que há por trás, ou seja, ele não precisa saber quantos servidores executam determinada ferramenta, quais as configurações de hardware utilizadas, como o escalonamento é feito, onde está a localização física do *datacenter*. O que importa ao usuário é saber que a aplicação está disponível em "nuvens", não importando a forma.

II.3.2 Saas, PaaS e IaaS

A computação em nuvem permite acessar um conjunto de recursos, soluções e aplicativos armazenados em um servidor *online*. Através de uma conexão de *internet* é possível criar, editar e compartilhar arquivos, gerenciar serviços, infraestrutura e capacidade de armazenamento, facilitando os processos de trabalho e aumentando a produtividade no ambiente corporativo.

Para que a computação em nuvem funcione adequadamente, algumas estruturas são essenciais. Na Figura II-3 - Tipos de computação em nuvem estão listadas as três camadas principais da c*loud computing*:

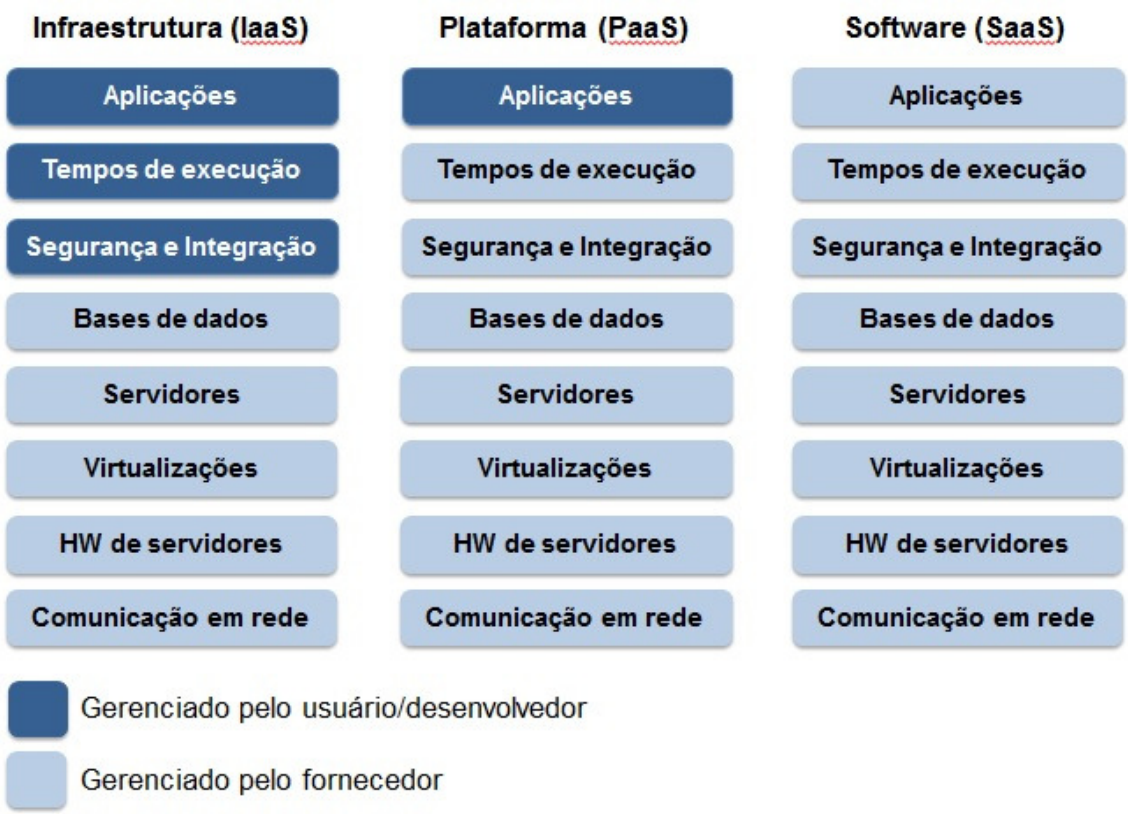


Figura II-3 - Tipos de computação em nuvem

O IaaS é a camada básica e estrutural para o funcionamento da c*loud computing*. Ela representa toda a parte física como servidores, *data centers, hardwares* e equipamentos de energia e refrigeração, que possibilitam o armazenamento e a transmissão de dados e aplicações de forma rápida por meio da *internet*. É ela que garante o funcionamento do serviço e permite que a plataforma trabalhe na criação do sistema a ser utilizado.

A infraestrutura pode ser alocada tanto dentro da empresa, exigindo a implementação dos *hardwares* necessários para o processo, como fora dela, em provedores terceirizados que podem estar localizados até mesmo em outros países. Como a tecnologia funciona através do uso da *internet*, o provedor fornece todo serviço remotamente, desde a infraestrutura até a aplicação final usada na empresa.

Toda essa atividade é coordenada por profissionais denominados arquitetos de infraestrutura, que organizam e dão manutenção para que o serviço funcione com qualidade e eficiência.

A segunda camada é o PaaS, também denominada Plataforma como Serviço. Essa camada é mais utilizada pelos desenvolvedores de aplicações, pois com base no IaaS eles criam soluções e recursos necessários para armazenamento, organização de

banco de dados, escalabilidade, suporte de segurança, sistemas operacionais ou novas linguagens de programação.

Pode-se dizer que os profissionais que trabalham com o PaaS criam todo o sistema que será utilizado pelo *software* para seu funcionamento. Essa união das duas primeiras camadas possibilita um acesso mais regular e estruturado à camada seguinte, o SaaS.

A última camada da computação em nuvem, e mais conhecida, é o SaaS. Isso se deve ao fato de que aplicações populares como *Gmail, Google Drive, Facebook, Internet Banking, Netflix* e tantas outras, estão alocadas nessa camada. Ela funciona como um modelo de distribuição de *software* no qual aplicações são hospedadas por um provedor de serviços e disponibilizadas através da *internet*, não sendo necessária a instalação direta em máquinas dos clientes. Além disso, o SaaS possibilita que o funcionário da empresa acesse informações e documentos do servidor corporativo remotamente desde que tenha uma conexão de *internet*.

Cada uma dessas camadas tem igual importância para o funcionamento da computação em nuvem. Quando são utilizadas em conjunto e de forma coerente, trazem uma série de benefícios à empresa como aumento da produtividade, escalabilidade, flexibilidade, entre outros.

Os termos *cloud computing* e computação em nuvens são relativamente recentes mas ao se analisar bem, é perceptível que a ideia não é, necessariamente, nova. Serviços de e-mail, como *Gmail* e *Yahoo! Mail*; discos virtuais na internet, como *Dropbox*; sites de armazenamento e compartilhamento de fotos ou vídeos, como *Flickr* e *YouTube*. Todos são exemplos de aplicações que, de certa forma, estão dentro do conceito de computação em nuvens.

Abaixo, uma breve lista de serviços que incorporam claramente o conceito de cloud computing:

- *Google Apps;*
- *Amazon;*
- *Panda Cloud Antivirus;*
- *Aprex;*

- *iCloud.*

II.4 Exemplo de nuvem entre ONS e Amazon Web Services

O Operador Nacional do Sistema Elétrico (ONS) é o órgão responsável pela coordenação e controle da operação das instalações de geração e transmissão de energia elétrica no Sistema Interligado Nacional (SIN), sob fiscalização e regulação da Agência Nacional de Energia Elétrica (Aneel).

O ONS desenvolve uma série de estudos e ações que são exercidas sobre o Sistema Interligado Nacional e seus agentes para manejar o estoque de energia de forma a garantir a segurança do suprimento contínuo em todo o País. O Operador Nacional é constituído por membros associados e membros participantes e atualmente conta com 800 funcionários com presença em quatro capitais brasileiras: Brasília, Florianópolis, Rio de Janeiro e Recife.

O desafio do Operador Nacional do Sistema Elétrico era lidar com situações inesperadas e por isso optaram pelas soluções da *Amazon Web Services* **[13]**. Existia a necessidade em planejar os estudos sobre situações desconhecidas e com isso a utilização de computadores poderosos para alcançarem tais objetivos era fundamental. Por lidarem com modelos matemáticos, a solução da nuvem da AWS fez com que o ONS pudesse explorar as máquinas virtuais, com custo provisionado e com a duração exata de cada um dos estudos elétricos.

O Operador Nacional do Sistema Elétrico avaliou durante seis meses a *Amazon Web Services* antes de tomar a decisão a respeito da parceria. Todos os recursos da AWS foram avaliados, com a adição da ferramenta open source MIT *StartCluster*, um utilitário para a criação e gerenciamento de clusters de computação distribuída hospedados no ambiente em nuvem da Amazon, levando à conclusão de que deveriam aderir aos serviços da nuvem.

O ONS lida com inúmeros modelos matemáticos e, para isso, usam uma alta capacidade computacional ou *high performance computing* (HPC) que os ajudam a resolver os estudos elétricos em um menor tempo e com um custo reduzido. Mensalmente é desenvolvido um planejamento eletro energético de longo prazo. A

matriz energética do Brasil atualmente é composta de quase 80% proveniente da água e, partindo deste princípio, há a necessidade de uma minuciosa avaliação se haverá necessidade ou não do uso das águas, pois, se ocorrer o uso demasiado destas e não chover corre-se o risco de não existir suprimento de energia o que pode gerar os *blackouts* ou cortes de carga.

Uma outra opção, é utilizar a energia contida nas usinas térmicas que é mais cara mas, se utilizarem estas usinas e a energia hidroelétrica não for usada, haverá um desperdício de suprimentos para a geração de energia. Sabendo de todas estas circunstâncias, o ONS utiliza um modelo que lhes permite realizar o planejamento de cinco anos e o quanto de energia que será utilizada.

Com a utilização dos serviços da *Amazon Web Services*, com o provisionamento do ambiente de cluster, usando as instâncias cc1.4xlarge e cc2.8xlarge da *Amazon Elastic Compute Cloud* (Amazon EC2), utilizando Linux (CentOS e Ubuntu), o ambiente para estudo destes casos possui uma capacidade elástica e uma maior precisão quando comparado ao ambiente anterior.

O *Amazon Elastic Compute Cloud* **[14]** é um serviço da web que fornece capacidade de computação redimensionável na nuvem. Ele foi projetado para facilitar a computação em nuvem na escala da web para os desenvolvedores, permitindo sua configuração com mínimo esforço, oferecendo controle completo dos recursos computacionais e reduzindo o tempo exigido para obter e inicializar novas instâncias do servidor, escalonando sua capacidade à medida que os requisitos de computação forem alterados.

Conforme diagrama da Figura II-4 é possível entender um pouco melhor as soluções da *Amazon Web Services* aplicadas no ONS. As instâncias de clusters interligadas são escalonadas à medida que são necessárias para os cálculos e fazem múltiplos e simultâneos acessos ao BD.

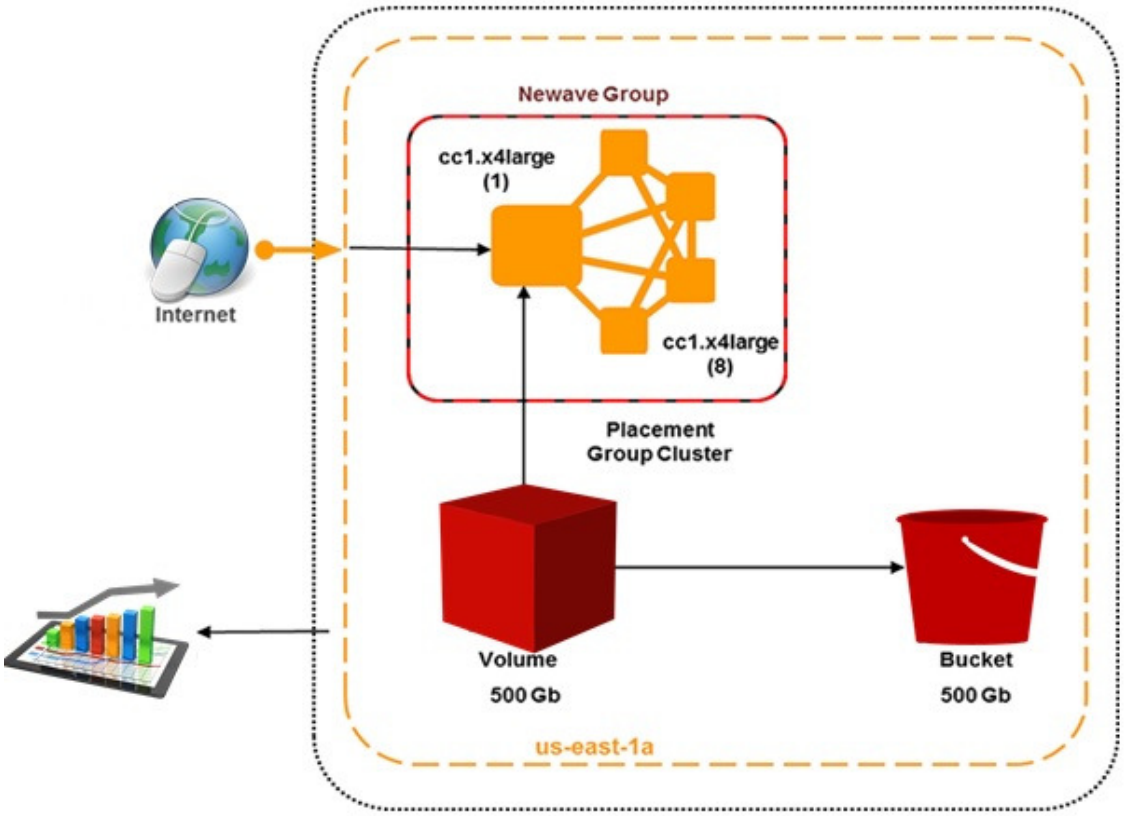


Figura II-4 - Diagrama da Amazon Web Services aplicados ao ONS

O Operador Nacional do Sistema Elétrico se beneficiou com os serviços da AWS por desenvolver seus modelos na nuvem. Hoje, com este modelo, os planejamentos são 50% mais rápidos e 30% mais baratos para serem executados.

A implementação da Nuvem proporcionou ao ONS a realização do planejamento de um ano todo, ou seja, 12 casos de situações não previstas. Tais casos se fossem desenvolvidos em um ambiente *on premise*, levariam cerca de 30 horas para serem executados e, usando a nuvem da *Amazon Web Services*, estes foram executados simultaneamente em 3 horas e meia. O resultado financeiro foi muito interessante, pois, com o uso das *Spot Instances*, permitiu-se que o custo final fosse em torno de 30 dólares para todo o trabalho realizado.

Com a utilização dos serviços não houve perda na latência e a disponibilidade da aplicação é excelente. Futuramente desejam incorporar aos serviços do ONS a *Amazon Virtual Private Cloud* (Amazon VPC), o que lhes permitirá um melhor provisionamento de uma seção da nuvem da AWS isolada logicamente executando os recursos predefinidos pelo ONS.

Como planos futuros, o ONS realizará testes com um grande volume de dados como uma solução para uma iniciativa de *Smart Grid*, focado na obtenção de dados de unidades remotas (PMU) e entendem que, para melhores resultados, existirá a necessidade de implementar o *Amazon Elastic MapReduce* (*Amazon EMR*). Também existem projetos para testes de soluções para backups.

A computação em nuvens se mostra como um forte aliado para os CIO's. A agilidade, a flexibilidade aliadas às reduções constantes de custos, podem mudar o papel de IT, tornando-o mais eficaz como parceiro de negócios e agregando valor à organização.

II.5 Comentários

Nesse capítulo foram abordados resumos dos conceitos fundamentais tanto sobre uma das partes principais de gerência do sistema elétrico como sobre computação em nuvem. Sobre o sistema elétrico foi feita uma pequena introdução ao modelo brasileiro e foi descrito o funcionamento do sistema de supervisão e aquisição de dados, parte essencial dos sistemas gerenciadores de energia. Sobre a computação em nuvem foram descritos seus três modelos e exemplos de aplicação. Ainda neste capítulo foi citado o exemplo de integração, já elaborado, entre o Operador Nacional do Sistema Elétrico (ONS) e um modelo de computação em nuvem (*Amazon Web Services*).

Capítulo III: Proposta

Esse trabalho tem como meta sugerir uma arquitetura de visualização de dados online, que forneça informações significativas, parciais e diferenciais aos responsáveis pela monitoração e possíveis utilizadores do sistema elétrico, com o intuito de possibilitar novos estudos e funcionalidades e dar uma visão intuitiva do estado do sistema.

Para implantação do modelo proposto se faz necessário o uso de ferramentas relacionadas ao gerenciamento do sistema elétrico e a computação em nuvem, ambas sendo apresentadas nesse capítulo. Posteriormente são referenciados os módulos principais da arquitetura e detalhadas suas características.

O tipo de computação em nuvem escolhido para a proposta é a PaaS, por ter um foco maior na implantação de diferentes tipos de aplicações, com repositórios pré-estabelecidos de recursos aparentemente ilimitados, eliminando a complexidade de implantação e configuração de infraestrutura e liberando o usuário a uma ampla e flexível escolha de tecnologias para construção de interfaces. As PaaS providenciam um controle completo e minucioso das formas de apresentação para desenvolvedores de acordo com alguns contextos específicos, de forma que é possível exibir os resultados independente do dispositivo utilizado. Essa plataforma facilita a rápida construção de aplicações fornecendo persistência de dados e funcionalidades de workflow, além de contribuir com a disseminação de princípios da SOA (*Service Oriented Architecture*), que permite a integração de dados e funcionalidades de aplicativos com outros sistemas e aplicações *on premise* ou *on demand*. Essas considerações levam o desenvolvedor a mantém o foco no negócio sem gastar energia com a escolha ou manutenção do sistema.

Para o gerenciamento de energia elétrica, a opção foi pelo SAGE, por ser um sistema aberto, com tecnologia de ponta para o gerenciamento de energia, por ser amplamente utilizado em território nacional e principalmente pelo fácil acesso ao sistema.

III.1 SAGE

O Sistema Aberto para Gerenciamento de Energia **[15]** - SAGE - contempla uma integração de pesquisa e desenvolvimento, que agrupa um leque de tecnologias computacionais avançadas, constituindo-se em um salto significativo na concepção de sistemas para centros de controle de energia elétrica.

Este produto, desenvolvido pelo Centro Pesquisas de Energia Elétrica - CEPEL resolve os problemas comuns aos sistemas atuais, implantados em várias empresas brasileiras, que sofrem pela dificuldade de incorporação de avanços tecnológicos e pelos custos de manutenção e expansão, principalmente devido à sua grande dependência em relação aos fornecedores originais dos sistemas.

Figura III-1 - Operador Nacional do Sistema Elétrico utilizando o SAGE

III.1.1 O Sistema SAGE

Com a experiência obtida no desenvolvimento, implantação e operação de sistemas proprietários e de suas limitações, e com a evolução das tecnologias computacionais e tendência para a padronização, observou-se a possibilidade e a necessidade de se estabelecerem novas diretrizes para o desenvolvimento de sistemas para centros de controle, empregando-se o conceito de Sistemas Abertos.

Na realidade o termo "sistema aberto", como aplicado em EMS's, não tem somente o significado de uma especificação pública, mas, acima de tudo, define uma arquitetura que contempla as seguintes características:

- Portabilidade: habilidade de implementar a mesma funcionalidade em diferentes plataformas de hardware;

- Expansibilidade: capacidade de expansão tanto em hardware (e.g. aumento de capacidade computacional ou de armazenamento) como em software (e.g. melhoramentos e implantação de novas funções de gerenciamento de energia). Também considera a habilidade de processamento em arquiteturas de diferentes capacidades;

- Modularidade: diferentes funções são implementadas por módulos de software com interfaces bem definidas, permitindo adição e remoção sem interferir em outros módulos;

- Interconectividade: habilidade de conectar diferentes plataformas de hardware através de uma rede padrão.

O sistema SAGE implementa as funções de gerenciamento de energia em centros de controle, suportado por uma arquitetura que contempla em toda a sua plenitude as características de sistemas abertos. Sua funcionalidade pode ser configurada para diversas aplicações no processo de automação das empresas, desde aplicações locais em usinas e subestações, com arquiteturas de baixo custo (PCs), até aplicações em centros de operação de grande porte suportadas por redes locais heterogêneas e *hardware* (*workstations*) de diferentes fabricantes. O *software* de aplicação pode diferir de acordo com o ambiente de aplicação, porém o suporte computacional permanece o mesmo.

A escalabilidade do SAGE fica bem demonstrada por estar instalado como sistema de supervisão local em empresas como COELBA e CHESF, e com instalação prevista, como EMS, no Centro Nacional de Supervisão dos Sistemas, da Eletrobrás.

Assim, o sistema SAGE se configura como uma solução unificada para todos os níveis de supervisão, com consequente redução dos custos de implantação e manutenção. Isto se torna de grande relevância devido ao fato que o custo total de um sistema de supervisão e controle não depende somente do investimento inicial de aquisição. Tipicamente, 70% do custo total deste tipo de sistema advém do processo evolutivo ao longo da vida útil do mesmo.

Sob o aspecto da facilidade de integração do sistema de supervisão com a rede de informação da empresa, o sistema SAGE habilita o centro de operação de uma empresa a se tornar um centro estratégico de aquisição e tratamento de informação, vital para o salto qualitativo na prestação de serviços de suprimento de energia elétrica.

O SAGE foi um sistema desenvolvido com o intuito de reduzir constantes problemas de sistemas de energia elétrica até então implantados em empresas de concessão, para que os equipamentos dos mais variados tipos e fabricantes pudessem interagir adequadamente, de forma a facilitar a incorporação de avanços tecnológicos e minimizar os custos de manutenção e expansão desses sistemas.

O SAGE, na interpretação de Moreale **[16]**, trabalha de duas formas:

- Modo SCADA (modo básico) formado por um subsistema de tratamento de informações (definição e manutenção offline da base de dados), subsistema de suporte computacional (gerência da base de dados e tarefa de controle dos processos em tempo real), aquisição e controle de dados (aquisição, tratamento e distribuição de dados e comunicação entre níveis hierárquicos diferentes) e interface gráfica para a intercomunicação. Este modo de funcionamento é o que é utilizado pelas empresas concessionárias dos sistemas de energia elétrica tanto nas subestações (SAGE/SCADA – REMOTO) como nos COS locais de hierarquia inferior de supervisão e controle (SAGE/SCADA – LOCAL), sendo este modo de funcionamento relacionado ao conteúdo deste trabalho;

- Modo EMS (modo avançado) que se constitui das funcionalidades do modo SAGE/SCADA somados às funcionalidades de análise de redes e aplicativos que possam atender a necessidades eventuais de alguns clientes. Como este modo é utilizado apenas em níveis hierárquicos superiores de supervisão e controle (COSR e CNOS), o mesmo não engloba o tema deste estudo.

