UNIOESTE - UNIVERSIDADE ESTADUAL DO OESTE DO PARANÁ
CENTRO DE ENGENHARIAS E CIÊNCIAS EXATAS
CURSO DE CIÊNCIA DA COMPUTAÇÃO

Avaliação do modelo de integração Fix Script a Portais de Serviço Corporativos Internos: um estudo de caso

Isabela Pimentel Loebel

FOZ DO IGUAÇU
2023

Isabela Pimentel Loebel

Avaliação do modelo de integração Fix Script a Portais de Serviço Corporativos Internos: um estudo de caso

Monografia submetida à Universidade Estadual do Oeste do Paraná, Curso de Ciência da Computação - Campus de Foz do Iguaçu, como requisito parcial para obtenção do grau de Bacharel em Ciência da Computação.

Prof. Renato Bobsin Machado

Rodrigo Renie de Braga Pinto

FOZ DO IGUAÇU

2023

Isabela Pimentel Loebel

Avaliação do modelo de integração Fix Script a Portais de Serviço Corporativos Internos: um estudo de caso

Documento assinado digitalmente
gov.br
RENATO BOBSIN MACHADO
Data: 11/06/2024 22:13:12-0300
Verifique em https://validar.iti.gov.br

Prof. Renato Bobsin Machado
Orientador(a)

Documento assinado digitalmente
gov.br
RODRIGO RENIE DE BRAGA PINTO
Data: 13/06/2024 14:40:38-0300
Verifique em https://validar.iti.gov.br

Rodrigo Renie de Braga Pinto
Coorientador(a)

Me. Teresinha Arnauts Hachisuca
Membro

Documento assinado digitalmente
gov.br
ADRIANA MOREIRA
Data: 12/06/2024 18:51:12-0300
Verifique em https://validar.iti.gov.br

Esp. Adriana Moreira
Membro

Dedico este trabalho à minha família, em especial à minha irmã, Ana Julia, para que saiba que a persistência é a chave para tudo. Também ao meu companheiro, Nickolas, por sua ajuda em me guiar de volta à luz da existência e por me fornecer a força necessária para continuar. Por fim, dedico este trabalho principalmente a Isabela do passado, que por vezes questionou o caminhar, mas nunca os objetivos. Eu, como Isabela do presente, afirmo que nós conseguimos.

Agradecimentos

A Deus, pela minha vida e por me permitir superar todos os obstáculos encontrados ao longo do curso e da vida. Aos meus pais, Gilberto e Clélia, e à minha irmã, Ana Julia, a melhor irmã e amiga que alguém poderia ter, por me incentivarem nos momentos difíceis e compreenderem minha ausência enquanto me dedicava à realização deste trabalho.

Agradeço especialmente ao meu companheiro de vida e profissão, Nickolas, pelo apoio, paciência e por trazer calma e clareza aos momentos mais turbulentos.

Expresso minha gratidão aos amigos que estiveram ao meu lado em todas as fases da vida, especialmente Vivian, Kevin, Kauany e Renan, pelo companheirismo e pela troca de experiências que me permitiram crescer tanto profissionalmente quanto pessoalmente, assim como às lembranças do passado que, embora fantasmas, despertaram em mim a voracidade de alcançar meus objetivos, já que somente eu poderia fazê-lo.

Por último, quero agradecer também à Universidade Estadual do Oeste do Paraná e seu corpo docente, sobretudo ao meu orientador, Prof. Renato Bobsin Machado, pela confiança depositada na minha proposta de projeto, por me manter motivada durante todo o processo e pela amizade construída ao longo dessa caminhada.