III.1.2 Gerência de dados e controle distribuído

No SAGE as atividades de gerência de dados utilizam uma estruturação baseada no conceito de Modelo de Dados. Um Modelo de Dados é uma forma específica de organização de conjuntos de dados de acordo com a sua utilização. Foram conceituados dois modelos básicos: Modelo de Dados Fonte e Modelo de Dados de Aplicação.

O Modelo de Dados Fonte é orientado para as necessidades do usuário e descreve o sistema elétrico e seus componentes. É implementado pela Base Fonte, a qual apresenta como características: organização relacional, residente em disco em ambiente *offlin*e; preenchida a partir de dados fornecidos pelo usuário e acessada por meio de uma interface SQL padrão.

O Modelo de Dados de Aplicação é orientado para as necessidades dos programas de aplicação, de modo a permitir o processamento eficiente destas aplicações e acesso rápido à informação em tempo-real. É implementado pelas bases de dados Referência e *online*.

A Base Referência define o Modelo de Dados de Aplicação em ambiente *offline* e é residente em disco. Existe uma única cópia na rede, armazenada em formato padrão (XDR -"*External Data Representation*") independente da representação binária particular dos números inteiros e reais dos diferentes nós da rede (potencialmente heterogênea). Esta base é preenchida a partir da Base Fonte com o uso de uma "Lógica de Carregamento" específica definida pelo usuário.

A Base *Online* define o Modelo de Dados de Aplicação no ambiente de tempo-real, e possui as seguintes características: é residente em memória; é distribuída pela rede; cada cópia utiliza a representação binária do nó da rede a que pertence. Esta base é preenchida a partir da Base Referência pela conversão da representação padrão para o formato binário usado pelo respectivo nó.

As bases Fonte e Referência possuem catálogos, implementados por sua vez como bases relacionais com interface padrão SQL. Não se utilizam assim os catálogos internos dos gerenciadores comerciais, devido à falta de padronização neste aspecto.

Atendendo ao requisito de expansibilidade do projeto, o Sistema de Gerência de Dados pode ser implementado em plataformas UNIX e *Microsoft Windows*. O único requisito é que se disponha de um gerenciador de bases de dados relacionais com interface SQL padrão.

A Base *Online* é implementada segundo o conceito de Memória Compartilhada Distribuída (MCD). Cada MCD corresponde a um repositório de objetos (dados e métodos) que são mantidos coerentes e podem ser replicados ou não nos diversos sítios da rede. Para cumprir os requisitos de alta disponibilidade (tolerância a falhas) e tempo de acesso limitado, as MCDs podem ser distribuídas e replicadas seletivamente ao longo

dos diversos sítios do sistema. Métodos específicos do usuário podem ser adicionados às MCDs.

Arquivos de Inicialização permitem manter uma versão inicial ("fria") da base de dados on-line, armazenada em formato padronizado XDR. Em tempo-real um programa de aplicação pode salvar o conteúdo de uma MCD em arquivo em disco ou recuperar dados salvos previamente.

O acesso à base de dados *online* é feito através de uma interface fornecida por um módulo responsável pela gerência de acesso às MCDs. Os dados em uma MCD são acessados com o uso de vistas onde se define o mapeamento entre dados de uma MCD e variáveis das aplicações.

É interessante observar que os métodos que efetuam somente leitura de dados, atuam sobre a cópia local da MCD com alto desempenho por não envolver tráfego em rede. As atualizações sobre a base, no entanto, são difundidas e atingem todas as cópias das respectivas MCDs nos sítios envolvidos.

No SAGE algumas funções do suporte são implementadas de maneira a suprir a falta de um padrão amplamente aceito para sistemas operacionais distribuídos. Um módulo especial foi construído para prover funções como sinalização, escalonamento e ativação de processos locais e remotos. O subsistema de suporte também é responsável pelo *boot* e *shutdown* do sistema e pelo controle da tolerância a falhas.

A tolerância a falhas é administrada em nível de processo. Cada processo estará ativo em um sítio e será replicado em outros como reserva quente. Se um sítio falha, todos os demais em execução serão informados. O subsistema de suporte em cada sítio verificará se deve iniciar a execução de algum processo que estava ativo no sítio sob falha.

III.1.3 Base de dados

A informação no SAGE flui a partir de duas fontes específicas. A primeira é feita através da configuração do sistema, de forma off-line através do carregamento da base fonte. A segunda é proveniente da monitoração do sistema elétrico pelas aplicações do SAGE, ou seja, são informações de tempo real que variam durante o processo de supervisão e controle, de forma on-line.

O fluxo da informação do SAGE está descrito na Figura III-2 a seguir.

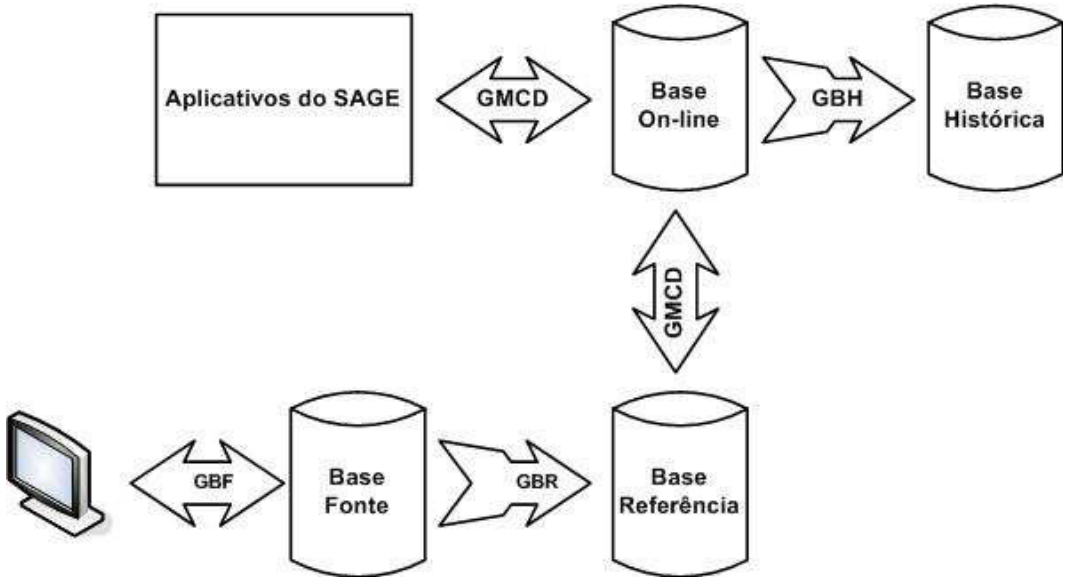


Figura III-2 - Fluxo da informação do SAGE através dos módulos

A informação no SAGE está organizada em bases de dados, cada qual com características próprias para atender a uma etapa do processo de supervisão e controle de sistemas elétricos em tempo real.

As bases de dados do SAGE são as seguintes:

- Base Fonte;
- Base Referência;
- Base Online;
- Base Histórica.

A base fonte é responsável pela configuração do SAGE. É preenchida a partir de arquivos texto (.dat) ou interface gráfica. Nela são descritas todas as características do sistema de supervisão e controle.

Suas principais características são:

- Configuração do SAGE;
- Cadastro do sistema monitorado;
- Residente em bancos de dados relacional;
- Preenchida manualmente pelo usuário.

A base referência é um estágio intermediário entre as bases fonte e on-line. É residente em disco, organizada em MCDs, Memória Compartilhada e Distribuída. É gerada durante o processo de configuração do SAGE e atualizada durante a operação em tempo real

Suas principais características são:

- Interface entre a Base Fonte e a Base On-line;
- Contêm dados cadastrais e provenientes do tempo real;
- Utilizada durante a ativação do SAGE;
- Armazenada em disco;
- Distribuída, entidades organizadas em classes de MCDs (Memória Compartilhada Distribuída);
- Base fria x Base preservada (com alterações feitas em tempo real mantidas).

Base *online* é a instância em memória da base de referência. É gerenciada pelo GMCD que inicializa a partir da base referência durante a ativação do SAGE. Através da base on-line as aplicações do SAGE trocam informações em tempo real.

Suas principais características são:

- Base de Operação do SAGE;
- Residente em memória;
- Distribuída, organizada em MCDs;
- Informação em tempo real;
- Arquitetura proprietária.

III.1.4 Base Histórica

A base histórica é descrita como um subitem do SAGE porque ela é responsável pelo armazenamento histórico sobre a operação do sistema elétrico monitorado e pode ser trabalhado aparte do SAGE, servindo como mecanismo de análise de registros do

sistema. Ela armazena tanto informações provenientes da operação do sistema monitorado, quanto provenientes da evolução deste. Suas estruturas basicamente armazenam séries temporais e são gerenciadas pelo GBH, sistema de Gerenciamento da Base Histórica. O GBH é configurável e permite que qualquer atributo da base on-line seja historiado.

Suas principais características são:

- Histórico do sistema monitorado;
- Armazena a evolução da topologia e informações de tempo real;
- Preenchida a partir da base de dados *online;*
- Residente em bancos de dados relacional ou não-relacional (Osisoft PI **[17]**).

O GBH é formado por um conjunto de aplicativos que permitem o armazenamento de informações de tempo-real previamente selecionadas pelo usuário em um banco de dados relacional ou não-relacional de forma possibilitar a recuperação de séries temporais representativas do comportamento do sistema elétrico monitorado.

O procedimento de carga da base histórica, ilustrado pela Figura III-3 é feito através de uma aplicação, gbh_xxx - onde xxx deve ser substituído pelo nome do servidor utilizado (Postgres8, PI, Oracle ou Informix). - que se conecta simultaneamente à base online do SAGE e a um banco de dados, e que grava neste banco os dados provenientes do tempo real.

Em caso de falha na conexão com o banco de dados, as informações são armazenadas localmente para posterior recuperação.

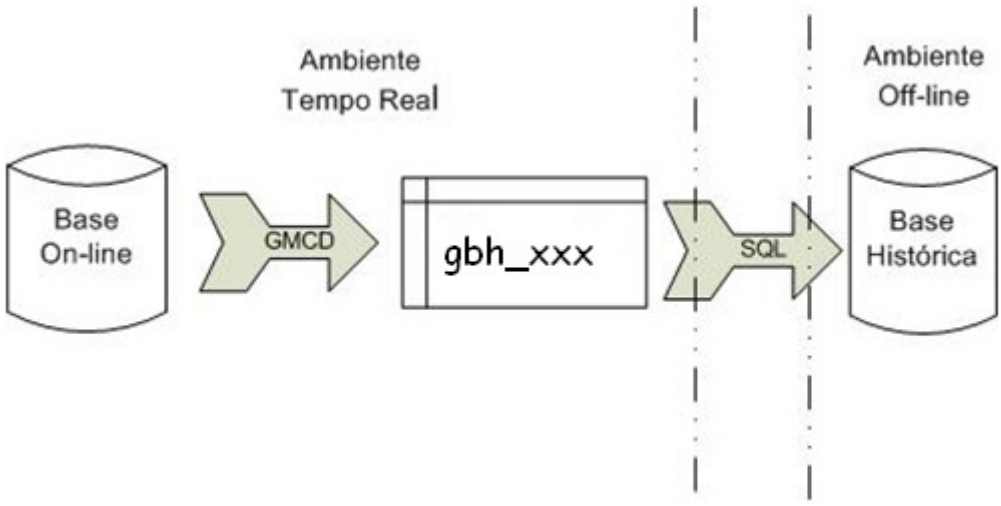


Figura III-3 - Procedimento de carga de dados da base histórica

Os recursos do sistema elétrico monitorado são representados no SAGE através de entidades que agregam atributos que descrevem as características e o comportamento em tempo real do referido recurso. Por exemplo, o recurso de ponto de medição analógica é representado no SAGE através da entidade PAS, que contém atributos que descrevem suas características com unidade da medida, equipamento associado, identificador, entre outros. Outros atributos da entidade PAS representam o comportamento em tempo real do ponto de medição, por exemplo, valor medido, indicador de qualidade da medida, e mais. Outros exemplos de recursos representados através de entidades no SAGE são: PDS, ponto digital do SAGE, EQP, Equipamento elétrico, LIA, Ligações de Aquisição.

No esquema relacional, toda tabela da base histórica é uma representação de uma entidade da base de dados *online* do SAGE, sendo que uma entidade da base *online* pode ser representada por uma ou mais tabelas na base histórica. As tabelas da base histórica podem ser classificadas em dois tipos, Referência e Dinâmicas, conforme Figura III-4.

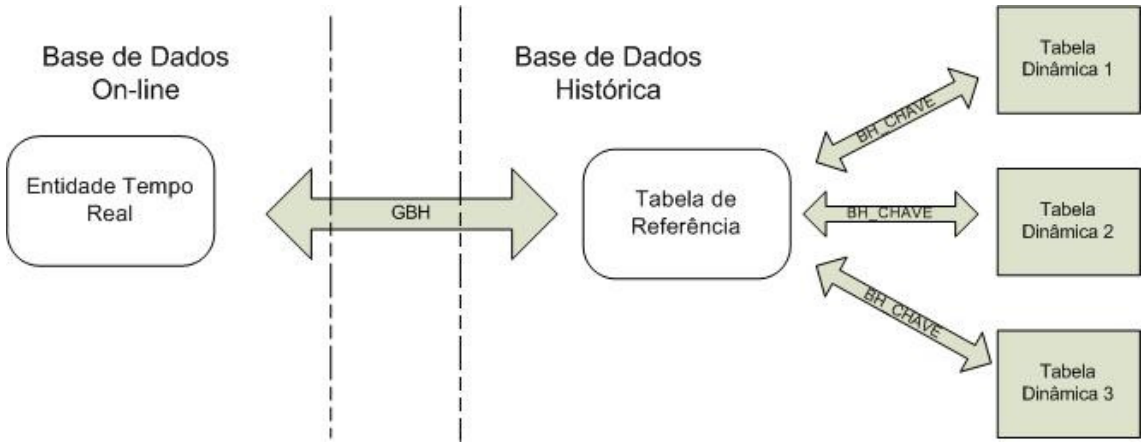


Figura III-4 - Esquema de representação das entidades do SAGE em tabelas na base histórica

As tabelas de referência têm como função manter um histórico da configuração do sistema elétrico. Elas armazenam as informações cadastrais. Seus atributos representam uma determinada característica do sistema tal como unidade medida, tipo de equipamento, identificador, etc. As entidades da base *online* só podem ter uma e somente uma tabela de referência.

- Formam um histórico da evolução do sistema monitorado
- Crescimento lento: só crescem quando é gerada uma nova configuração de base
- Contém dois atributos indicando início e fim de operação do recurso
- Armazena identificadores das tabelas dinâmicas
- São ligadas às tabelas dinâmicas através de atributos específicos

As tabelas de referência são constituídas de pelo menos seis atributos, as datas/horas de inicio e fim de operação do recurso, chave de ligação com as tabelas dinâmicas, o índice do recurso na base *online*, o identificador do recurso na base on-line, e um atributo auxiliar para manutenção de alterações nas características do recurso e rastreamento. A Tabela III-1 detalha os atributos de uma tabela de referência:

Tabela III-1 - Descrição dos atributos principais de uma tabela histórica de referência

Campo	Tipo	Descrição
BH_CHAVE	INT	Chave de ligação com a tabela de dinâmica
BH_DTHR_INI	DATA	Data/hora de início de operação do recurso.
BH_DTHR_FIM	DATA	Data/hora de fim de operação do recurso. Assume valor NULO se o recurso está em operação.
BH_INDTR	INT	Índice do recurso na base on-line do SAGE. Alterado toda vez que é gerada uma nova "base" do SAGE. Uso interno do programa gbh_xxx.
BH_SINONIMO	INT	Utilizado para indicar alterações nas características
ID	CHAR	Identificador do recurso na base on-line do SAGE.
ATR1	XXX	atributo estático um do recurso
ATR2	XXX	atributo estático dois do recurso
.	.	.
ATRN	XXX	atributo estático N do recurso

Já as tabelas dinâmicas, armazenam as informações provenientes do processamento do sistema como, por exemplo, valores medidos, estados digitais, etc. Seus atributos representam valores do sistema que variam em tempo real. Uma importante característica das tabelas dinâmicas é que elas crescem continuamente.

Principais características das tabelas dinâmicas:

- Grande volume de dados, crescimento contínuo;
- Esquemas de gravação conforme natureza dos dados;
- Particionadas em várias tabelas;
- Armazenam séries temporais
- Possuem dois tipos de registros: integridade e variação.

Uma tabela dinâmica armazena séries temporais, composta de pelo menos três atributos básicos, um identificador, uma data/hora e um ou mais valores associados. A Tabela III-2 abaixo detalha os atributos de uma tabela dinâmica:

Tabela III-2 - Descricação dos atributos principais de uma tabela histórica dinâmica

Campo	Tipo	Descrição
BH_CHAVE	INT	Chave de ligação com a tabela de referência que possui o identificador do recurso.
BH_DTHR	DATA	Data/hora da medida e/ou evento relacionado ao recurso monitorado.
BH_VARIACAO	INT	Indica se o registro foi inserido devido a uma integridade (bh_variacao=0) ou variação (bh_variacao=1). (OPCIONAL)
ATR1	XXX	Atributo dinâmico um. Valor medido ou estado digital.
ATR2	XXX	Atributo dinâmico dois. Valor medido ou estado digital.
.	.	.
ATRN	XXX	Atributo dinâmico N. Valor medido ou estado digital.

A fim de permitir a seleção de quais instâncias das entidades historiadas serão efetivamente armazenadas no histórico, foi definido um mecanismo genérico para que seja possível, durante a configuração da base fonte do SAGE, indicar se aquele recurso será ou não historiado. Por exemplo, durante a definição do histórico de pontos de medição analógica (PAS) foi determinado que o atributo HISTSLC definido na base fonte pode assumir os valores SIM e NAO o que indica se o ponto deve ser historiado ou não.

Outra característica importante é a forma com que são gravados os dados nas tabelas dinâmicas, denominados esquemas de gravação. Os esquemas de gravação podem ser classificados em dois grupos: síncrono e assíncrono. No esquema síncrono os valores do tempo real são varridos periodicamente e no esquema assíncrono a varredura se dá sempre que a base tempo-real identifica que ocorreram mudanças nos dados relacionados.

Devido ao grande volume de dados históricos o sistema possibilita a divisão em várias tabelas conforme a necessidade do usuário. A política de particionamento é feita em função do período que se deseja armazenar em cada tabela, um dia, uma semana ou um mês. No caso de não ser definida uma política, todos os dados serão armazenados em uma única tabela.

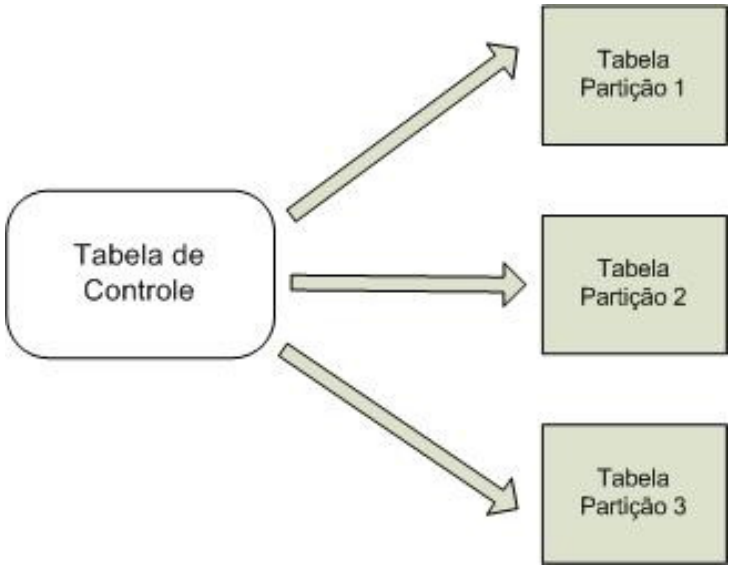


Figura III-5 - A tabela de controle e os esquema de partições

Sempre que uma partição é fechada ao fim de seu período, sua tabela é exportada para um arquivo de texto, junto com os scripts SQL necessários para sua recuperação. Toda vez que uma partição atingir a idade especificada pelo campo

HORIZONTE, definido em um de seus arquivos de configurações, a tabela é excluída da base histórica e colocada do estado OFFLINE.

Um esquema recente no SAGE implantou o conceito de herança das tabelas dinâmicas, onde cada tabela particionada é "filha" de uma tabela única e seus dados podem ser selecionados diretamente por essa tabela (contanto que as tabelas filhas ainda estejam disponíveis na base), sem precisar entender como estão dispostas as partições.

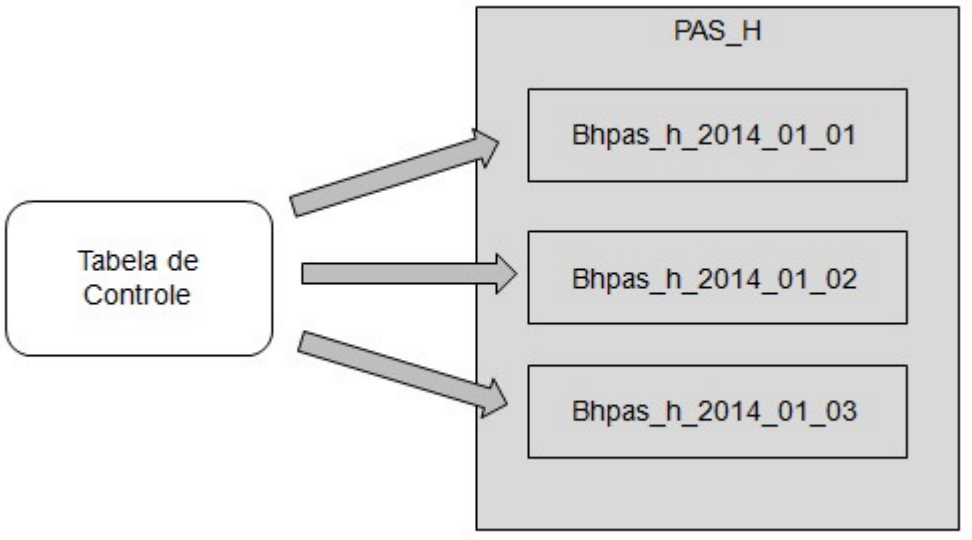


Figura III-6 - Esquema de herança com uma tabela única principal herdando das partições

Há ainda no SAGE um banco de dados não relacional, o PI da OSIsoft. Esse banco de dados utiliza as chamadas Tags para o controle de cada ponto de interesse no sistema, sendo que cada uma possui seus próprios atributos específicos nesse banco de dados. No que diz respeito ao armazenamento histórico, o sistema utiliza internamente o conceito de *Snapshot* e *Archive*, onde o primeiro representa uma "foto" do último estado das tags e o segundo representa todos os outros valores históricos já registrados.