"Não espere por circunstâncias ideais, tome decisões e torne-as ideais."
(George Bernad Shaw)

Resumo

Conforme Abinader e Lins [2006], a evolução tecnológica proporciona uma ampla variedade de ferramentas para o desenvolvimento de aplicações web, cada uma com suas próprias limitações e peculiaridades, especialmente no que tange à troca de informações de maneira simples e automática. O ServiceNow, apesar de ser uma ferramenta poderosa, compartilha uma deficiência comum a muitas grandes empresas, conforme destacado por Menéndez [2002]: a integração entre aplicações. Esta integração envolve a utilização unificada de processos provenientes de diferentes sistemas e desenvolvidos em diversas linguagens. A imprecisão no planejamento dessas integrações, especialmente em termos de desempenho, revela a necessidade de explorar novas abordagens. Este estudo visa facilitar e formalizar a seleção da melhor abordagem para unificar dados externos com o ServiceNow (SNow) através da ferramenta Fix Script. Embora o ServiceNow demonstre alta capacidade de gestão de grandes volumes de dados via Mid Server, a eficácia da plataforma pode variar conforme a natureza da integração. Consultas para recuperar um único dado por meio de script podem não se beneficiar da mesma otimização que aquelas envolvendo grandes volumes de dados. A análise conclui haver uma correlação entre o volume de dados e o tempo de execução, indicando que, no contexto do ServiceNow, um alto volume de dados é estatisticamente menos oneroso em termos de desempenho quando consultado via Fix Script. No entanto, para consultas individuais, essa abordagem pode não ser ideal. O cenário ideal para a integração via Fix Script envolve a importação de grandes volumes de dados, processáveis em segundo plano. Apesar do desempenho superior com grandes volumes de dados, a integração via Fix Script pode ainda apresentar um tempo de execução relativamente alto.

Palavras-chaves: ServiceNow, integração, Fix Script, volume de dados, tempo de execução, desempenho.

Abstract

As per Abinader e Lins [2006], technological evolution provides a wide variety of tools for the development of web applications, each with its own limitations and peculiarities, especially when it comes to exchanging information in a simple and automatic way. ServiceNow, despite being a powerful tool, shares a common deficiency with many large companies, as highlighted by Menéndez [2002]: the integration between applications. This integration involves the unified use of processes from different systems and developed in various languages. The imprecision in planning these integrations, especially in terms of performance, reveals the need to explore new approaches. This study aims to facilitate and formalize the selection of the best approach to unify external data with ServiceNow (SNow) through the Fix Script tool. Although ServiceNow demonstrates high capacity for managing large volumes of data via Mid Server, the effectiveness of the platform can vary depending on the nature of the integration. Queries to retrieve a single piece of data via script may not benefit from the same optimization as those involving large volumes of data. The analysis concludes that there is a correlation between the volume of data and the execution time, indicating that, in the context of ServiceNow, a high volume of data is statistically less burdensome in terms of performance when queried via Fix Script. However, for individual queries, this approach may not be ideal. The ideal scenario for integration via Fix Script involves the importation of large volumes of data, processable in the background. Despite superior performance with large volumes of data, integration via Fix Script may still present a relatively high execution time.

Keywords: ServiceNow, integration, Fix Script, data volume, execution time, performance.

Lista de ilustrações

Lista de tabelas

Lista de Abreviaturas e Siglas

APIs Application Programming Interface — Interface de Programação de Aplicações

CMDB Configuration Management Database — Banco de Dados de Gerenciamento de Configuração

COBIT Control Objectives for Information and Related Technology — Objetivos de Controle para Informação e Tecnologias Relacionadas

CORBA Common Object Request Broker Architecture — Arquitetura Comum de Solicitação de Objeto Distribuído

DCOM Distributed Component Object Model — Modelo de Objeto de Componente Distribuído

ERP Enterprise Resource Planning System — Sistema de Planejamento de Recursos Empresariais

HTTP Hypertext Transfer Protocol — Protocolo de Transferência de Hipertexto

IaaS Infrastructure as a Service — Infraestrutura como um Serviço

IB Itaipu Binacional

ISO/IEC 385000 Corporate Governance of Information Technology — Governança Corporativa de Tecnologia da Informação

ITBM IT Business Management — Gestão de Negócios de TI

ITIL Information Technology Infrastructure Library — Biblioteca de Infraestrutura de Tecnologia da Informação

ITOM IT Operations Management — Gestão de Operações de TI

ITSM Information Technology Service Management — Gerenciamento de Serviços de TI

ITSMF IT Service Management Forum — Fórum de Gerenciamento de Serviços de TI

JSON JavaScript Object Notation — Notação de Objeto Javascript

MID Server Management, Instrumentation, and Discovery Server — Servidor de Gerenciamento, Instrumentação e Descoberta

NIST National Institute of Standards and Technology — Instituto Nacional de Padrões e Tecnologia

PaaS Plataform as a Service — Plataforma como um Serviço

QoS Quality of Service — Qualidade de Serviço

REST Representational State Transfer — Transferência de Estado Representacional

RMI Remote Method Invocation — Invocação Remota de Método

RPCs Remote Procedure Call — Chamadas de Procedimentos Remotos

SaaS Software as a Service — Software como um Serviço

SGBD Sistema de Gerenciamento de Banco de Dados

SLA Service Level Agreement — Acordo de Nível de Serviço

SNow ServiceNow

SOAP Simple Object Access Protocol — Protocolo Simples de Acesso a Objetos

SQL Structured Query Language — Linguagem de Consulta Estruturada

TI Information Technology — Tecnologia da Informação

XML Extensible Markup Language — Linguagem de Marcação Extensível

Sumário

1 Introdução

1.1 Contexto

Na área de tecnologia da informação, há uma busca inerente por praticidade e agilidade na integração de novos serviços com outros já existentes. De acordo com Abinader e Lins [2006], inúmeras ferramentas e tecnologias surgiram nos últimos anos, destinadas ao desenvolvimento de aplicações para a Internet. Entretanto, nenhuma é completa e todas possuem suas limitações e particularidades, como, por exemplo, a troca de informação de forma simples e automática.

Como conjunto de uma solução ideal atual, faz-se cada vez mais envolvida a computação em nuvem. Nesse ambiente, os recursos de Tecnologia da Informação (TI) são fornecidos como um serviço, permitindo que os usuários o acessem sem a necessidade de conhecimento sobre a tecnologia utilizada. Assim, os usuários e as empresas passaram a acessar os serviços sob demanda e independente de localização, o que aumentou a quantidade de serviços disponíveis [SOUSA; MOREIRA; MACHADO, 2009].

Tratando-se do acesso e disponibilidade de ambientes de computação em nuvem, têm-se diferentes tipos de modelos de implantação. A restrição ou abertura de acesso depende do processo de negócio, do tipo de informação e do nível de visão, tendo seus modelos de implantação divididos em nuvem pública, privada, comunidade e híbrida [SOUSA; MOREIRA; MACHADO, 2009].

No ensaio a ser redigido, será utilizada a plataforma em nuvem ServiceNow(SNow), contratada pela Itaipu Binacional(IB) e implementada em modelo privado. Lançada ao mercado como uma ferramenta inovadora, "a ServiceNow é uma plataforma de gerenciamento de serviços de tecnologia da informação (IT Service Management — ITSM) e automação empresarial" [COSTA, 2022a].

Já visionando a necessidade de unificação perante diferentes bases de dados, a SNow atende à demanda proposta por Sukhija et al. [2020]: "uma solução de monitoramento eficaz requer a coleta de informações heterogêneas e informações provenientes de sistemas e fontes complexos e diferentes" (tradução livre). Afirmando, novamente, a imprescindível necessidade de comunicação entre sistemas, visto que em uma corporação de grande porte, incomumente é usado apenas um software para controle de tarefas, mas sim um conjunto.

Juntamente com lançamento, foi concebido o Mid Server para atuar como um facilitador, permitindo a integração de sistemas diversos com a base de dados da instância

do ServiceNow na nuvem. Além disso, o Mid Server oferece uma extensa variedade de funcionalidades para simplificar a integração de sistemas diversos com a base de dados em nuvem do ServiceNow, abrangendo suporte a uma variedade de fontes de dados, como bancos de dados SQL, arquivos JSON, XML, CSV e APIs REST/SOAP, bem como por meio de scripts para formatos personalizados, conforme exposto por ServiceNow [2022c].

A implementação e configuração eficazes para integrações específicas ainda demandam orientações precisas, por tanto, o estudo proposto tem como foco a investigação aprofundada de uma opção de integração fornecida pelo Mid Server, sendo esta através do Fix Script. Este trabalho adotará uma abordagem sistemática, realizando testes de conexões por meio de API SOAP disponibilizada pela IB e banco de dado para comparar e analisar o método de integração. O objetivo é desenvolver uma orientação transparente para otimizar a eficiência e eficácia das integrações de dados, visando aprimorar a implementação e configuração de futuras integrações no contexto empresarial.