O Snapshot é um estado único (isto é, uma cópia) de um volume de armazenamento em um determinado ponto no tempo. O *Snapshot* é basicamente uma tecnologia de *backup* de dados. Dirigida para capacidades maiores de armazenamento, a utilização dessa técnica traz vantagens. Por exemplo, um *backup* completo de um volume inteiro pode levar um longo tempo e também usar uma grande quantidade de espaço de armazenamento, mesmo para arquivos que permanecerão inalterados durante algum tempo. Além disso, ao realizar um *backup* de volumes ou subconjuntos destes em um ambiente de multiprocessamento simétrico, as operações de gravação ainda podem

continuar a modificar os dados do arquivo naquele local, evitando a atomicidade e, por sua vez, possivelmente levando a corrupção de dados. Existem maneiras de contornar isto, fazendo com que o volume permaneça *offline* ou marcado como somente leitura antes do processo de arquivamento, mas em ambientes de produção de alta disponibilidade, isto pode não ser uma opção. Este é o momento em que o *Snapshot* entra.

Usado para evitar tempo de inatividade e manter atomicidade, o *Snapshot* é uma cópia somente de leitura de um determinado volume em um ponto específico no tempo, normalmente implementado por um mecanismo de *copy-on-write*. Quando se escreve em um *Snapshot* ou no volume original (*Archive*), a gravação não será vista pelo outro. Quando um volume marcado para *Snapshot* é gravado e os dados são modificados, os blocos de dados originais e inalterados ou dados de arquivos (no caso de um *Snapshot* baseado em arquivo) serão copiados para o espaço alocado para o *Snapshot.* Depois que todos os dados originais e não modificados são copiados para o *Snapshot*, o volume original (Archive) será atualizado com os novos dados. Quando o volume do *Snapshot* precisa ser montado, usando um sistema de ponteiros, o ele fará referência ao volume principal com os dados originais salvos no *Snapshot*. Com essa técnica, torna-se possível arquivar dados valiosos de forma incremental, sem perder a produtividade ou o risco de sofrer qualquer tipo de corrupção de dados.

O histórico de dados cadastrais no BD não relacional é feito em duas partes.

Os dados estáticos das entidades historiadas com relacionamento com os dados do esquema clássico da base de pontos do PI são mapeados, configurando o atributo da base do SAGE com o mesmo nome do respectivo parâmetro na definição de uma *tag*. Por exemplo, o atributo UNIDADE de PAS que contém a unidade da medida, é configurado para ser armazenado com o nome *EngUnits* no PI, que é o parâmetro nesse servidor que indica a unidade.

Neste BD, para cada atributo configurado na tabela dinâmica será gerada uma *tag*. Os dados da tabela de referência associada à tabela dinâmica serão utilizados na definição da *tag*, e o nome do atributo especificado será utilizado como sufixo adicionado ao identificador do recurso historiado. Por exemplo, ao se configurar para serem historiados os valores estimados e considerados de uma medida analógica com o nomes “v” e “e”, então para o ponto SPEMG_440_TR9_P_AMPI_1 serão criadas duas *tags*: SPEMG_440_TR9_P_AMPI_1.v e SPEMG_440_TR9_P_AMPI_1.e

A base histórica do SAGE foi desenvolvida de forma a permitir ao usuário escolher que dados do tempo-real serão historiados, e a qualquer momento alterar a estrutura da base e os processos de gravação sem perder a compatibilidade com os dados armazenados.

Durante o processo de carga da base histórica os arquivos contendo a descrição da base histórica são lidos e criticados, e armazenados no banco de dados em estruturas específicas conhecidas como catálogo da base histórica.

Na última etapa da configuração, é gerado a partir do catálogo da base histórica o programa de carga, gbh_xxx, assim como as estruturas da base histórica. Caso já exista uma base histórica nenhuma alteração é feita nesta, sendo o programa gbh_xxx responsável por eventuais alterações necessárias na base de dados no momento de sua ativação.

A fim de facilitar a imediata implantação da base histórica o GBH já vem pré-configurado para historiar as seguintes informações:

- Medidas analógicas;
- Estados de pontos digitais;
- Estados de equipamentos elétricos;
- Comunicação de dados;
- Alarmes;
- Seqüência de eventos.

III.1.5 Visão Geral

O sistema SAGE constitui-se em uma solução unificadora para os diversos níveis hierárquicos em que se organiza a operação em tempo-real de sistemas elétricos (sistemas de supervisão de usinas e subestações, sistemas regionais e centrais, etc) permitindo tornar uniforme o processo de expansão da automação e de manutenção dos sistemas existentes.

Por suas características o SAGE habilita ainda o centro de operação de uma empresa a se tornar um centro estratégico de aquisição e tratamento de informação, vital para o salto qualitativo na prestação de serviços de suprimento de energia elétrica.

III.2 Openshift – Computação em nuvem gratuita

Na tentativa de oferecer aplicativos com mais rapidez, muitas organizações de TI enfrentam constantes desafios de produtividade. A *Red Hat* acredita que a solução para que a TI seja mais eficiente e inovadora está na nuvem com o uso da Plataforma como serviço (PaaS) pública OpenShift Online **[18]** .

O OpenShift minimiza os desafios da produtividade de TI:

- Permitindo que os desenvolvedores se concentrem no que é mais importante (código dos aplicativos) para acelerar o desenvolvimento.
- Automatizando os processos para simplificar a prestação de serviços.
- Possibilitando o uso de sua infraestrutura com mais eficiência nos ambientes de desenvolvimento, teste e produção.

O desenvolvedor pode manter o foco no desenvolvimento da aplicação sem se preocupar com a infraestrutura, e realizar suas implantações de forma bem ágil através das ferramentas disponibilizadas.

Openshift é um PaaS - plataforma como serviço - que roda sobre uma determinada infraestrutura baseada em *RedHat Enterprise Linux* (RHEL), seja sobre uma estrutura única, servidores virtualizados ou nuvens híbridas, públicas ou privadas e contém:

- Um conjunto de instância de RHEL chamadas de nós (*Nodes*) Openshift, onde as aplicações dos usuários finais vão residir. Cada nó Openshift é também um ambiente de multi-locação para as aplicações;

- *Brokers*, que gerenciam múltiplos nós e ainda servem como um mecanismo de gerenciamento interno, coordenando a sua instrumentação e sua automação. Estão ilustrados na Figura III-7.

- As engrenagens (Gears), que são recipientes seguros onde as aplicações rodam dentro de nós. Esse modelo de recipiente formado pelas engrenagens permite alta densidade com segurança e está ilustrado na Figura III-8

O Openshift utiliza o susbsistema SELINUX para fornecer uma camada pré-configurada e extra sólida de segurança ao redor das engrenagens e utiliza os grupos de controle (CGROUPS) do subsistema Linux para alocar recursos computacionais (RAM, disco, CPU) para as engrenagens.

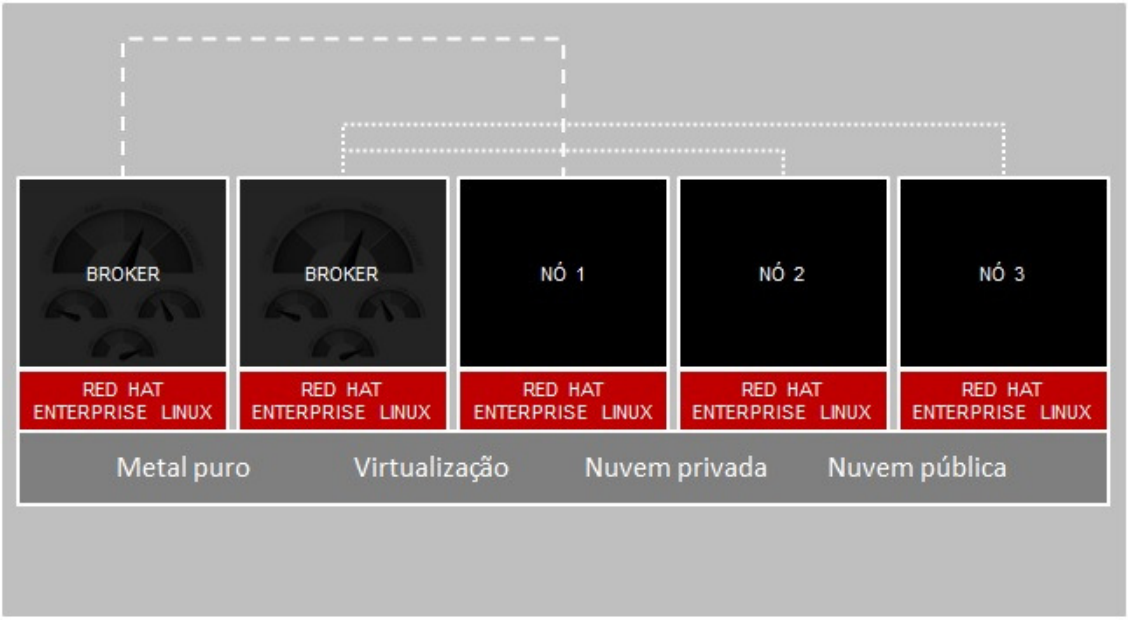


Figura III-7 - Esquema de interação entre os Brokers e os Nós Openshift

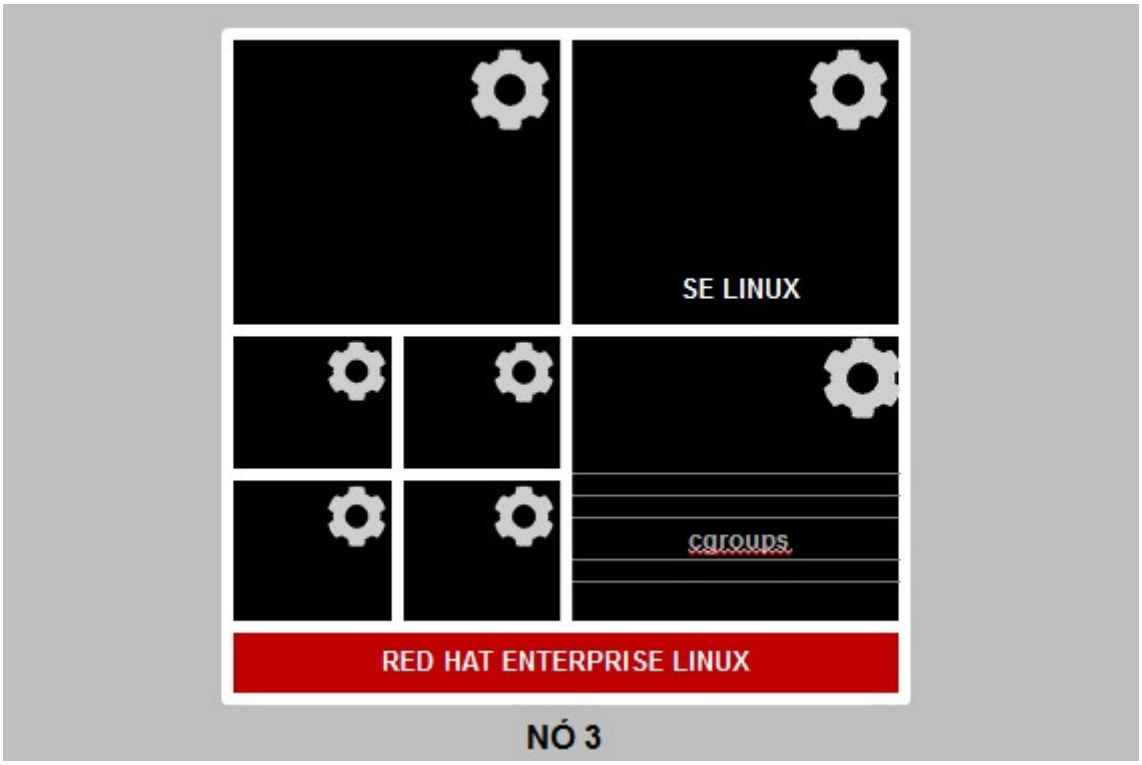


Figura III-8 - Disposição das engrenagens em um nó OpenShift

Atualmente a solução gratuita permite a cada usuário hospedar até três aplicativos gratuitamente, ou então hospedar um aplicativo e dedicar os recursos computacionais reservados para as outras duas aplicações em uma única aplicação, tudo isso em uma plataforma estável e escalável sem custos. A versão comercial da plataforma permite ao usuário a alocação de mais recursos, o apoio técnico da empresa e muitas outras funcionalidades, porém a versão gratuita continuará disponível para aqueles que não precisam de tantos recursos.

O OpenShift, tem suporte nativo a PHP, Java, Ruby, Pyhton, Perl, Node.js, Rails, Cakephp, Spring Framework, Django, Tomcat, JBoss e algumas aplicações pré-configuradas como Wordpress, Drupal, Magento, Redmine e muitas outras.

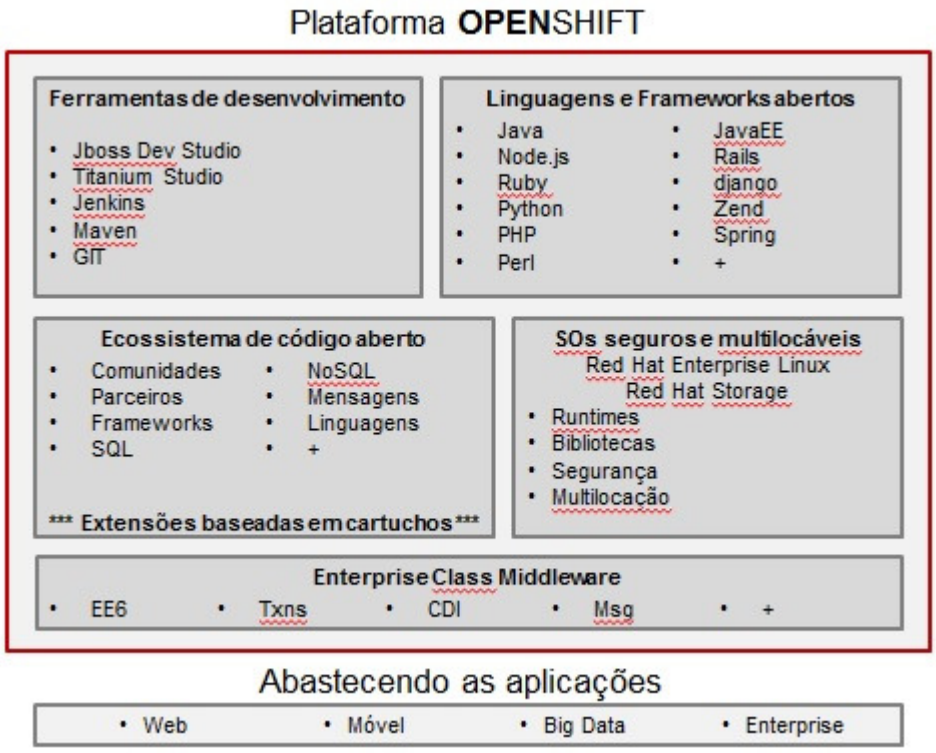


Figura III-9 - Visão geral de recursos e características da plataforma OpenShift

III.3 Arquitetura do Sistema

Devido a sua flexibilidade, os sistemas SCADA podem ser utilizados em larga variedade de processos, que incluem sua aplicação como ferramenta de análise operacional e gerencial, dependendo das condições desejadas, mas principalmente na aquisição de dados e monitoramento. Em geral, esses sistemas são usados com eficiência quando existe um grande número de variáveis a serem coletadas em intervalos relativamente pequenos (da ordem de segundos).

Uma preocupação pertinente a todo processo de evolução tecnológica é a integração com o legado tecnológico em operação no presente. Hoje, um sistema de supervisão e controle de energia elétrica do ponto de vista gerencial pode ser descrito conforme a Figura III-10 a seguir:

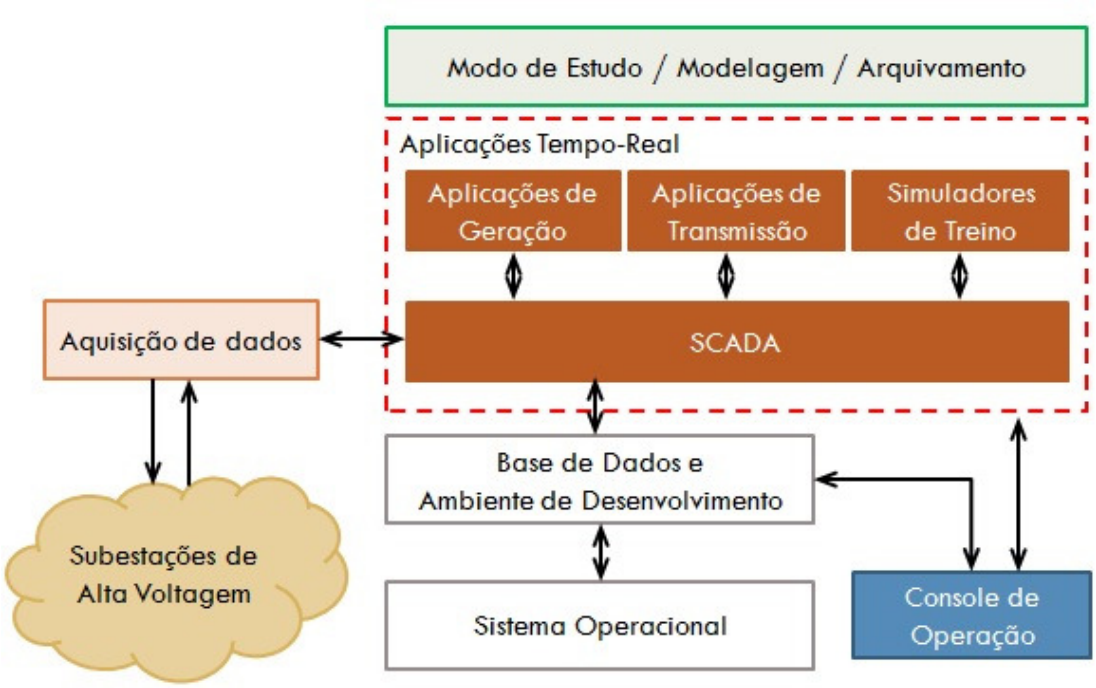


Figura III-10 - Arquitetura básica de sistemas SCADA

Cada sistema de gerenciamento possui sua própria disposição de equipamentos e estrutura de dados, com suas semelhanças e diferenças. Portanto, para uma efetiva integração com uma nova arquitetura de visualização e utilização de dados provenientes do SCADA, é necessário que esta seja modular, provendo portabilidade, e seja o mais independente possível do sistema de aquisição utilizado. Para tal feito, também se faz necessária a criação de um molde de estrutura de armazenamento de dados, que sirva tanto para comunicação entre os módulos quanto para buscar todo tipo de informação, desde bases históricas até dados do tempo real.

Baseado nessa proposta é possível definir um modelo de transmissão de dados entre um sistema de gerenciamento de energia e uma aplicação simples em nuvem da seguinte forma:

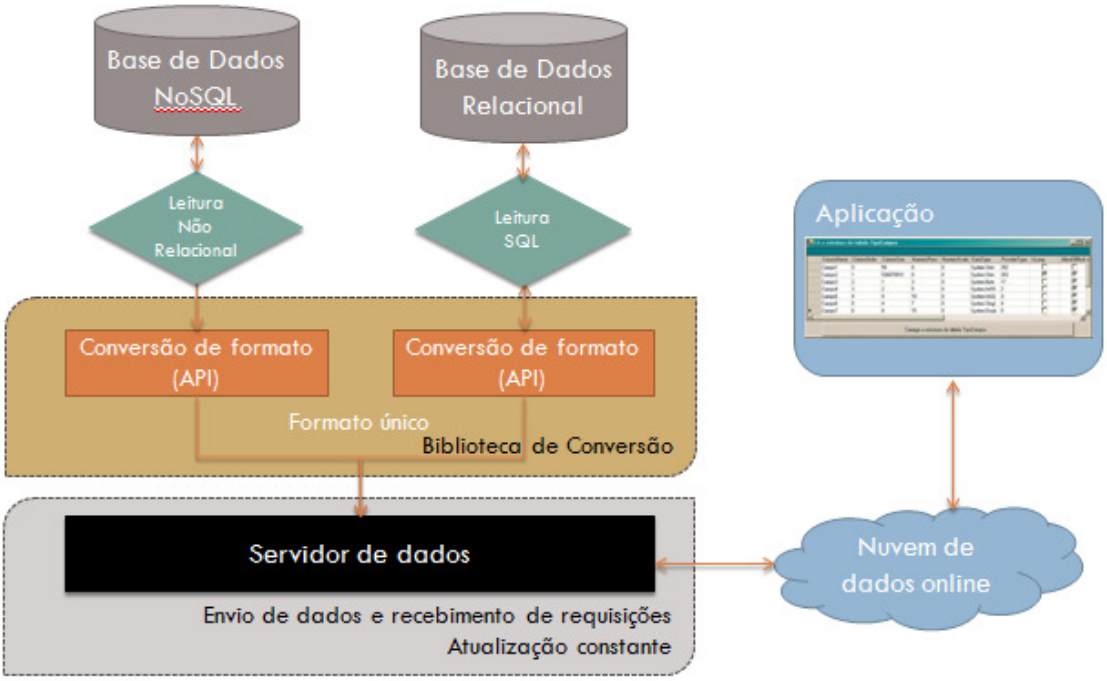


Figura III-11 - Composição dos módulos da arquitetura proposta

O modelo propõe o acoplamento de aplicações em nuvem aos sistemas SCADA por meio de consultas diretas a um servidor de dados que redireciona as pesquisas aos bancos de dados locais.