1.2 Motivação

Apesar de ser uma ferramenta poderosa para as organizações que a adotam, o ServiceNow compartilha da mesma carência identificada em muitas grandes empresas, conforme destacado por Menéndez [2002]: a integração entre aplicações, que envolve a utilização unificada de processos provenientes de diferentes sistemas e desenvolvidos em diversas linguagens.

A imprecisão no planejamento dessas integrações, considerando especialmente o desempenho, revela a necessidade de explorar um novo campo de estudo. Esta pesquisa visa facilitar e formalizar a seleção da abordagem mais adequada para unificar os dados provenientes de bancos de dados externos com o ServiceNow (SNow) através da ferramenta Fix Script. O propósito deste estudo é fornecer diretrizes claras e práticas recomendadas para a integração eficaz de dados entre o ServiceNow e sistemas externos, visando preencher uma lacuna significativa na estratégia de integração empresarial, a fim de obter padrões que informem possíveis melhoras a ser implementadas.

A análise se concentrará na identificação e avaliação da integração via Fix Script, visando otimizar o desempenho e garantir a interoperabilidade entre sistemas heterogêneos. Essa abordagem visa oferecer uma contribuição significativa para a melhoria da eficiência operacional e o aprimoramento dos processos de negócio nas organizações que utilizam o ServiceNow como plataforma central de gestão de serviços.

Espera-se que os resultados deste estudo forneçam *insights* e recomendações práticas para os profissionais de TI e gestores de integração, capacitando-os a tomar decisões informadas e estratégicas no contexto da integração de dados empresariais. Isso, por sua

vez, promoverá uma maior eficiência e agilidade nas operações organizacionais.

1.3 Objetivos

O alvo principal desse estudo é a criação de um guia de melhores práticas e escolhas para otimização de integrações de bases de dados externas com o ambiente em nuvem ServiceNow que utilizem Fix Scripts como ferramenta de integração.

Dentre os principais objetivos específicos destacam-se:

- Otimizar e prever escolhas a serem tomadas por profissionais de TI ao integrar bases externas ao SNow;
- Fornecer um comparativo do tempo no consumo de API SOAP, agregado ao tempo de processamento do Mid Server;
- Fornecer uma comparação do tamanho do impacto na performance em relação ao volume de dados;
- Apoiar a Itaipu Binacional na tomada de decisões, posteriormente, quanto às integrações ao ServiceNow.

1.4 Estrutura do Estudo

O estudo está estruturado em sete capítulos, onde, a seguir, são apresentados resumos do conteúdo de cada um.

Capítulo 2: Banco de dados

Este capítulo aborda os conceitos fundamentais de bancos de dados, incluindo modelos de bancos e bancos de dados em nuvem.

Capítulo 3: Microsserviços

Este capítulo explora o contexto dos microsserviços e discute diferentes modelos de serviços, incluindo Software como um Serviço (SaaS), Plataforma como um Serviço (PaaS) e Infraestrutura como um Serviço (IaaS).

Capítulo 4: API

Este capítulo se concentra no contexto das Interfaces de Programação de Aplicações (APIs) e discute diferentes tipos de APIs, incluindo SOAP e REST.

Capítulo 5: Gestão de TI

Este capítulo aborda o contexto da gestão de tecnologia da informação, discute ITIL e ServiceNow, incluindo a arquitetura do ServiceNow, o banco de dados do ServiceNow e a comunicação do ServiceNow através do MID Server.

Capítulo 6: Plano de Pesquisa

Este capítulo apresenta o planejamento e cronograma de execução da pesquisa, discute o material e método utilizado e os critérios de avaliação.

Capítulo 7: Desenvolvimento

Este capítulo detalha as tecnologias envolvidas no processo de desenvolvimento do estudo, assim como o próprio fluxo do desenvolvimento e os resultados obtidos.

Capítulo 8: Conclusão

Este capítulo final resume as descobertas do estudo, discute suas implicações e sugere direções para pesquisas futuras.