O usuário final utiliza a aplicação, localizada em uma estrutura de nuvem, para fazer requisições a um servidor de dados remoto, localizado no ambiente do sistema de gerenciamento de energia elétrica.

O servidor, a partir da posse dos dados, interpreta as mensagens de consulta recebidas e faz uma chamada a biblioteca de acesso e conversão de dados.

A biblioteca, a partir dos dados de entrada, verifica em qual das fontes se encontram as informações requisitadas e realiza a busca no sistema, retornando em seguida esses dados em uma estrutura única e independente, transformadas em um arquivo para retorno a nuvem.

O servidor então envia, dentro de um protocolo, a estrutura com o retorno da consulta para a aplicação na nuvem, que então informa ao usuário, por meio de um display, o resultado.

Essa arquitetura é funcional à medida que pode ser independente do tipo de nuvem utilizada ou do sistema de gerenciamento de energia, precisando apenas de algumas alterações em funções de determinados módulos para se adequar a novos sistemas. A Figura III-12 detalha a dependência entre os módulos.

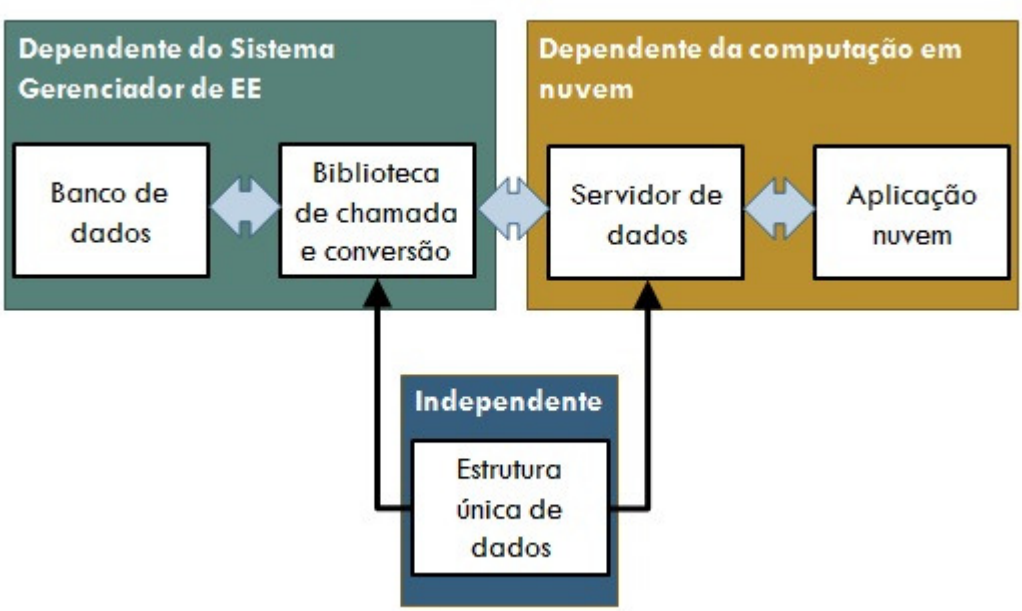


Figura III-12 - Esquema de dependência dos módulos

III.4 Estrutura

III.4.1 Bancos de dados

Na arquitetura proposta são indicados dois tipos de bancos utilizados como fonte de dados do sistema elétrico, um NoSQL **[19]** e um relacional. Cada um desses bancos tem seu esquema definido de acordo com a configuração escolhida pelos usuários do sistema de gerenciamento de energia.

O banco de dados relacional utilizado na arquitetura proposta é o PostgreSQL, um sistema gerenciador de banco de dados objeto relacional (SGBDOR), desenvolvido como projeto de código aberto e nativo do SAGE, escolhido para uso pela facilidade de interação.

Hoje, o PostgreSQL é um dos SGBDs (Sistema Gerenciador de Bancos de Dados) de código aberto mais avançados, contando com recursos como: consultas complexas, chaves estrangeiras, integridade transacional, controle de concorrência multi-versão, suporte ao modelo híbrido objeto-relacional, facilidade de Acesso, gatilhos, visões, suporte a linguagens procedurais (PL/pgSQL, PL/Python, PL/Java, PL/Perl) e indexação por texto.

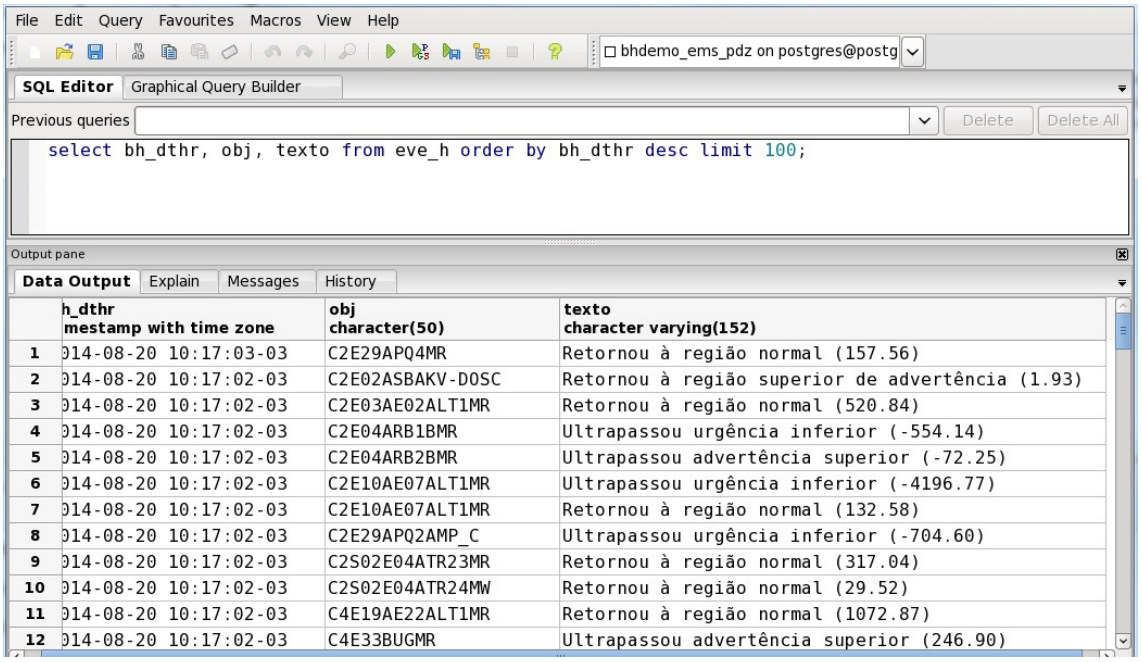
Os bancos de dados NoSQL (Not Only SQL) são banco de dados não-relacionais que rompem uma longa história de banco de dados relacionais com propriedades ACID.

Os BDs que estão sob estes rótulos não podem exigir esquemas de tabela fixa e, geralmente, não suportam instruções e operações de junção SQL. São tendências em arquiteturas de computadores, como a computação em nuvem, e a necessidade crescente de prover serviços escaláveis estão pressionando bancos de dados numa direção onde eles necessitam oferecer escalabilidade horizontal. As bases NoSQL armazenam os dados com técnicas que visam atender a esse requisito. Há alguns exemplos proeminentes de softwares de código fechado que atendem estes requisitos, sendo alguns deles Google's BigTable, IBM Cloudant e Amazon's DynamoDB e alguns exemplos de software open-source como Apache Cassandra (originalmente desenvolvido para o Facebook), Apache HBase, LinkedIn's e vários outros.

É importante entender que os NoSQL não têm intuitos de eliminar a utilização de bancos de dados relacionais, mas oferecer uma alternativa a eles, pois durante muito tempo o modelo relacional foi usado como "bala de prata" para todos os problemas de persistência.

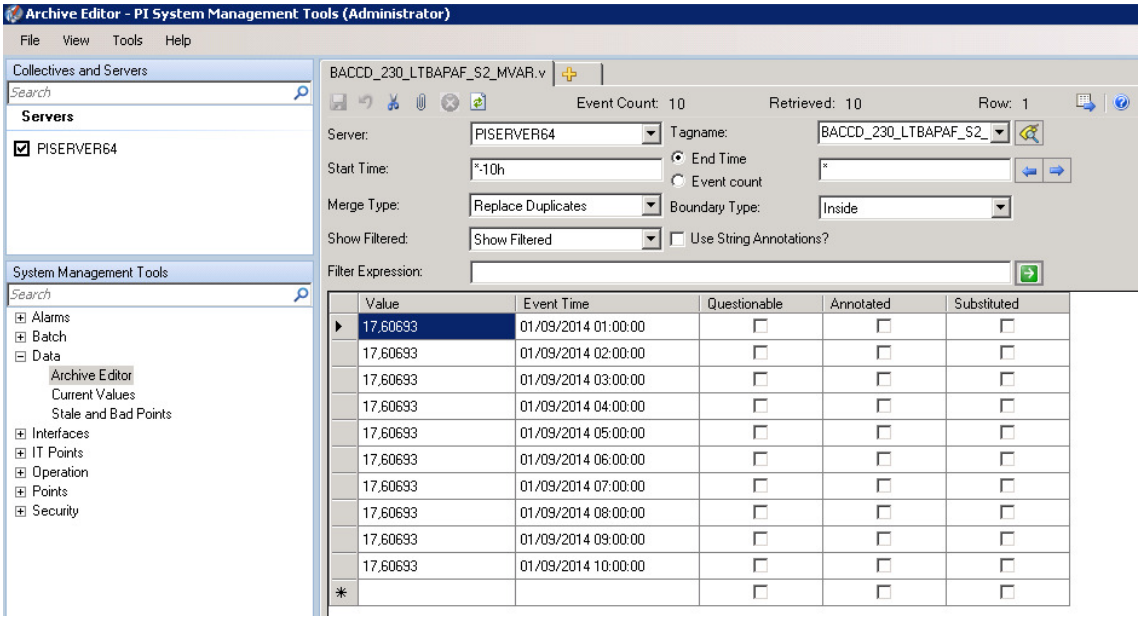
O BD NoSQL utilizado é o PI, um sistema proprietário da Osisoft que está se tornando parte do SAGE.

Ambos os bancos de dados utilizados, conforme a Figura III-13 e a Figura III-14 mostram, contêm uma base histórica configurada e ativa, conforme as figuras abaixo, e obedecem à estrutura de séries históricas temporais, onde são armazenados dados com estampa de tempo, identificadores do ponto e valores assumidos no intervalo. Há ainda a possibilidade de possuírem outras informações (atributos) relevantes a arquitetura proposta.



	h_dthr mestamp with time zone	obj character(50)	texto character varying(152)
1	014-08-20 10:17:03-03	C2E29APQ4MR	Retornou à região normal (157.56)
2	014-08-20 10:17:02-03	C2E02ASBAKV-DOSC	Retornou à região superior de advertência (1.93)
3	014-08-20 10:17:02-03	C2E03AE02ALT1MR	Retornou à região normal (520.84)
4	014-08-20 10:17:02-03	C2E04ARB1BMR	Ultrapassou urgência inferior (-554.14)
5	014-08-20 10:17:02-03	C2E04ARB2BMR	Ultrapassou advertência superior (-72.25)
6	014-08-20 10:17:02-03	C2E10AE07ALT1MR	Ultrapassou urgência inferior (-4196.77)
7	014-08-20 10:17:02-03	C2E10AE07ALT1MR	Retornou à região normal (132.58)
8	014-08-20 10:17:02-03	C2E29APQ2AMP_C	Ultrapassou urgência inferior (-704.60)
9	014-08-20 10:17:02-03	C2S02E04ATR23MR	Retornou à região normal (317.04)
10	014-08-20 10:17:02-03	C2S02E04ATR24MW	Retornou à região normal (29.52)
11	014-08-20 10:17:02-03	C4E19AE22ALT1MR	Retornou à região normal (1072.87)
12	014-08-20 10:17:02-03	C4E33BUGMR	Ultrapassou advertência superior (246.90)

Figura III-13 - Séries de dados temporais na base histórica para bancos de dados PostgreSQL



Value	Event Time	Questionable	Annotated	Substituted
17,60693	01/09/2014 01:00:00			
17,60693	01/09/2014 02:00:00			
17,60693	01/09/2014 03:00:00			
17,60693	01/09/2014 04:00:00			
17,60693	01/09/2014 05:00:00			
17,60693	01/09/2014 06:00:00			
17,60693	01/09/2014 07:00:00			
17,60693	01/09/2014 08:00:00			
17,60693	01/09/2014 09:00:00			
17,60693	01/09/2014 10:00:00			

Figura III-14 - Séries de dados temporais na base histórica para bancos de dados PI

III.4.2 RecSet

Os RecSets, ou RecordSets, são estruturas de dados em formato único já utilizadas no SAGE, independentes do sistema, residentes em memória e semelhantes aos RecordSets em Java, porém descritas em linguagem de código C ANSI. Nela estão

representados registros em uma tabela base ou os registros que resultam da execução de uma consulta. São usados para manipular dados no nível de registro. Todos os objetos do Recordset são construídos por meio de registros (linhas) e campos (colunas). Na estrutura criada é possível representar em código uma ou mais tabelas base que podem ser usadas para adicionar, alterar ou excluir registros e campos, manipular conteúdos, fazer ordenações ou buscas, gerar relatórios ou arquivos com sua estrutura, entre outras funcionalidades. Um exemplo de estrutura de RecSet pode ser visto na Figura III-15.

```
,-------------------------------------------------------------. /tmp/QueryDB-8886.tmp
|     | string             | string    | string     | string     |
| 11 | id                  | v         | dthr_exata | dthr_ceil  |
|----+--------------------+-----------+------------+------------|
|  0 | RTN_FLN_DIN_ANA059 | 6154.0498 | 1409227200 | 1409227200 |
|  1 | RTN_FLN_DIN_ANA059 | 6154.0498 | 1409223600 | 1409223600 |
|  2 | RTN_FLN_DIN_ANA059 | 6154.0498 | 1409220000 | 1409220000 |
|  3 | RTN_FLN_DIN_ANA059 | 6154.0498 | 1409216400 | 1409216400 |
|  4 | RTN_FLN_DIN_ANA059 | 6154.0498 | 1409212800 | 1409212800 |
|  5 | RTN_FLN_DIN_ANA059 | 6154.0498 | 1409209200 | 1409209200 |
|  6 | RTN_FLN_DIN_ANA059 | 6154.0498 | 1409205600 | 1409205600 |
|  7 | RTN_FLN_DIN_ANA059 | 6154.0498 | 1409202000 | 1409202000 |
|  8 | RTN_FLN_DIN_ANA059 | 6154.0498 | 1409198400 | 1409198400 |
|  9 | RTN_FLN_DIN_ANA059 | 6154.0498 | 1409194800 | 1409194800 |
| 10 | RTN_FLN_DIN_ANA059 | 6154.0498 | 1409191200 | 1409191200 |
`-------------------------------------------------------------'
```

Figura III-15 - Estrutura de um RecordSet vista como um tabular por linha de comando

A importância do RecSet vem da capacidade de modularização que ele fornece, já que ao fazer a portabilidade da arquitetura para outro sistema de gerenciamento de energia ou para outro modelo de computação em nuvem não seria necessária nenhuma alteração em sua estrutura para se adequar a um novo sistema.

As funções do RecSet e suas utilidades estão definidas no Apêndice A.

III.4.3 Biblioteca de acesso e conversão - AcessaBh

O AcessaBh é uma biblioteca auxiliar desenvolvida para este trabalho, programada em linguagem C com suporte a PL\SQL, e tem como características principais o acesso a bancos de dados relacionais e não relacionais e a conversão do retorno de consultas em uma estrutura única, os RecSets, com objetivo de facilitar a manipulação de dados e o envio/recebimento destes por parte de um servidor.

Essa biblioteca foi criada com funções específicas de acesso aos bancos de dados para que uma mesma chamada retornasse o mesmo modelo de resultado independente do banco de dados consultado.

As bases de dados históricas, no que diz respeito ao SAGE, armazenam dados de acordo com os mais variados mecanismos de gravação, podendo estes ser por:

- Integridade: a cada intervalo pré-definido acontece uma gravação de todos os pontos marcados para serem historiados;

- Variação: um flag no sistema de gerenciamento marca quando há uma alteração no ponto e na próxima varredura para gravação no histórico sua informação será armazenada. As varreduras, por padrão no SAGE, acontecem a cada 2 segundos;

- Variação externa de acordo com filtros: se as variações obedecerem a certas condições pré-estabelecidas, os pontos serão gravados;

- Integridade com variação: ocorre em ambas as situações de integridade e variação.

Os modos de gravação fazem com que as estampas de tempos possam ser bem diversificadas, podendo haver muitas ou poucas ocorrências para um mesmo ponto em um determinado intervalo.

Nos BDs relacionais, tomando pontos analógicos como exemplo, uma consulta simples de um determinado ponto em um intervalo grande tende a fornecer um número elevado de registros como resposta. Este tipo de retorno não é adequado para nosso sistema, já que pretende-se trabalhar com amostras de dados relativamente pequenas. Então, além da conversão para RecSets, também é necessário o desenvolvimento de um mecanismo para conter esse tipo de problema.

Uma maneira de limitar o resultado é dividir a resposta em intervalos pré-definidos. Para cada marca de intervalo é possível definir qual dos valores encontrados que será mantido como resposta. Então, podemos ter quatro situações:

- Manter o primeiro valor encontrado entre a marcação corrente e a marcação posterior;

- Fazer uma média dos valores entre a marcação corrente e a marcação anterior;

- Fazer uma média dos valores entre a marcação corrente e a marcação posterior;

- Manter o último valor encontrado entre a marcação corrente e a anterior.

A escolha foi por manter o último valor encontrado entre a marcação corrente e a anterior. Nos casos das médias, talvez seja interessante para pontos analógicos, mas logo ficaria inviável para pontos digitais ou estados de equipamentos. Já no caso de se manter o primeiro valor encontrado não faria tanto sentido, porque quando é feita uma busca para algum valor na base de dados, mesmo que na base de tempo real, a referência é sempre ao último valor obtido e nunca ao próximo.

Portanto, como exemplo para a situação, tem-se um ponto analógico com os seguintes valores em determinado dia:

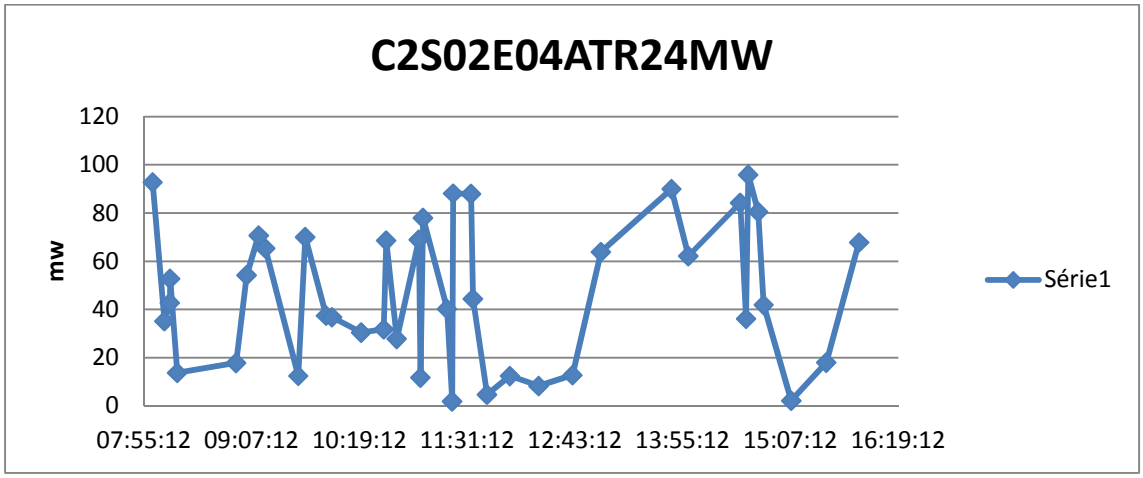


Figura III-16 - Medidas para o ponto analógico (PAS) C2S02E04ATR24MW entre 08:00 h e 16:00 h

Utilizando um período mais específico desse ponto, conforme Tabela III-3 e Figura III-17 abaixo, é possível analisar o procedimento por separação de intervalos:

Tabela III-3 - Valores do ponto C2S02E04ATR24MW entre 08:00 h e 9:00 h discriminados no tempo

Por variação	
Hora	Valor
08:00:50	92,79828
08:08:38	35,19403
08:12:20	42,8003
08:12:26	52,79033

08:17:16	13,74581
08:56:33	17,86272

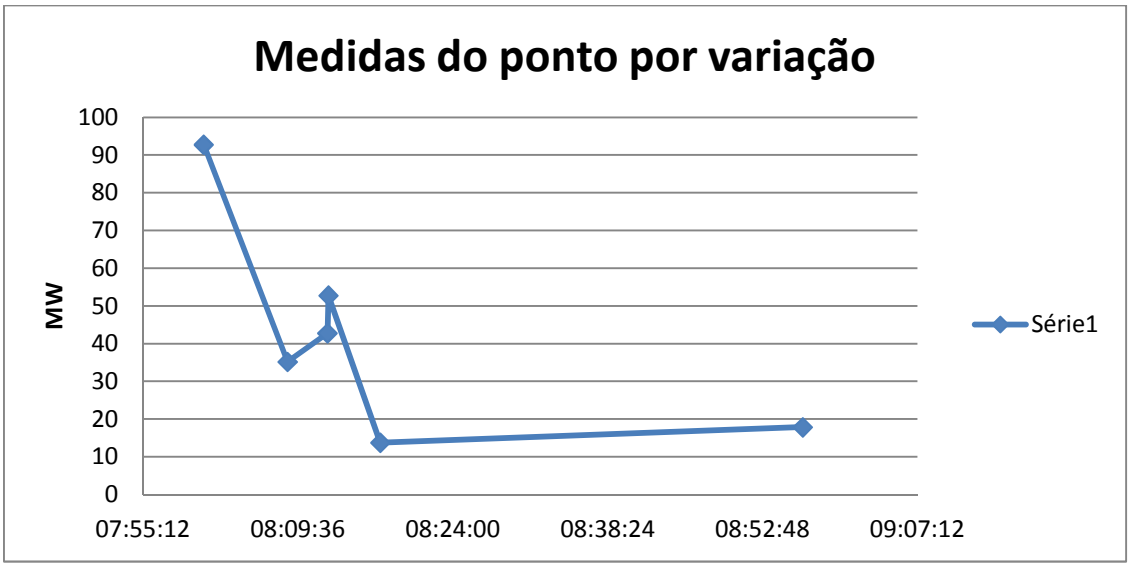


Figura III-17 - Gráfico de valores para o C2S02E04ATR24MW entre 08:00 h e 9:00 h

Aplicando-se o procedimento de último valor medido em intervalos cada vez menores:

Tabela III-4 - Medidas do ponto utilizando último valor medido a cada 20 minutos

Último valor a cada 20 min	
Hora	Valor
08:00:00	92,79828
08:10:00	35,19403
08:20:00	13,74581
08:30:00	13,74581
08:40:00	13,74581
08:50:00	13,74581
09:00:00	17,86272

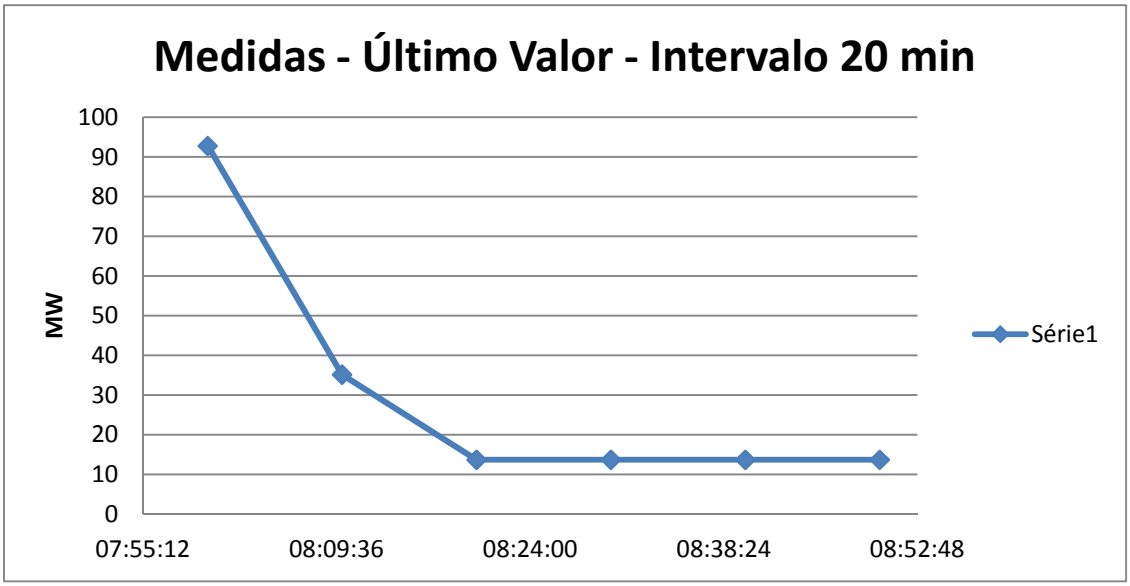


Figura III-18 - Gráfico de valores para o ponto a cada 20 minutos

Tabela III-5 - Medidas do ponto utilizando último valor medido a cada 5 minutos

Último valor a cada 5 min	
Hora	Valor
08:00:00	92,79828
08:05:00	92,79828
08:10:00	35,19403
08:15:00	52,79033
08:20:00	13,74581
08:25:00	13,74581
08:30:00	13,74581
08:35:00	13,74581
08:40:00	13,74581
08:45:00	13,74581
08:50:00	13,74581
08:55:00	13,74581
09:00:00	17,86272

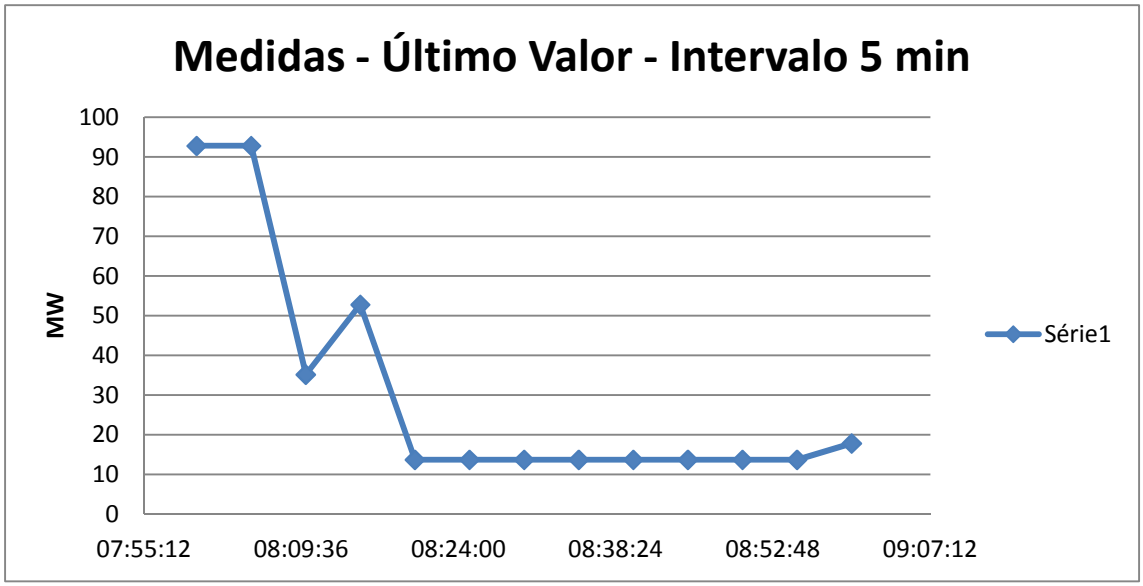


Figura III-19 - Gráfico de valores para o ponto a cada 5 minutos

É notável que a diminuição progressiva do intervalo, obtendo mais valores para as amostras, tende a apresentar um gráfico semelhante ao gráfico original do período por variação.

O usuário final, a partir desse modelo, pode inicialmente fazer buscas mais abrangentes e aos poucos ir refinando os intervalos e os períodos de acordo com as necessidades.

O BD não relacional utilizado como parte deste estudo possui uma API para consulta que serve para integração com outras ferramentas de desenvolvimento. Nas chamadas das funções dessa API existem alguns parâmetros configuráveis, dentre eles um que indica que o retorno dos dados nas consultas deve obedecer a esse critério de último valor por intervalo.

Na primeira chamada de uma função desta API para um ponto específico (uma tag no banco não relacional), o resultado da consulta para todos os intervalos é armazenado em um buffer interno e retornado a cada nova chamada. Esse procedimento é repetido para cada ponto consultado. Ou seja, a função de consulta serve apenas para uma única tag por vez, de forma que a API armazena em buffer o resultado completo da consulta e as demais chamadas para esta tag retornam em sequência os demais valores no tempo.

Internamente, no caso do PostgreSQL, foi implementada uma função no BD para retornar os últimos valores por intervalo e na lógica interna da biblioteca foi

elaborado um comando SELECT que utiliza esta função para selecionar os dados. Esta lógica pode ser encontrada no Apêndice B.

Esse modelo de busca está implementado em uma função da biblioteca e pode ser utilizado para todas as tabelas dinâmicas que possuem identificadores associados nas tabelas de referência, e, por consequência, na base de tempo real do SAGE. Porém, existem casos em que uma tabela dinâmica não possui referência direta (identificadores) e apenas armazena informações de ocorrências vindas do tempo real, exemplo este que acontece em uma tabela histórica de eventos.

As tabelas de eventos são um caso particular de dados históricos e usualmente não apresentam um número elevado de registros como as tabelas históricas referenciadas, justamente por não utilizarem múltiplos pontos como referência. É possível, então, definir apenas uma consulta simples para o período e fazer o retorno dos registros da forma como foram registrados na base de dados. Esse tipo de caso foi definido em uma segunda função da biblioteca.

No caso do BD não relacional, foi definida uma tag específica para cada ocorrência de evento.

De posse dos dados, a biblioteca aloca o conteúdo em RecSets e realiza algumas manipulações internas, como, por exemplo, o preenchimento de buracos no espaçamento entre as estampas de tempo vindas dos SELECT da primeira função. Este fato ocorre quando não existe um valor armazenado para determinado intervalo.

Ao final das atribuições e manipulações, essa estrutura é retornada ao servidor de dados.

Na Figura III-20 é possível observar o método de chamada de algumas funções definidas na biblioteca de acesso e conversão por meio de um programa de demonstração.

```
Uso:
  AcessaBhDemo_<BD> -F <1> -e <ENT> -a <ATRIBUTO1,ATRIBUTO2> -c <ID0,ID1,ID2,ID3> -i <DD/MM/Y
YYY-HH:MM:SS> -f <DD/MM/YYYY-HH:MM:SS> -I <SS>
  AcessaBhDemo_<BD> -F <2> -a <ARQ> -i <DD/MM/YYYY-HH:MM:SS> -f <DD/MM/YYYY-HH:MM:SS> -I <SS>

  AcessaBhDemo_<BD> -F <3> -e <ENT> -a <ATRIBUTO1,ATRIBUTO2> -c <ARQ> -i <DD/MM/YYYY-HH:MM:SS
> -f <DD/MM/YYYY-HH:MM:SS> -I <SS>
     Onde:
           -F =  funcao da lib AcessaBh a ser usada
            1: GetHist
            2: PegaFilme
            3: GetHistDif

            F = 1 :
           -e =  entidade associada
           -a =  atributos a serem procurados (separados por virgula ',' e SEM espacos)
           -c =  identificadores das chaves (separados por virgula ',' e SEM espacos)
           -i =  inicio do periodo no formato DD/MM/YYYY-HH:MM:SS
           -f =  fim do periodo no formato DD/MM/YYYY-HH:MM:SS
           -I =  intervalo em segundos

            F = 2 :
           -a =  Path do arquivo csv com as chaves (header no formato 'ent,atr,id')
           -i =  inicio do periodo no formato DD/MM/YYYY-HH:MM:SS
           -f =  fim do periodo no formato DD/MM/YYYY-HH:MM:SS
           -I =  intervalo em segundos

            F = 3 :
           -e =  entidade associada
           -a =  atributos a serem procurados (separados por virgula ',' e SEM espacos)
           -c =  Path do arquivo csv com os identificadores (header no formato 'id')
           -i =  inicio do periodo no formato DD/MM/YYYY-HH:MM:SS
           -f =  fim do periodo no formato DD/MM/YYYY-HH:MM:SS
           -I =  intervalo em segundos

           -h =  help
   Exemplo:
      AcessaBhDemo_pi -F 1 -e pas -a VALOR,A1_FLAGS -c SNCLPR-ANA-B4-KV,SNCLPR-ANA-B4-A -i 08
/08/2014-13:00:00 -f 08/09/2014-13:00:00 -I 43200
```

Figura III-20 - Programa de demonstração de algumas funções da biblioteca de acesso e conversão

III.4.4 Servidor de Dados

O servidor de dados, desenvolvido para este trabalho, é um processo que funciona como um aplicativo em *loop* a partir do momento da ativação, aguardando requisições da nuvem. Na chegada de uma nova requisição, no formato CSV, um novo chamado a biblioteca de conversão AcessaBH é feita e então novos dados são disponibilizados nesse mesmo formato sendo então enviados a nuvem.

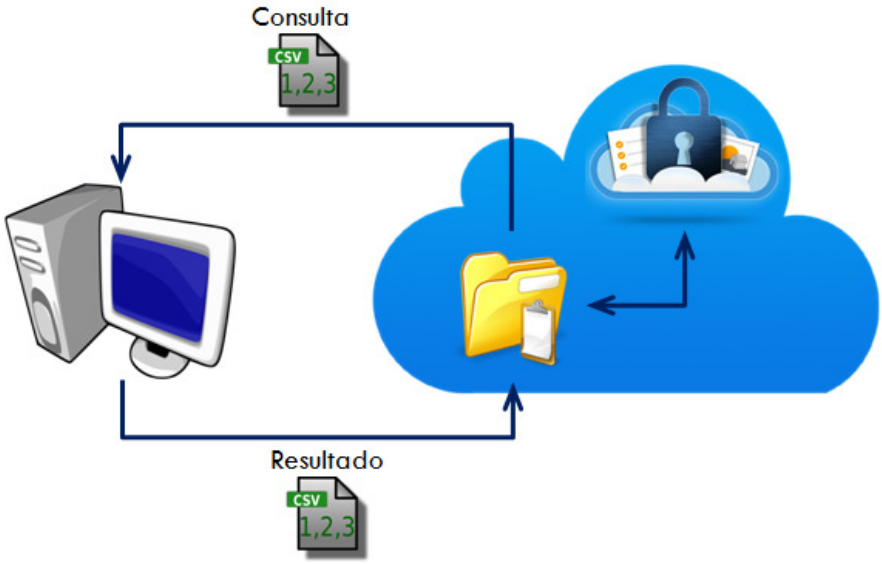


Figura III-21 - Procedimento de recebimento e envio de consultas a partir de checagem de diretório da nuvem

O servidor de dados depende parcialmente da lógica de comunicação com a nuvem. Como a comunicação é feita através de arquivos CSV's, foi necessário a implantação de um esquema de aguardo e recebimento de arquivo, sendo utilizado o esquema de temporização em segundos:

- A cada intervalo um processo de localização invoca a conexão ssh e verifica se o arquivo de consulta está disponível em um diretório específico na nuvem;
- Caso o arquivo esteja disponível, ele é transferido ao servidor;
- O servidor analisa então a consulta e invoca os outros módulos de acesso aos dados passando os parâmetros necessários;
- O resultado da consulta é recebido no formato de RecSet, transformado em arquivo CSV e então enviado também pela mesma conexão SSH (por SFTP) ao diretório de dados na nuvem.

III.4.5 Aplicativo na nuvem

Um aplicativo na nuvem Openshift pode ser criado de formas diferentes, utilizando o console web, o Eclipse IDE ou ainda por linha de comando e o próprio sistema “aloca” uma engrenagem (Gear) onde essa aplicação será executada.

Ao criar o aplicativo é possível também selecionar linguagens de programação, bancos de dados ou outros serviços disponíveis via “cartuchos” do OpenShift. Essa personalização da capacidade de um cartucho permite a inclusão de tecnologias previamente conhecidas. O sistema instala automaticamente o cartucho de linguagem Runtime em sua engrenagem original, cria uma segunda engrenagem para o banco de dados, se necessário, e conecta os dois na rede.

A plataforma usa a ferramenta GIT para o gerenciamento de código dentro da plataforma. GIT é um sistema de controle de versões distribuído, gratuito e de código aberto projetado para lidar com tudo, desde pequenos a grandes projetos com rapidez e eficiência. É fácil de aprender e tem um desempenho relativamente rápido. Supera ferramentas de SCM como Subversion, CVS, Perforce, e ClearCase com recursos como ramificação local, áreas de preparação conveniente, e múltiplos fluxos de trabalho. Basta realizar um "git push" para implantar o código na aplicação e então o código fica armazenado dentro de um repositório GIT em sua respectiva engrenagem (Gear).

O OpenShift automatiza o processo com outras ferramentas como Maven (Java), Jenkins (CI) e Apache (HTTP), e com isso ela constrói, testa e publica automaticamente. Um exemplo de disposição utilizando esses recursos pode ser visto na Figura III-22. Além dessas ferramentas, a plataforma também automatiza o dimensionamento da aplicação, à medida que usa um software balanceador de carga HA-Proxy para escalar horizontalmente conforme o aumento de carga, conforme visto na Figura III-23.

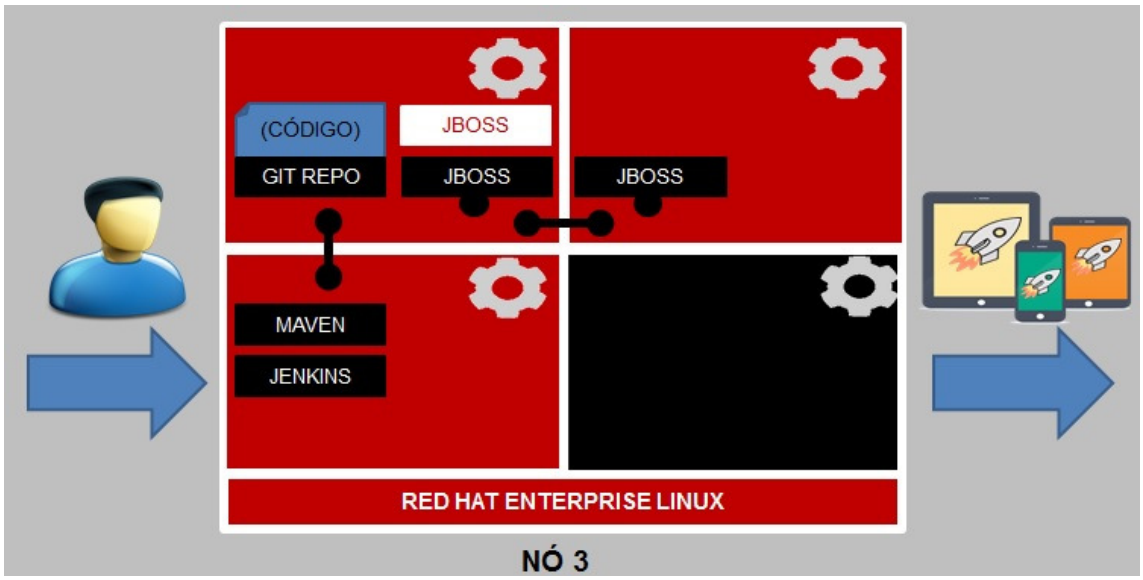


Figura III-22 - Três engrenagens do OpenShift configuradas com diferentes recursos

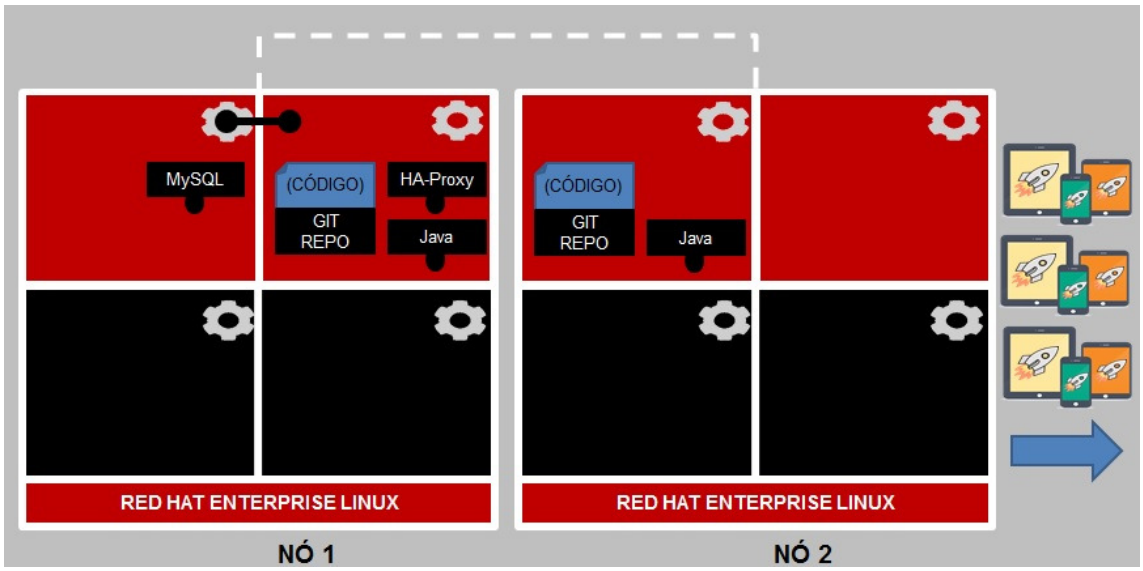


Figura III-23 - Openshift e o escalonamento das aplicações entre os nós

Seguindo o próprio passo a passo do site do Openshift, é possível criar um aplicativo simples para a visualização dos dados. A opção de linguagem escolhida foi Java pela simplicidade, facilidade de uso e leitura de arquivos CSV e principalmente pelo conhecimento prévio sobre a linguagem de programação.

Uma configuração básica foi elaborada no Openshift para integração com o servidor de dados, usando Maven e Java em um único nó, para efeitos de testes. A interface pode ser pensada de diversas formas e, em um ambiente de produção, pode ser adequada pelos desenvolvedores conforme critérios estabelecidos pelos usuários finais. No estudo, como o foco era apenas a utilização da nuvem e não como os dados vão estar dispostos pela interface, foi criada uma aplicação simples e intuitiva, com campos de

entrada de dados, conforme requeridos pela biblioteca de acesso ao BD, como entidade a ser pesquisada, atributos, identificadores do ponto, período inicial, período final e intervalo em segundos. A resposta da consulta aparece em um tabular disposto abaixo dos parâmetros de entrada.

III.4.6 Segurança na comunicação

Secure Shell (SSH) é um protocolo de rede para comunicação segura de dados, serviços de shell remoto ou execução de comandos e outros serviços seguros sobre redes entre dois computadores - um servidor e um cliente – que se conectam através de um canal seguro por uma rede insegura.

Questões de segurança e ameaças sempre foram temas abordados quando se trata de Computação em Nuvem **[20]**. O SSH é importante nesse modelo para resolver problemas de conectividade, evitando os problemas de expor uma máquina virtual baseada em nuvem diretamente na *internet*. Um túnel SSH pode fornecer um caminho seguro na Internet passando por um firewall até uma máquina virtual.

O OpenShift usa o SSH para autenticar as credenciais de conta para os servidores permitindo assim uma comunicação segura, de forma que a chave corresponde a um mecanismo de autenticaç. A autenticação bem-sucedida é necessária para gerenciar o ambiente da nuvem e a plataforma da *Red Hat* suporta ambas as chaves RSA e DSA. Não é permitida a atualização do aplicativo criado sem a definição correta da chave SSH.

Para ser autenticado com êxito pelo servidor OpenShift, a chave privada no computador local deve coincidir com a chave pública salva no servidor remoto OpenShift, ou seja, um par de chaves.

Quando a chave pública está presente em um lado e a chave privada correspondente está presente no outro lado, por padrão não existe a necessidade de digitação de senhas. No entanto, para aumentar a segurança da própria chave é possível a adição de uma senha para verificação a cada nova conexão.