2 Banco de dados

2.1 Contexto

Nesse capítulo são apresentados os conceitos e premissas de bancos de dados, assim como sua importância no meio computacional. De acordo com Manovich [2015], banco de dados é definido como uma coleção estruturada de dados. Os dados são armazenados e organizados para permitir agilidade na busca e na recuperação por um computador, ou seja, não há nada além de uma simples coleção de itens.

Como afirma Date [2004], banco de dados pode ser considerado o equivalente eletrônico de um armário de arquivamento; isto é, são repositórios para coleções de informações, as quais são chamadas de dados, que podem ser tudo aquilo que tenha algum significado ao indivíduo ou à organização a que o sistema deve servir.

Observa-se que os termos *dados* e *informações* são tratados como sinônimos no meio de TI (Tecnologia da Informação), mas assim como Date [2004] afirma em sua obra, a distinção entre ambos é de suma importância, onde *dados* é usado para referir-se ao que realmente é armazenado no banco de dados, em contrapartida, *informações* trata-se do significado desses dados para determinado usuário.

2.2 Entendendo o banco de dados

Como supramencionado, a diferenciação de um dado e uma informação deve ser clara, um exemplo apresentado por Alves [2014], é a disposição de duas frases, onde a primeira é "está muito quente hoje" e a segunda "a temperatura atual é de 38 graus Celsius", a primeira expressão trata-se de uma *informação* que pode ser obtida através da segunda frase, visto que a segunda informa o valor da alta temperatura, sendo assim, um *dado*. Por fim, fica claro a aplicabilidade do banco de dados, que armazena e processa dados, não informações, como afirma Alves [2014].

O processo de criação de um banco de dados pode ser categorizado em três etapas, sendo elas a definição, construção e manipulação. Alves [2014] conceitua que a etapa de definição do banco de dados concerne à especificação dos tipos de dados, das estruturas e das tabelas de restrições que devem ser impostas aos dados que serão armazenados. Já a construção é o processo de acumular os dados num meio de armazenamento totalmente controlado pelo SGBD (Sistema de Gerenciamento de Banco de Dados). Após os passos anteriores serem concluídos, pode-se aplicar a manipulação, sendo operações como atualização (entende-se inclusão, exclusão e alterações de registros) do banco de dados e

extração de informações, como consultas e relatórios.

Dentre os modelos de bancos de dados, há dois padrões muito utilizados de acordo com Alves [2014] e Manovich [2015], o padrão de bancos de dados relacionais e o padrão de banco de dados orientados a objetos, sendo distintos pela classificação quanto ao modelo de dados. Sucintamente, "o modelo relacional se caracteriza pela organização dos dados em tabelas (ou relações), formadas por linhas e colunas. Um banco de dados relacional permite que se tenham informações divididas entre várias tabelas de dados, entretanto, certas informações de uma tabela são obtidas a partir de outras, apresentando assim, a necessidade de um campo em comum em diversas tabelas para ser possível definir relacionamentos entre elas" [ALVES, 2014].

Na Figura 1, Alves [2014] apresenta um exemplo de banco de dados relacional pequeno, contendo dados referentes ao conteúdo de uma adega de vinhos. A Figura 2 mostra um exemplo de uma requisição de busca contra esse banco de dados, assim, com os dados retornados por essa requisição pode-se obter informações, como, por exemplo, todos os vinhos que tenham o atributo "pronto" vinculados ao ano de 2004.

DEP#*	VINHO	PRODUTOR	ANO	GARRAFAS	PRONTO
2	Chardonnay	Buena Vista	2001	1	2003
3	Chardonnay	Geyser Peak	2001	5	2003
6	Chardonnay	Simi	2000	4	2002
12	Joh. Riesling	Jekel	2002	1	2003
21	Fumé Blanc	Ch. St. Jean	2001	4	2003
22	Fumé Blanc	Robt. Mondavi	2000	2	2002
30	Gewürztraminer	Ch. St. Jean	2002	3	2003
43	Cab. Sauvignon	Windsor	1995	12	2004
45	Cab. Sauvignon	Geyser Peak	1998	12	2006
48	Cab. Sauvignon	Robt. Mondavi	1997	12	2008
50	Pinot Noir	Gary Farrell	2000	3	2003
51	Pinot Noir	Fetzer	1997	3	2004
52	Pinot Noir	Dehlinger	1999	2	2002
58	Merlot	Clos du Bois	1998	9	2004
64	Zinfandel	Cline	1998	9	2007
72	Zinfandel	Rafanelli	1999	2	2007

Figura 1 – Exemplo de banco de dados de uma adega de vinhos.