Basicamente, as chaves SSH são necessárias para:

- Todas as operações GIT (incluindo o clone GIT inicial quando se cria um aplicativo);

- Visitar o OpenShift Shell do aplicativo via conexão SSH.

Esse método de autenticação e conexão remota é essencial na comunicação entre os módulos propostos.

A Chave pública, gerada localmente, pode ser adicionada a aplicação pela própria interface de acesso web. O usuário deve fazer o acesso com sua conta no site e entrar na aba de configurações, conforme demonstrado na Figura III-24.

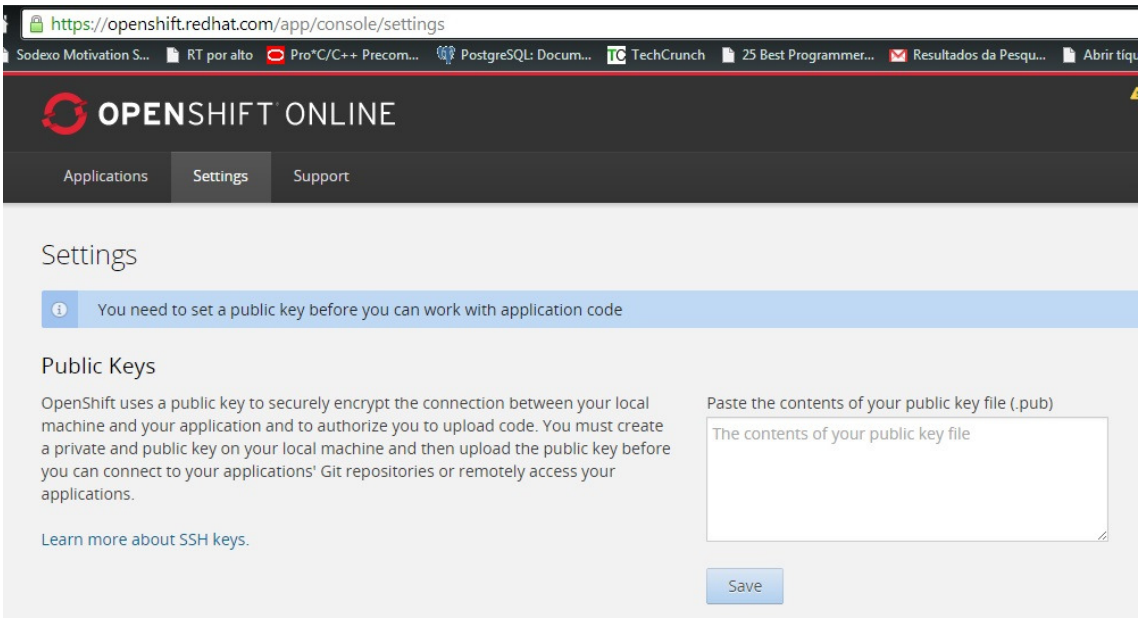


Figura III-24 - Configuração da chave pública para acesso remoto aos servidores do Openshift

Feito o procedimento, a plataforma informa ao usuário como fazer o acesso via SSH, liberando com isso a comunicação remota para as transferências de arquivo via SFTP.

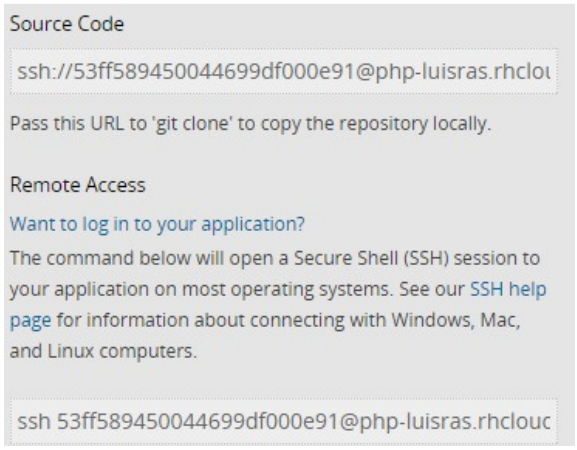


Figura III-25 - Comandos para acesso aos servidores Openshift

```
luisras@branco:~/.ssh > ssh 53ff589450044699df000e91@php-luisras.rhcloud.com

    *********************************************************************

    You are accessing a service that is for use only by authorized users.
    If you do not have authorization, discontinue use at once.
    Any use of the services is subject to the applicable terms of the
    agreement which can be found at:
    https://www.openshift.com/legal

    *********************************************************************

    Welcome to OpenShift shell

    This shell will assist you in managing OpenShift applications.

    !!! IMPORTANT !!! IMPORTANT !!! IMPORTANT !!!
    Shell access is quite powerful and it is possible for you to
    accidentally damage your application.  Proceed with care!
    If worse comes to worst, destroy your application with "rhc app delete"
    and recreate it
    !!! IMPORTANT !!! IMPORTANT !!! IMPORTANT !!!

    Type "help" for more info.


[php-luisras.rhcloud.com 53ff589450044699df000e91]\> exit
exit
Connection to php-luisras.rhcloud.com closed.
luisras@branco:~/.ssh > sftp 53ff589450044699df000e91@php-luisras.rhcloud.com
Connecting to php-luisras.rhcloud.com...
sftp> exit
```

Figura III-26 - Demonstração de autenticação remota aos servidores Openshift feita por linha de comando

III.5 Comentários

Neste capítulo foram descritas as ferramentas utilizadas para experiência, SAGE e OpenShift, respectivamente software de operação privado de um sistema elétrico e plataforma gratuita PaaS para sistemas na nuvem.

A Arquitetura, elaborada com base nessas duas ferramentas, foi descrita e composta por módulos. Cada um dos módulos, por sua vez, também foi descrito e detalhado, sendo indicados também se estes foram desenvolvidos ou apenas utilizados (já disponíveis). A lógica de comunicação e segurança entre a nuvem e o sistema de gerenciamento também foi tema deste capítulo.

Capítulo IV: Experiência e Resultados

Antes de falarmos sobre medições, é importante deixar bem claro que a medição é algo muito importante quando estamos fazendo algum tipo de análise de desempenho. É o primeiro passo que deve ser tomado, pois uma vez com dados em mãos, podemos começar a imaginar e formular hipóteses para determinar o que está acontecendo com o ambiente que estamos analisando.

A comunicação é questão fundamental neste modelo, principalmente para que não haja um atraso demasiadamente longo para o usuário final. O diagrama da Figura IV-1 exemplifica a comunicação entre os módulos do sistema:

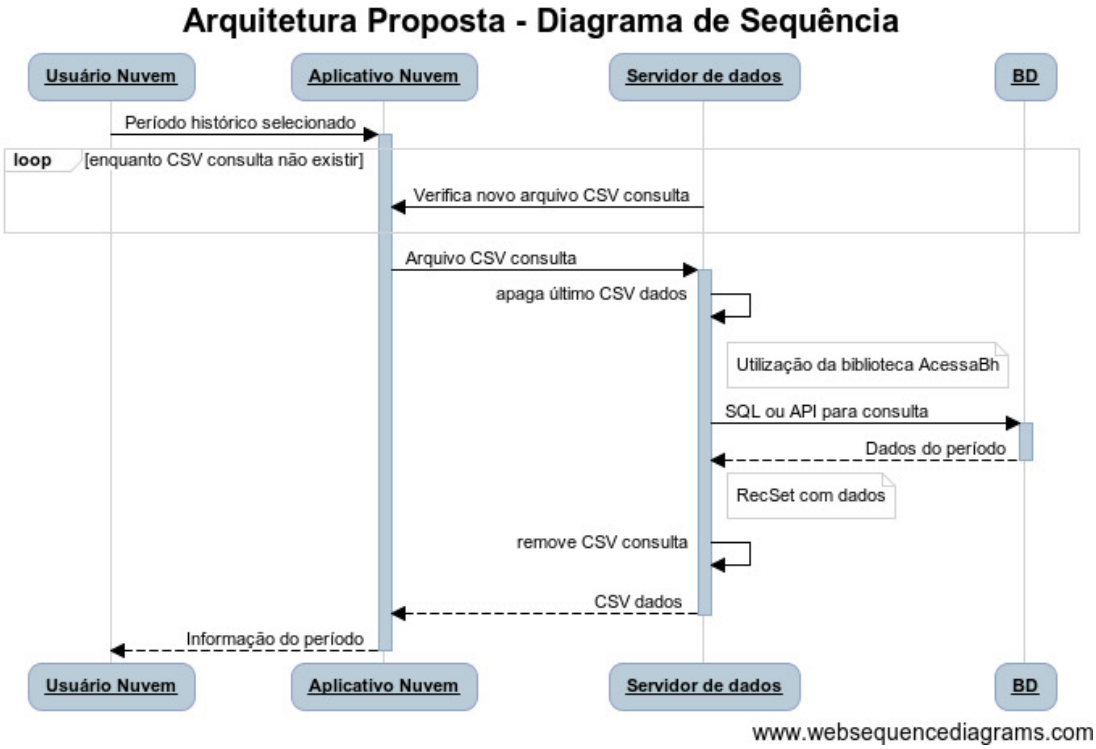


Figura IV-1 - Diagrama de sequência entre os módulos da arquitetura

Sabendo como funciona a troca de informações na arquitetura, é possível avaliar que o desempenho total do sistema pode ser medido em função do somatório do tempo de resposta de cada chamada aos módulos.

Inicialmente, tem-se:

$$T_T = T_{TS} + T_{TA}, \qquad \text{IV-1}$$

onde, T_T é o tempo médio total gasto em uma chamada pelo aplicativo na nuvem, T_{TS} é o tempo médio total gasto no servidor de dados no que diz respeito às tarefas de comunicação com a nuvem, e T_{TA} representa o tempo médio de consulta e retorno das chamadas aos BD.

Desmembrando T_{TS} e T_{TA} na equação IV-1 temos:

$$T_{TS} = T_{Rcsv} + T_{Ecsv} \quad \text{IV-2}$$

$$T_{TA} = T_C + T_M \quad \text{IV-3}$$

Onde, T_{Rcsv} é o tempo médio de detecção ou recebimento do arquivo CSV por parte do servidor de dados, T_C é o tempo médio de da consulta ao BD, recebida através do arquivo CSV, e feita pelo servidor de dados através da biblioteca AcessaBh, T_M é o tempo médio de criação e manipulação dos dados do RecSet para o retorno e T_{Ecsv} é o tempo médio total de envio e detecção do arquivo CSV com o resultado por parte da nuvem.

O usuário final realiza a consulta passando como parâmetros os dados relativos aos períodos inicial e final, a entidade e os atributos do SAGE que estão sendo pesquisados, o identificador do ponto e o intervalo de consulta para cada ponto. Deste modo, o número de registros no RecordSet de resposta (além dos identificadores e estampas de tempo associados) é dado pela fórmula:

$$N_R = [(P_F - P_I) \div I] \times N_{ATR} \times N_I \quad \text{IV-4}$$

Onde, N_R representa o número de registros, P_F e P_I representam o período final e o período inicial em segundos convertidos em inteiro, I representa o intervalo em segundos, N_{ATR} representa o número de atributos selecionados e N_C representa o número de identificadores de ponto. O número de registros é fundamental para o desempenho na comunicação porque o tamanho do arquivo de transferência está intimamente ligado a quantidade de registros no RecSet de resposta.

IV.1 Ambiente e métodos de avaliação

Para simulação do ambiente local, foi utilizada uma infraestrutura cedida pelo LASC, Laboratório Avançado de Supervisão e Controle, localizado no CEPEL, que gentilmente permitiu o uso de suas instalações para o desenvolvimento deste trabalho. A infraestrutura conta com:

- Uma máquina como servidor de dados com processador Pentium II de 2,9 GHz, memória RAM de 4GB e HD de 40GB, com SAGE instalado, sendo o sistema operacional CentOS 6.5, cliente PostgreSQL 9.3 e suporte a API do BD não relacional PI;

- Uma máquina idêntica a anterior, com SAGE instalado e ativo, sistema operacional CentOS 6.5, cliente PostgreSQL 9.3 e suporte a API do BD não relacional PI, servindo como processo de carga de dados para a base histórica;

- Um servidor de BD relacional com processador Pentium II de 2,9 GHz, memória RAM de 4GB e HD de 50GB, com sistema operacional CentOS 6.5 com PostgreSQL versão 9.3 e base histórica de dados já pré-configurada e carregada com 69.000 pontos analógicos;

- Um servidor de BD não relacional Windows Server 2008 R2 montado em uma máquina virtual, com processamento de 2,9 GHz, memória RAM de 4GB e HD de 100 GB, com PI-SDK 2010 R2 versão 1.4.0 instalado e base histórica pré-configurada e carregada com os mesmos 69.000 pontos analógicos da base relacional.

O ambiente remoto foi preparado tanto pelo site da computação em nuvem gratuita OpenShift, que cria a infraestrutura remota, como por linha de comando para a atualização dos componentes e aplicativos. A Figura IV-2 abaixo demonstra a arquitetura de comunicação criada para os testes.

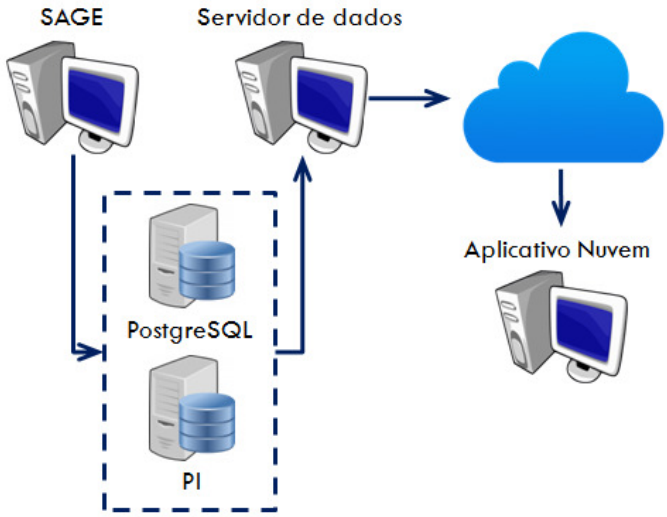


Figura IV-2 - Arquitetura de comunicação entre os módulos

Tendo definido e posto em prática os componentes da arquitetura, é possível agora avaliar o comportamento dos mecanismos a ela inerentes.

Os testes foram separados em duas baterias. A primeira bateria de testes pretende encontrar o tempo de resposta a consultas relativas a períodos variáveis para um único ponto, enquanto a segunda bateria analisa o tempo de resposta a consultas relativas a múltiplos pontos em curtos períodos, ou seja, a primeira parcela dos testes busca analisar a arquitetura do ponto de vista temporal e a segunda analisa pelo aspecto espacial. Esses tipos de consultas são muito usados pelos centros de operação em tempo-real, tanto internamente de forma automatizada nos códigos como manualmente por pedidos do usuário. Esses testes visam simular possíveis situações de consulta por parte de usuários que venham a utilizar essa arquitetura ou alguma semelhante.

Todos os testes no servidor de dados foram feitos localmente na máquina e nenhuma aplicação, além do servidor, estava sendo executada.

Cada etapa de medição foi executada cinco vezes para diminuir a possibilidade de erros devido a distúrbios da rede e o tempo médio de execução foi obtido através da média aritmética dos tempos de execução com exceção das maiores e menores medidas.

IV.2 Testes de aspecto temporal

Para a primeira bateria, foram elaborados três tipos de consulta, cada uma delas com período abrangente de 12 horas de amostras de pontos analógicos (PAS), e para cada amostra de dados foram utilizados intervalos de 1 minuto, 5 minutos e 20 minutos, variando entre 1, 2 e 5 atributos. O teste de aspecto temporal utiliza apenas um identificador de ponto, portanto $N_I = 1$ para a Equação IV-4.

O número de registros em cada caso é dado pela Tabela IV-1:

Tabela IV-1 - Número de registros para cada cada teste unitário para um único ponto

Teste	$P_F - P_I$ (seg)	I (seg)	N_{ATR}	N_R
1	43200	1200	1	36
2	43200	1200	2	72
3	43200	1200	5	180
4	43200	300	1	144
5	43200	300	2	288
6	43200	300	5	720
7	43200	60	1	720
8	43200	60	2	1440
9	43200	60	5	3600

Medindo o T_{Rcsv} para cada teste unitário acima, verificou-se:

Tabela IV-2 - Tempo de recebimento do arquivo CSV por parte do servidor

N_R	Teste	T_{Rcsv} (seg)
36	1	0,436
72	2	0,439
180	3	0,519
144	4	0,424
288	5	0,434
720	6	0,401
720	7	0,453
1440	8	0,454
3600	9	0,404

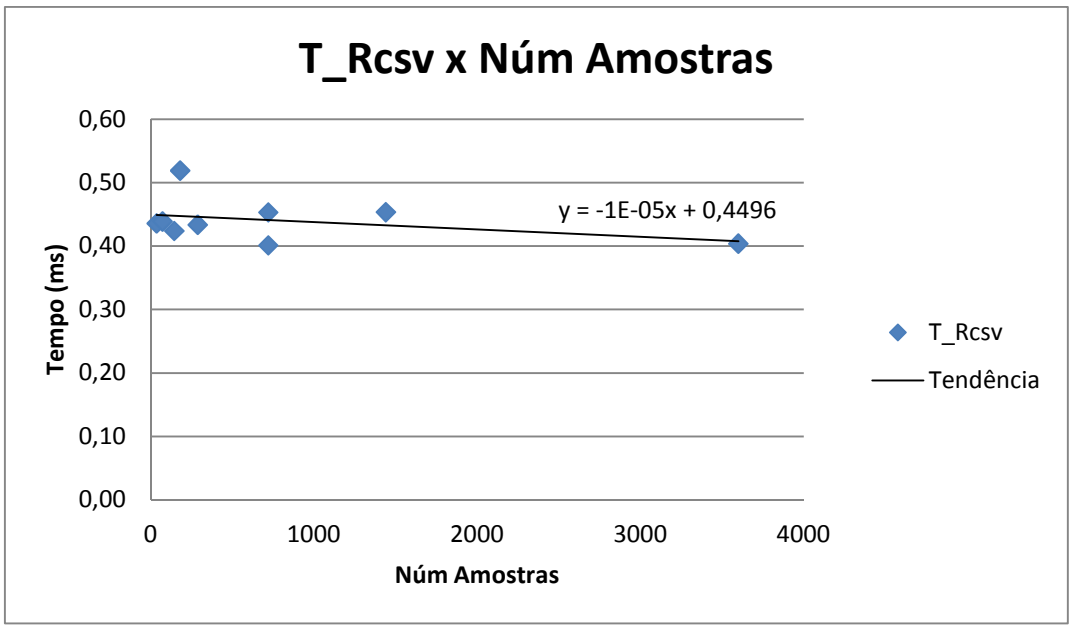


Figura IV-3 - Gráfico dos valores T_{Rcsv} no tempo

É possível observar pelo gráfico da Figura IV-3 que os atrasos totais medidos foram bem semelhantes entre si, principalmente devido ao fato do formato do arquivo de consulta gerado em CSV ser idêntico em todos os casos, alterando somente os parâmetros internos.

O atraso de cerca de meio segundo, em média, para cada caso se deve a rede de comunicação entre a nuvem OpenShift e o SAGE, que apresenta uma pequena latência na transferência de arquivos.

Para T_C e T_M em IV-3 tem-se dois cenários distintos:

- Utilizando-se BD relacional, no caso PostgreSQL
- Utilizando-se BD não relacional, no caso OsisoftPI

Ao se falar de banco de dados, várias medições são importantes. Este teste tem como foco a medição de tempo de execução das instruções SQL ou das chamadas de API que foram enviadas ao servidor para a formação do RecSet. Apesar de existirem várias maneiras pelas quais é possível de se realizar esse procedimento, e cada qual possa gerar dados que nem sempre são os mesmos, a preocupação é somente relativa ao tempo de retardo do resultado a partir da chamada das funções de requisição.

Conforme detalhado anteriormente, a biblioteca de acesso ao banco e conversão de dados AcessaBh utiliza internamente um SELECT integrado com uma função interna do BD para selecionar os últimos valores para determinado intervalo. O T_C leva em consideração o tempo desta consulta.

No caso do BD não relacional, o T_C leva em consideração a soma dos tempos de todas as chamadas a API relativas a uma mesma consulta.

O T_M, para cada caso, é variável porque a manipulação baseia-se na forma como a API ou a consulta é realizada, obedecendo a uma lógica interna para construção do RecSet de retorno.

Portanto, tem-se:

- Para BD relacional

Tabela IV-3 - Tempo total de chamada a biblioteca de acesso em função de T_C e T_M para BD relacional

N_R	Teste	T_C (seg)	T_M(seg)	T_{TA} (seg)
36	1	0,770	0,314	1,084
72	2	0,780	0,316	1,096
180	3	0,800	0,329	1,129
144	4	0,780	0,344	1,124
288	5	0,810	0,337	1,147
720	6	0,830	0,359	1,189
720	7	0,820	0,331	1,151
1440	8	0,870	0,333	1,203
3600	9	0,900	0,336	1,236

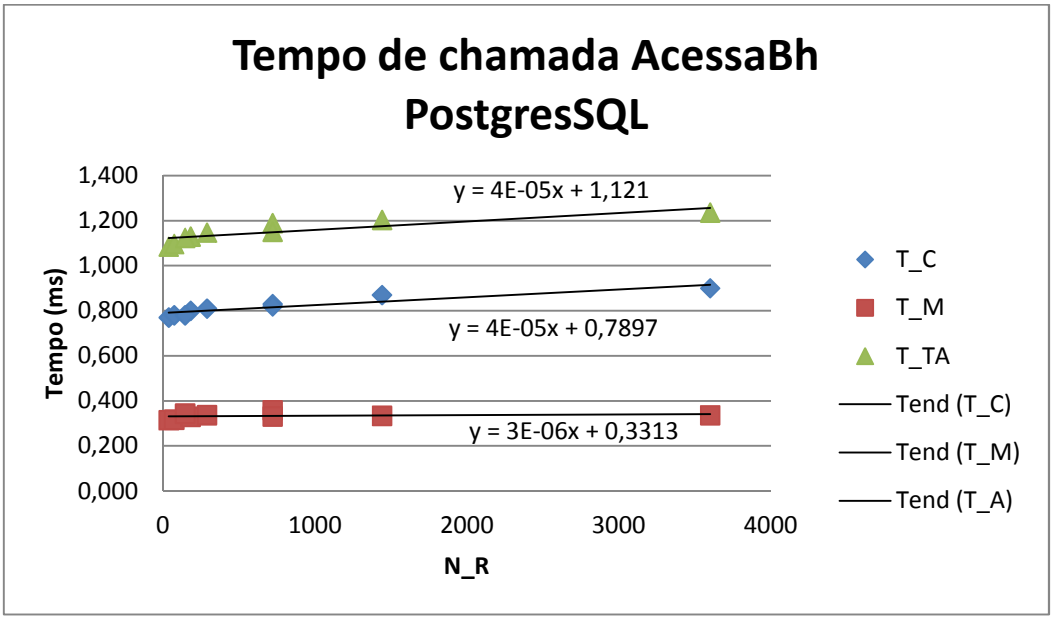


Figura IV-4 - Gráfico de área dos tempos de manipulação e consulta ao BD relacional

- Para BD não relacional

Tabela IV-4 - Tempo total de chamada a biblioteca de acesso em função de T_C e T_M para BD não relacional

N_R	Teste	T_C (seg)	T_M(seg)	T_{TA} (seg)
36	1	0,003	0,920	0,923
72	2	0,003	0,970	0,973
180	3	0,004	1,050	1,054
144	4	0,004	0,913	0,917
288	5	0,006	0,840	0,846
720	6	0,007	0,866	0,873
720	7	0,007	0,920	0,927
1440	8	0,017	0,859	0,876
3600	9	0,011	0,927	0,938

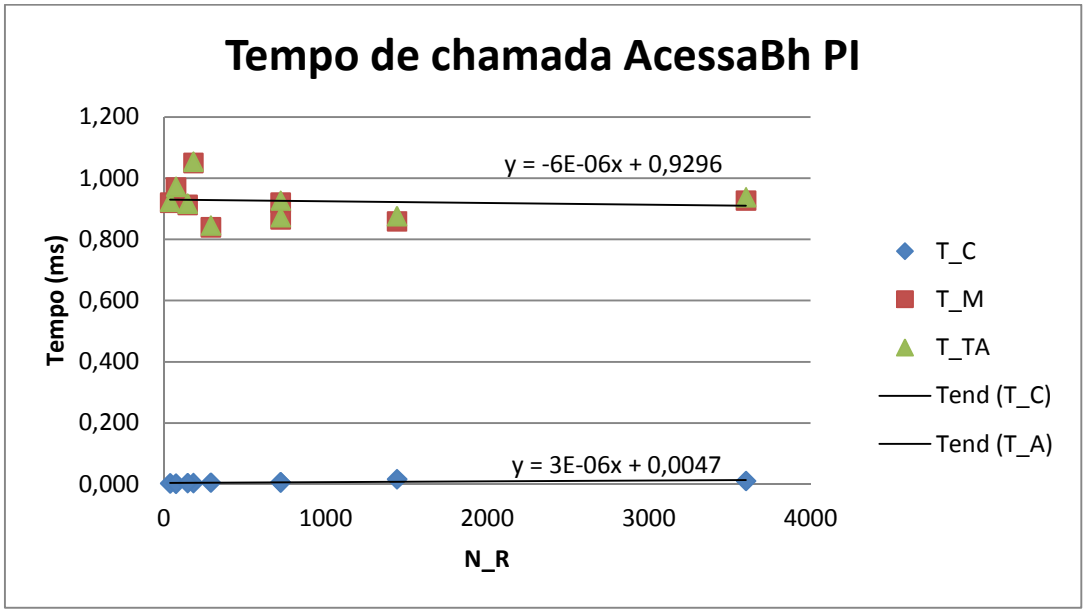


Figura IV-5 - Gráfico de área dos tempos de manipulação e consulta ao BD não relacional

Por mais que sejam feitos inúmeros testes em relação à T_C, este sempre irá sofrer variação, devido a vários fatores do próprio BD, como utilização do servidor, carga, e plano de execução. Fatores externos ao banco de dados também vão influenciar no tempo de execução. Por isso é muito difícil notar dois tempos de execução iguais mesmo que para uma mesma instrução SQL ou chamada a API, mas ainda assim é possível ter uma ideia do tempo de consulta de cada teste por meio das simulações, onde a média dos tempos indica como seria o funcionamento em um ambiente de produção.

Ambos os bancos apresentaram atrasos satisfatórios, como observado pelos gráficos das figuras Figura IV-4 e Figura IV-5, com uma superioridade bem observada

na Tabela IV-4 para o banco não relacional em termos de consulta. É perceptível também que no nível de manipulação dos dados via código, o tempo é mais elevado para os bancos não relacionais, isto por conta da utilização da API, que implica em uma lógica de codificação específica e que inclui várias chamadas e retornos às funções. No caso de bancos relacionais, as estruturas são mais genéricas e fáceis de trabalhar, por isso a manipulação é quase nula se comparado ao tempo de consulta em ambos os casos.

Outro aspecto notado pelas tabelas indica que conforme o número de amostras por intervalo aumenta, gerando um número maior de registros para apenas um ponto, o banco não-relacional tende a fornecer tempos de resposta mais rápidos.

Dando prosseguimento aos testes, o tempo médio total de envio e detecção do arquivo CSV por parte da nuvem, T_{Ecsv}, depende amplamente da rede de comunicação que a própria gerência pela conexão SSH. Obviamente, como o assunto é transferência de arquivos, o tempo total leva em consideração o tamanho do arquivo CSV gerado, e por isso é importante analisar o T_{Ecsv} também levando em conta esse critério.

No teste, dadas as condições anteriores, foi observado que:

Tabela IV-5 - Valores para T_{Ecsv} em função dos testes e do tamanho do arquivo CSV formado para envio

Teste	T_{Ecsv} (em seg)	Tamanho CSV (em KB)
1	0,94	4
2	0,96	4
3	0,97	5
4	1,04	12
5	1,05	14
6	1,20	21
7	1,35	56
8	1,65	64
9	1,78	89

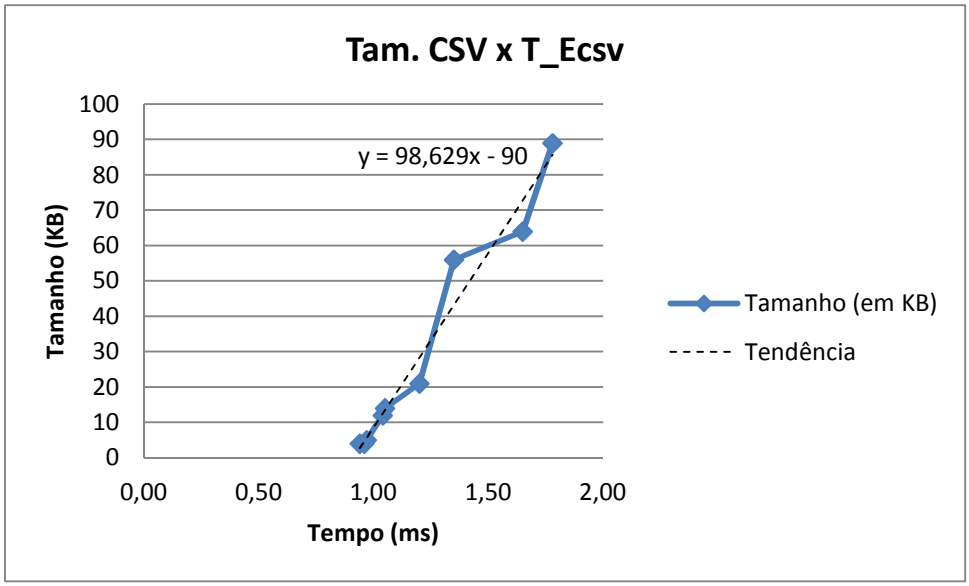


Figura IV-6 - Gráfico do tempo de envio dos arquivos CSV em função do tamanho do arquivo

O tempo de transferência total para o maior arquivo não chegou a alcançar um atraso longo, levando no pior caso testado cerca de 2 segundos de acordo com o gráfico da Figura IV-6.

Colocando-se os tempos medidos na Equação IV-1 desmembrada em IV-2 e IV-3, é obtido o atraso total de uma consulta:

- BD relacional

Tabela IV-6 - Tempo total da consulta para um único identificador com intervalos e períodos distintos e BD relacional

T_{Rcsv} (seg)	T_C (seg)	T_M (seg)	T_{Ecsv} (em seg)	T_T (em seg)
0,436	0,770	0,314	0,940	2,460
0,439	0,780	0,316	0,960	2,495
0,519	0,800	0,329	0,970	2,618
0,424	0,780	0,344	1,040	2,587
0,434	0,810	0,337	1,050	2,631
0,401	0,830	0,359	1,200	2,790
0,453	0,820	0,331	1,350	2,954
0,454	0,870	0,333	1,650	3,307
0,404	0,900	0,336	1,780	3,420

- BD não relacional

Tabela IV-7 - Tempo total da consulta para um único identificador com intervalos e períodos distintos e BD não relacional

T_{Rcsv} (seg)	T_C (seg)	T_M (seg)	T_{Ecsv} (em seg)	T_T (em seg)
0,436	0,003	0,920	0,940	2,299
0,439	0,003	0,970	0,960	2,371
0,519	0,004	1,050	0,970	2,543
0,424	0,004	0,913	1,040	2,381
0,434	0,006	0,840	1,050	2,329
0,401	0,007	0,866	1,200	2,474
0,453	0,007	0,920	1,350	2,730
0,454	0,017	0,859	1,650	2,980
0,404	0,011	0,927	1,780	3,122

No aspecto temporal de consulta, os testes apresentaram um desempenho ótimo para ambos os bancos de dados, conforme observado nas tabelas Tabela IV-6 e

Tabela IV-7, com atrasos nos piores casos de cerca de 3 segundos para 3600 registros sendo enviados. Outra observação importante é notada em buscas que privilegiam intervalos mais curtos, onde a tendência aponta para uma resposta melhor em bancos de dados não relacionais nesta arquitetura.

IV.3 Testes de aspecto espacial

Na segunda bateria de testes, foram elaborados 3 tipos de consulta, cada uma delas com 10, 25 e 100 identificadores de PAS, em um período de 1 hora e intervalo de 5 minutos, sendo pesquisados 1, 2 e 5 atributos em cada uma das amostras.

Em relação à fórmula aplicada anteriormente, a diferença no aspecto espacial acontece em todas as variáveis, já que agora a inclusão de múltiplos pontos implica:

- Na alteração do tamanho do arquivo CSV de envio ao servidor, englobando agora mais identificadores pontos;
- Na alteração do tempo de consulta ao BD e manipulação interna do retorno pela biblioteca AcessaBh;
- E na modificação do tamanho do arquivo CSV de retorno à nuvem.

Portanto, aplicando a equação IV-4 para obter o número total de registros para cada teste, tem-se:

Tabela IV-8 - Número de registros para cada cada teste unitário para múltiplos pontos

Teste	$P_F - P_I$ (seg)	I (seg)	N_{ATR}	N_I	N_R
1	3600	60	1	10	600
2	3600	60	2	10	1200
3	3600	60	5	10	3000
4	3600	60	1	25	1500
5	3600	60	2	25	3000
6	3600	60	5	25	7500
7	3600	60	1	100	6000
8	3600	60	2	100	12000
9	3600	60	5	100	30000

Medindo o T_{Rcsv} para cada teste unitário acima, verificou-se:

Tabela IV-9 - Tempo de recebimento do arquivo CSV por parte do servidor

N_R	Teste	T_{Rcsv} (seg)
600	1	0,446
1200	2	0,469
3000	3	0,474
1500	4	0,474
3000	5	0,474
7500	6	0,452
6000	7	0,463
12000	8	0,494
30000	9	0,459

Ao comparar os dados obtidos no aspecto temporal com os do aspecto espacial, dados pelas tabelas Tabela IV-2 e Tabela IV-9 respectivamente, há uma diferença muito pequena, provavelmente devida há uma ligeira variação no conteúdo do arquivo ou até mesmo a aspectos da rede.

Medindo os tempos em T_{TA} dado pela equação IV-3:

- Para BD relacional

Tabela IV-10 - Tempo total de chamada a biblioteca de acesso em função de T_C e T_M para BD relacional

N_R	Teste	T_C (em seg)	T_M (em seg)	T_{TA}(em seg)
600	1	0,430	0,164	0,594
1200	2	0,470	0,183	0,653
3000	3	0,500	0,197	0,697
1500	4	0,730	0,214	0,944
3000	5	0,760	0,230	0,990
7500	6	0,780	0,246	1,026
6000	7	1,340	0,262	1,602
12000	8	1,370	0,278	1,648
30000	9	1,380	0,294	1,674

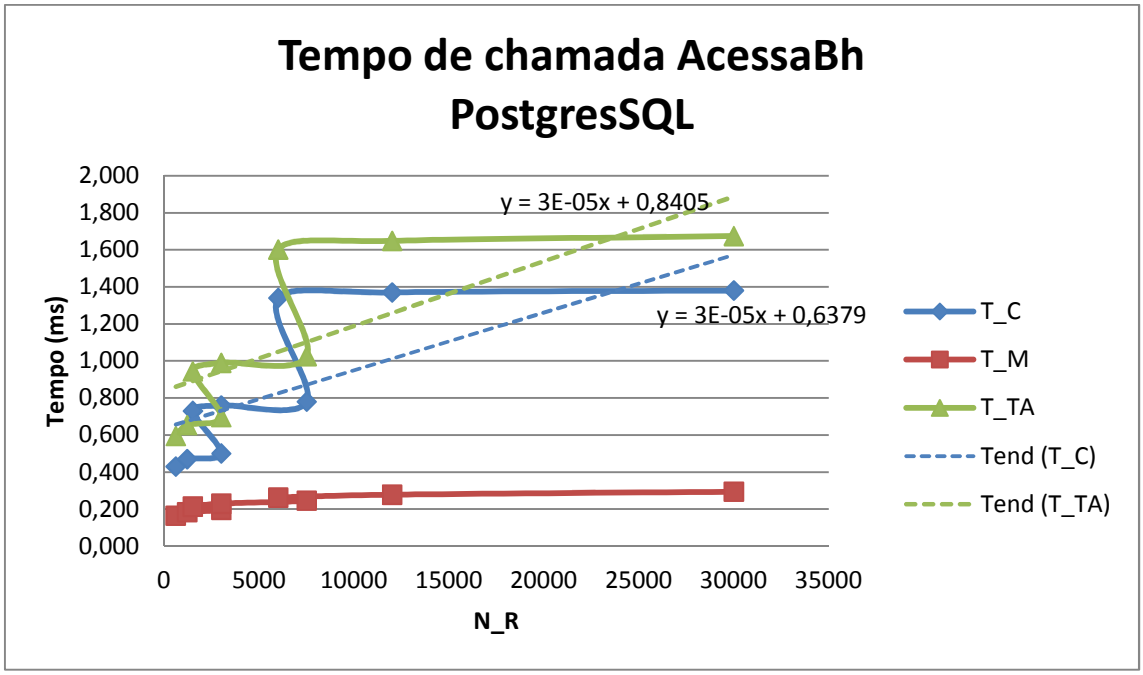


Figura IV-7 - Gráfico de área dos tempos de manipulação e consulta ao BD relacional

- Para BD não relacional

Tabela IV-11 - Tempo total de chamada a biblioteca de acesso em função de T_C e T_M para BD relacional

N_R	Teste	T_C (em seg)	T_M (em seg)	T_{TA}(em seg)
600	1	0,030	1,047	1,077
1200	2	0,057	1,009	1,066
3000	3	0,114	1,117	1,231
1500	4	0,130	1,047	1,177
3000	5	0,342	0,811	1,153
7500	6	0,943	1,509	2,452
6000	7	0,195	1,100	1,295
12000	8	0,510	0,967	1,477
30000	9	1,350	1,933	3,283

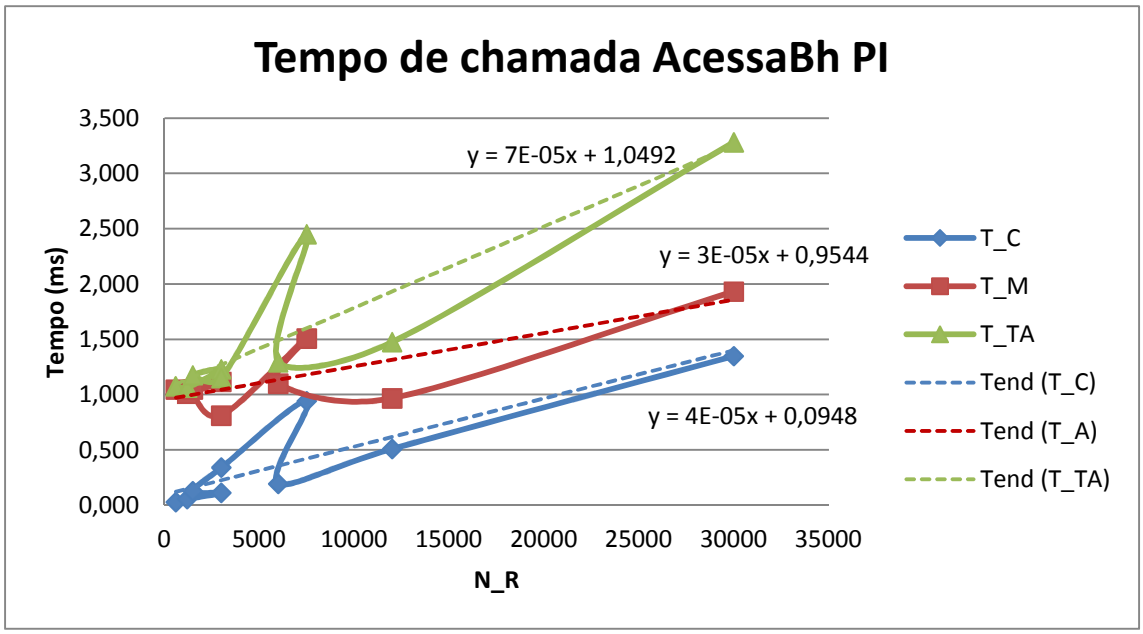


Figura IV-8 - Gráfico de área dos tempos de manipulação e consulta ao BD não relacional

É possível perceber pelos gráficos nas figuras Figura IV-7 e Figura IV-8 a diferença para o exemplo temporal, com os testes apresentando desempenhos semelhantes em consultas de poucos pontos. Percebe-se também que uma tendência conforme o número de pontos ou atributos aumenta, leva a crer que o banco PostgreSQL tenha um desempenho melhor que o banco não relacional. Isto ocorre provavelmente porque, na lógica interna da biblioteca de acesso, a primeira chamada a função de consulta da API do PI é feita para uma tag específica e as demais chamadas para essa mesma tag retornam os demais valores no tempo (lembrando que cada tag do PI é formada no SAGE pela junção do identificador do ponto com o atributo relacionado).

O tempo de retorno de arquivos CSV para a nuvem, T_{Ecsv}, apresentou-se como:

Tabela IV-12 - Valores para T_{Ecsv} em função dos testes e do tamanho do arquivo CSV formado para envio

Teste	T_{Ecsv}	Tamanho (em KB)
1	0,94	18
2	0,99	36
3	2,03	96
4	1,06	46
5	1,99	95
6	3,92	184
7	3,74	158
8	6,53	330
9	14,46	742

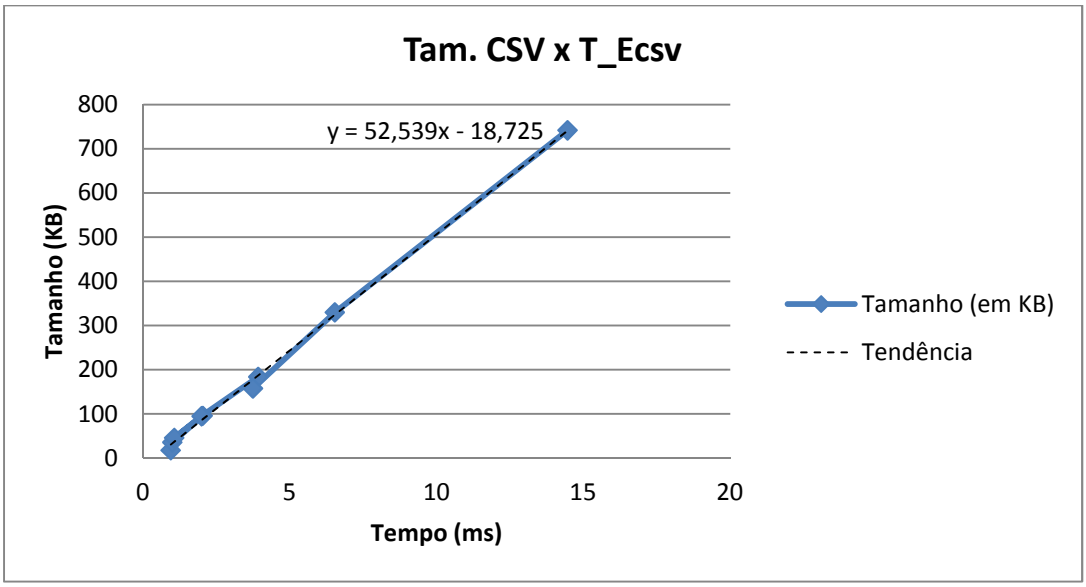


Figura IV-9 - Gráfico do tempo de envio dos arquivos CSV em função do tamanho do arquivo

Pelo gráfico da Figura IV-9 é possível perceber que a taxa média de *upload* de arquivo para a nuvem Openshift é de cerca de 50KB/s, uma taxa bem razoável para o problema, já que não estamos trabalhando com dados demasiadamente pesados.

O tempo total no modelo da temporalidade de acordo com as equações IV-1, IV-2 e IV-3 ficou:

- BD relacional

Tabela IV-13 - Tempo total da consulta para um único identificador com intervalos e períodos distintos e BD relacional

T_{Rcsv} (seg)	T_C (seg)	T_M (seg)	T_{Ecsv} (em seg)	T_T (em seg)
0,446	0,430	0,164	0,940	1,980
0,469	0,470	0,183	0,990	2,112
0,474	0,500	0,197	2,030	3,201
0,474	0,730	0,214	1,060	2,477
0,474	0,760	0,230	1,990	3,453
0,452	0,780	0,246	3,920	5,398
0,463	1,340	0,262	3,740	5,805
0,494	1,370	0,278	6,530	8,671
0,459	1,380	0,294	14,460	16,593

- BD não relacional

Tabela IV-14 - Tempo total da consulta para um único identificador com intervalos e períodos distintos e BD não relacional

T_{Rcsv} (seg)	T_C (seg)	T_M (seg)	T_{Ecsv} (em seg)	T_T (em seg)
0,446	0,030	1,047	0,940	2,463
0,469	0,057	1,009	0,990	2,525
0,474	0,114	1,117	2,030	3,735
0,474	0,130	1,047	1,060	2,711
0,474	0,342	0,811	1,990	3,617
0,452	0,943	1,509	3,920	6,824
0,463	0,195	1,100	3,740	5,498
0,494	0,510	0,967	6,530	8,501
0,459	1,350	1,933	14,460	18,202

No aspecto espacial, os testes apresentaram desempenhos razoáveis para ambos os bancos de dados, conforme visto nas colunas de T_T das tabelas Tabela IV-13 e Tabela IV-14, sendo possível observar como um número elevado de registros (da ordem de dezenas de milhares) no arquivo CSV de resposta tem uma influência grande no tempo total de consulta.

É perceptível também que consultas de múltiplos pontos tendem a apresentar um atraso maior ao se utilizar a API do BD não relacional, afetando o tempo total de consulta. Ainda assim, o atraso maior é verificável pelo número total de registros no arquivo CSV de resposta.

IV.4 Comentários

Tendo como linha de orientação os elementos básicos descritos na introdução aos testes, o objetivo maior do estudo era identificar como os blocos computacionais se comportam dentro da arquitetura de comunicação proposta. As variáveis foram analisadas de acordo com a troca de mensagens, e os cenários distintos foram observados minuciosamente.

Os requisitos de desempenho e disponibilidade do ambiente computacional de um centro de controle fazem com que sua infraestrutura computacional seja bem específica e complexa. Ainda assim, foi provado que é possível disponibilizar parte destes dados, ainda que em pouca quantidade, utilizando essa mesma infraestrutura. Os tempos de resposta às requisições foram bem razoáveis, tornando o modelo de consulta a partir de nuvens uma proposta segura e viável, principalmente ao se levar em consideração que a fonte de dados é histórica e não há uma necessidade de tempos de respostas tão rápidos para casos de estudo. Caso contrário, em tempo real, por exemplo, tempos de respostas de mais de 10 segundos podem se apresentar inviáveis dependendo da situação.

Capítulo V: Conclusões e perspectivas

As motivações para adoção de computação em nuvem são bastante variadas. O cenário do mercado já é bastante propício para iniciativas pequenas ou grandes serem desenvolvidas considerando seus modelos, com riscos bem identificáveis e gerenciáveis.

A tendência aponta para serviços cada vez mais baratos e, com gerenciamento de recursos e custos mais simplificado e detalhado. Isso facilitará cada vez mais verificar vantagens ou desvantagens na adoção de nuvem.

Se olharmos para os próximos anos, o uso da Computação em Nuvem deve aumentar significativamente. Além da tecnologia estar disponível (proliferação de conexões de banda larga, processadores mais baratos e cada vez mais poderosos e custo de armazenamento caindo significativamente) permitindo a criação de grandes data centers, pesquisas vêm mostrando que os usuários mais jovens usam de forma mais intensa estes serviços do que os da geração anterior. Essa nova geração que desponta no mercado de consumo e trabalho tende a acelerar sua utilização. Soma-se a isso a pressão por mais eficiência da infraestrutura de computação, que é cada vez maior. Há pouco menos de dois anos atrás, no mundo inteiro eram gerados a cada dia quinze petabytes de informação. Os custos de energia elétrica aumentaram pelo menos oito vezes do ano de 1996 até o ano de 2011 e mais de um terço da população mundial já está na internet. Portanto, fica evidente que os modelos atuais de gestão de infraestrutura não são mais adequados.

Uma grande vantagem na nuvem é que prover infraestrutura é extremamente simples. É possível usar o mínimo que atenda às necessidades computacionais e escalonar tanto de forma automática ou agendada os recursos de forma a prover as necessidades previstas. Porém, hoje em dia as nuvens gratuitas ainda possuem limitações quanto aos recursos disponíveis (memória e espaço em disco), fazendo com que muitas aplicações na maioria das vezes fiquem com restrições indesejáveis.

O protocolo de comunicação com a nuvem é dependente das duas partes (local e remoto), levando a crer, a partir das experiências feitas, que o atraso no envio e recebimento dos dados está amplamente associado à rede e a forma como a nuvem gratuita (local) permite a troca de informação com o servidor de dados (remoto). Isso também é possível de se perceber intuitivamente, ao se realizar consultas de múltiplos

pontos com estampas de tempo bem próximas ao horário corrente e observando os atrasos.

As experiências aqui relatadas levaram em consideração pequenas amostras de dados para consulta, já que um usuário *backend* da arquitetura consegue analisar visualmente apenas uma pequena gama de dados. Isto vai ao encontro do uso da nuvem gratuita, que limita o espaço em disco e consequentemente a quantidade de informações disponíveis. No entanto, nada impede que em projetos maiores e bem definidos o trabalho seja feito com amostras mais largas e nuvens mais completas.

A diferença entre a solução tradicional, local, e a utilizando computação em nuvem, está ligada aos tempos de envio e recebimento dos arquivos por parte do servidor na nuvem, que não acontece na solução tradicional. Mesmo considerando o atraso relativo a esses tempos, a proposta se mostra eficiente a medida que o tempo total é aceitável para ambientes de estudos.

Uma possível expansão da pesquisa pode levar em consideração um módulo novo na parte ligada aos sistemas gerenciadores, disponibilizando informações vindas diretamente do tempo real, ou seja, no caso do SAGE, vindo diretamente das MCDs, o que diminuiria a latência na consulta por parte da biblioteca de acesso. Dessa forma também poderia não haver mais requisições definidas por período por parte do usuário final, e sim um constante envio de pequenos blocos de dados por parte do servidor para manter o aplicativo de visualização atualizado sobre as mudanças dos pontos selecionados. Inicialmente, o módulo de acesso ao tempo real poderia ser introduzido na arquitetura da seguinte forma:

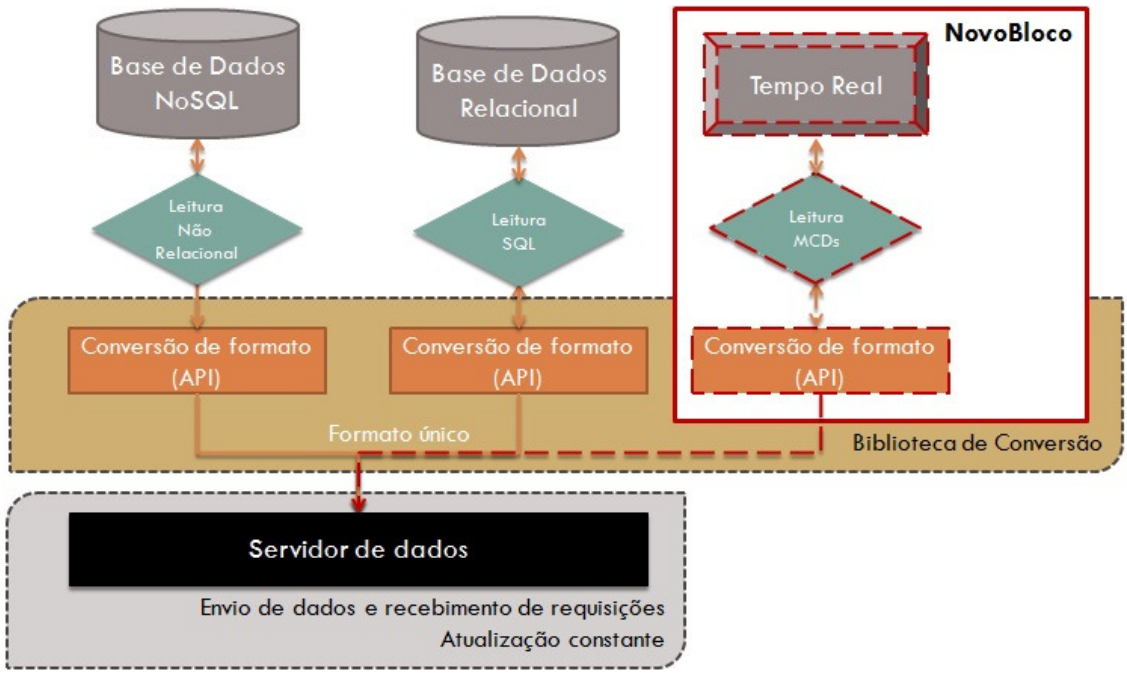


Figura V-1 - Inclusão de possível novo módulo de acesso aos dados do tempo real

Ainda, podem ser elaboradas novas formas de representação dos dados no lado da nuvem, com interfaces múltiplas e concomitantes, como por exemplo, um display gráfico de tendência para um determinado ponto analógico e um replay de ocorrência de eventos a partir de um histórico de dados.

Uma grande vantagem dessa arquitetura é que ela pode permitir o uso de banco de dados na nuvem para armazenar apenas os dados requisitados (e não a base completa dos sistemas gerenciadores), permitindo que usuários distintos possam acessar essa base e fazer uma averiguação mais detalhada sobre um mesmo ponto ou problema específico.

Portanto, a proposta inicial se mostra viável e expansível, existindo inúmeras possibilidades de ampliação da sua capacidade, que podem ser elaboradas dependendo das necessidades dos utilizadores do sistema de gerencia de energia elétrica. Se as motivações forem adequadamente mapeadas e as avaliações destacarem muito mais vantagens do que desvantagens na utilização de computação em nuvem, certamente qualquer iniciativa de implantação do modelo terá ótimas condições de alcançar sucesso.

Os centros de controle de hoje estão no estado de transição da arquitetura centralizada de ontem para arquitetura distribuída de amanhã **[21]**.

Referências Bibliográficas

[1] TAURION, C., 2009, *Computação em Nuvem: Transformando o mundo da tecnologia da informação*. Rio de Janeiro, Brasport.

[2] CARR, N., 2013, *Big Switch: Rewiring the World, from Edison to Google*. 1 ed., New York, USA, W. W. Norton & Company.

[3] BUYYA, R.; YEO, C. S.; VENUGOPAL, S.; BROBERG, J.; BRANDIC, I., 2009, "Cloud Computing and Emerging IT Platforms: Vision, Hype, and Reality for Delivering Computing as the 5th Utility", *Future Generation Computer Systems*, v.25, n.6 (Jun), pp 599-616.

[4] VISCIPOWER**.** *Sistema Elétrico Brasileiro*. Available at: **<**http://www.dca.fee.unicamp.br/projects/vdx/siqueira/eletr.html>. Accessed on: May 12 2014.

[5] ONS - Operador Nacional do Sistema Elétrico. *Operação do Sistema Elétrico*. Disponível em <http://www.ons.org.br/educativo/index.aspx>. Acessado em 13 de Junho de 2014.

[6] SCHWEITZER ENGINEERING LABORATORIES, 2010, "Exemplos de automação em sistemas de supervisão e controle de subestações e redes de distribuição", *O Setor Elétrico*, v.5, n.56 (Sep), pp 54-60.

[7] DY LIACCO, T. E., 1974, "Real-Time Computer Control of Power Systems". *Proceedings of the IEEE*, v.62, n.7 (Jul), p.884-891.

[8] ARMBRUST, M. et. al., 2010, "Above the Clouds: A View Of Cloud Computing", *Communications of the ACM*, April.

[9] MELL, P., GRANCE, T., 2011, "The NIST definition of cloud computing: recomendations of the National Institute of Standards and Technology", *NIST Special Publication,* (Jan) 800-145.

[10] VOUK., M. A., 2008, "Cloud Computing - Issues, Research and Implementations", *Journal of Computing and Information Technology*, v.16, n.4, 235 – 246.

[11] SOUSA, F., MOREIRA, L., MACHADO, J., 2009, "Computação em Nuvem: Conceitos, Tecnologias, Aplicações e Desafios. In: Antônio Costa de Oliveira; Raimundo Santos Moura; Francisco Vieira de Souza". *(Org.). III Escola Regional de Computação Ceará, Maranhão e Piauí* (ERCEMAPI). 1 ed. Teresina: SBC, 2009, v. 1, p. 150-175.

[12] ABOULNAGA, A., SALEM, K., SOROR, A. A., MINHAS, U. F., KOKOSIELIS, P., KAMATH, S. 2009. "Deploying database appliances in the cloud." *IEEE Data* Eng. Bull., v.32, n.1, 13 – 20.

[13] AMAZON. Estudo de caso da AWS: Operador Nacional do Sistema Elétrico. Available at: <http://aws.amazon.com/pt/solutions/case-studies/operador-nacional-do-sistema-eletrico/>. Accessed on: Jan 19 2014.

[14] AMAZON. *Amazon EC2*. Available at: <http://aws.amazon.com/ec2/>. Accessed on: May 08 2014.

[15] CENTRO DE PESQUISAS EM ENERGIA ELÉTRICA - CEPEL. 2014. SAGE - Sistema Aberto de Supervisão e Controle. *Manual de Administração,* 2014. . Available at: <http://www.sage.cepel.br >. Accessed on: June 02 2014.

[16] MOREALE, M. dos S. 2007. *Técnicas para Treinamento de Operadores de Sistema Elétrico Utilizando Simulador Com Base na Interface de Tempo Real* (Tese de Mestrado). Florianópolis, SC, Brasil : Universidade Federal de Santa Catarina.

[17] OSISOFT. *The PI System*. Available at: < http://www.osisoft.com//>. Accessed on: 14 May 2014.

[18] OPENSHIFT. *The Open-Hybrid Cloud Application Platform by Red Hat*. Available at: <http://www.openshift.com>. Accessed on: December 23 2013.

[19] NoSQL. *The last generation of databases*. Available at: <http://nosql-database.org/> Accessed on: January/2014.

[20] SHAIKH, F.B.; HAIDER, S.,2011,"Security Threats in Cloud Computing", *International Conference for Internet Technology and Secured Transactions*, Abu Dhabi, United Arab Emirates, 11-14 December 2011.

[21] WU, F. F.; MOSLEHI, K.; BOSE, A., 2005, "Power System Control Centers: Past, Present, and Future". *Proceedings of the IEEE*, v. 93, n.11 (Nov), pp 1890-1908

APÊNDICE A: Biblioteca RecSet

Biblioteca para armazenamento e transporte de informações.

O armazenamento de dados é feito no formato tabular, com linhas (records) e colunas ou campos (fields).

Um RecSet é criado com a função RsNew, que recebe um nome opcional. Ela retorna um ponteiro para um RecSet.

A função RsDelete apaga um RecSet. Ela recebe uma referencia para uma variável que contem um ponteiro para um RecSet, pois aproveita para colocar NULL na variável, reduzindo a possibilidade de se tentar usar um RecSet ja removido.

Pode-se consultar o numero de linhas ({@link #RsRecordCount RsRecordCount}) e colunas ({@link #RsFieldCount RsFieldCount}).

As linhas são numeradas, começando com 0. Elas são inseridas e removidas dinamicamente, com o uso das funções RsAddRecords, RsInsertRecords, RsRemoveRecords e RsZapRecords.

A função RsAddRecords retorna o índice da primeira linha adicionada, devendo ser colocado em uma variável local e ser usado para preencher suas células.

Colunas também são inseridas e removidas dinamicamente, com o uso das funções RsAddField, RsAddFixedStrField, RsRemoveField e RsZapFields. O nome de uma coluna pode ser alterado com a funcao RsRenameField.

A função RsHasField verifica a existência de uma coluna com determinado nome.

A relação de nomes das colunas de um RecSet pode ser obtida com um simples loop, de zero a RsFieldCount, usando a função RsGetFieldName. Cada coluna é identificada por um nome. Ao ser criada, especifica também o tipo de dado que suas células conterão:

RS_CHAR

RS_SHORT

RS_INT

RS_LONG

RS_FLOAT

RS_DOUBLE

RS_STRING

RS_FIXED_STRING

RS_RECSET

RS_POINTER

RS_TIME

O conteúdo de cada célula pode ser definido com as funções RsSet* e consultado com as funções RsGet*.

Células de tipos numéricos são inicializadas com zero. As demais são inicializadas com NULL.

A função RsSetInteger é usada para armazenar qualquer tipo de dado inteiro (char, short, int, long e time).

A função RsSetString cria uma copia da string recebida, não dependendo da aplicação para guardar o dado. Se receber NULL, libera a memória e armazena NULL.

Já as funções RsSetRecSet e RsSetPointer simplesmente armazenam ponteiros para os argumentos recebidos.

A função RsDelRecSet, alem de colocar NULL na célula correspondente, apaga o RecSet que estava sendo apontado.

As funções RsGet* aceitam consultas para colunas com tipo de dado diferente da função usada, convertendo a informação automaticamente, a não ser que tal conversão não seja possível. Por exemplo, ao usar a função RsGetInteger para uma célula de uma coluna com tipo de dados string, ela tenta converter a string para numero inteiro.

A função RsGetString converte qualquer dado numérico para string.

Uma aplicação pode ter acesso direto ao buffer de memória alocado para armazenar as células de uma coluna com a função RsGetFieldBuffer.

Alem das células, dados podem ser armazenados em uma lista de propriedades do RecSet, acessíveis através das funções RsSetProperty e RsGetProperty.

Cada coluna também tem sua própria lista de propriedades, acessada pelas funções RsSetFieldProperty e RsGetFieldProperty

Buscas dentro do RecSet são feitas com as funções RsSearch*.

Se for chamada a função RsBuildHashTable para uma determinada coluna, as buscas passam a usar algoritmos extremamente rápidos de procura. Isto só é possível para colunas de tipo string e sem duplicidade no conteúdo.

As funções RsReadTextFile, RsReadCsvFile e RsReadIniFile são usadas para ler arquivos.

As funções RsRecSetToCsv e RsCsvToRecSet convertem RecSets em strings CSV e vice-versa.

A função RsSplitString cria um RecSet a partir da separação de uma string em pedaços, mantendo a string original intacta.

A função RsRegexSplitString cria um RecSet a partir da divisão de uma string em trechos separados por uma expressão regular.

A função RsRegexSplitField cria um RecSet a partir da divisão do conteúdo de um campo de um RecSet em trechos separados por uma expressão regular.

A função RsLockRecords pode ser usada para bloquear os registros de um RecSet, não permitindo mais a inserção ou remoção de linhas.

O mesmo pode ser feito com colunas através da função RsLockFields.

A função RsDump é um meio pratico de depurar uma aplicação, mostrando em stdout o conteúdo de um RecSet.

A função RsLastStatus é usada para se obter o código de erro da ultima operação efetuada.

A descrição de cada código de erro está disponível através da função RsStatusDescr.

APÊNDICE B: Esquema de consulta a último valor por período em banco PostgreSQL

Comandos a serem executados no banco de dados PostgreSQL:

-- Função que sempre retorna o último valor não nulo.

```
CREATE OR REPLACE FUNCTION public.last_agg ( anyelement, anyelement )
RETURNS anyelement LANGUAGE sql IMMUTABLE STRICT AS $$
        SELECT $2;
$$;
```

-- Criação de um "agregado" para a função.

```
CREATE AGGREGATE public.last (
        sfunc    = public.last_agg,
        basetype = anyelement,
        stype    = anyelement
);
```

-- Select genérico utilizado na base histórica do SAGE para consulta de último valor por intervalo.

```
SELECT ID,
TO_TIMESTAMP((CEIL((EXTRACT(EPOCH FROM BH_DTHR) -
EXTRACT(EPOCH FROM TIMESTAMP <hora_inicial> ))/ <intervalo> )*
<intervalo> )+EXTRACT(EPOCH FROM TIMESTAMP <hora_inicial> ))  AS
BH_DTHR_CEIL,
LAST( <atributo_dinamico_1> ),  LAST( <atributo_dinamico_2> ),...  LAST(
<atributo_dinamico_N> )
FROM PAS_H H, PAS_R R
WHERE  BH_DTHR >= <hora_inicial>
AND BH_DTHR < <hora_final>
```

AND ID IN (**<nome_identificador_1>**, **<nome_identificador_2>** ,… **<nome_identificador_N>**)

AND R.BH_SINONIMO IS NULL

AND R.BH_CHAVE=H.BH_CHAVE

GROUP BY ID, TO_TIMESTAMP((CEIL((EXTRACT(EPOCH FROM BH_DTHR) - EXTRACT(EPOCH FROM TIMESTAMP **<hora_inicial>**))/ **<intervalo>**)* **<intervalo>**)+EXTRACT(EPOCH FROM TIMESTAMP **<hora_inicial>**))

ORDER BY ID, TO_TIMESTAMP((CEIL((EXTRACT(EPOCH FROM BH_DTHR) - EXTRACT(EPOCH FROM TIMESTAMP **<hora_inicial>**))/ **<intervalo>**)* **<intervalo>**)+EXTRACT(EPOCH FROM TIMESTAMP **<hora_inicial>**)) ASC;

--Exemplo de uma consulta:

SELECT ID,

TO_TIMESTAMP((CEIL((EXTRACT(EPOCH FROM BH_DTHR) - EXTRACT(EPOCH FROM TIMESTAMP '2014-09-04 22:00:03'))/300)*300)+EXTRACT(EPOCH FROM TIMESTAMP '2014-09-04 22:00:03')) AS BH_DTHR_CEIL, LAST(V)

FROM PAS_H H, PAS_R R

WHERE BH_DTHR >= '2014-09-04 22:00:03'

AND BH_DTHR < '2014-09-04 23:00:03'

AND ID IN ('PAIN_500_LTPAMB_2_KV')

AND R.BH_SINONIMO IS NULL

AND R.BH_CHAVE=H.BH_CHAVE

GROUP BY ID, TO_TIMESTAMP((CEIL((EXTRACT(EPOCH FROM BH_DTHR) - EXTRACT(EPOCH FROM TIMESTAMP '2014-09-04 22:00:03'))/300)*300)+EXTRACT(EPOCH FROM TIMESTAMP '2014-09-04 22:00:03'))

ORDER BY ID, TO_TIMESTAMP((CEIL((EXTRACT(EPOCH FROM BH_DTHR) - EXTRACT(EPOCH FROM TIMESTAMP '2014-09-04 22:00:03'))/300)*300)+EXTRACT(EPOCH FROM TIMESTAMP '2014-09-04 22:00:03')) ASC;