Fonte: [DATE, 2004].

Busca:

```
SELECT VINHO, DEP#, PRODUTOR
FROM   ADEGA
WHERE  PRONTO = 2004 ;
```

Resultado (mostrado, por exemplo, na tela do monitor):

VINHO	DEP#	PRODUTOR
Cab. Sauvignon	43	Windsor
Pinot Noir	51	Fetzer
Merlot	58	Clos du Bois

Figura 2 – Exemplo de busca.

Fonte: [DATE, 2004].

Quanto ao modelo orientado a objetos, podendo ser referido como NoSQL, surgiu a fim de superar problemas dispostos pelo banco de dados relacional. Como explicado por Oliveira [2014], a grande motivação para o movimento NoSQL foi de resolver o problema de escalabilidade dos bancos tradicionais, tendo em vista que pode ser muito caro e/ou complexo escalar um banco de dados relacional.

Observa-se que a grande maioria, se não todos, os comandos em um sistema de gerenciamento de banco de dados relacional são formulados utilizando uma linguagem denominada SQL (*Structured Query Language* — Linguagem de Consulta Estruturada). Um exemplo desta linguagem foi apresentado na Figura 2, como apresentado por Date [2004]. O autor ainda afirma que, inicialmente, foi desenvolvida como uma linguagem própria da IBM, mas o SQL tornou-se um padrão internacional amplamente aceito por praticamente todos os produtos disponíveis comercialmente. Esta linguagem incorpora operações tanto para a definição de dados quanto para a manipulação desses dados.

No âmbito da TI, o conceito de banco de dados encontra-se atualmente interligado com a computação em nuvem. Este entrelaçamento é uma resposta à significativa expansão na quantidade de dados gerados e armazenados globalmente, a qual não é mais quantificada em termos de gigabytes ou terabytes, mas sim em peta, exa e até zettabytes, como destacado por Costa et al. [2012]. A computação em nuvem está se tornando uma das palavras-chave da indústria de TI. “A nuvem é uma metáfora para a Internet ou infraestrutura de comunicação entre os componentes arquiteturais, baseada em uma abstração que oculta à complexidade da infraestrutura. Cada parte desta infraestrutura é provida como um serviço e, estes são normalmente alocados em centros de dados, utilizando hardware compartilhado para computação e armazenamento” [BUYYA; RANJAN; CALHEIROS, 2009; BUYYA et al., 2009].

2.3 Banco de dados em nuvem

"Computação em nuvem é uma tendência recente de tecnologia cujo objetivo é proporcionar serviços de TI sob demanda com pagamento baseado no uso" [BUYYA; RANJAN; CALHEIROS, 2009]. A computação em nuvem tem a aspiração de ser global e oferecer serviços abrangentes, atendendo desde o usuário final que armazena seus documentos pessoais na internet até empresas que terceirizam toda a infraestrutura de Tecnologia da Informação para outras organizações. Nunca uma abordagem para a utilização prática demonstrou-se tão abrangente e global: não se restringindo apenas à entrega sob demanda de recursos de computação e armazenamento, mas abarcando toda a pilha de computação disponível na nuvem, como destacado por Sousa, Moreira e Machado [2009].

Ainda em 2009, Abadi [2009] afirmava que SGBDs são candidatos potenciais para a implantação em nuvem. Isso ocorre porque, em geral, as instalações destes sistemas são complexas e envolvem uma abundante quantia de dados, ocasionando um custo elevado, tanto em hardware quanto em software. Para muitas empresas, especialmente para *start-ups* e médias empresas, o pagamento baseado no uso do modelo de computação em nuvem, juntamente com o suporte para manutenção do hardware, é muito atraente.

Na infraestrutura de nuvem, os usuários dos serviços contam com determinadas garantias, como desempenho e disponibilidade. Tais garantias de QoS (*Quality of Service* — Qualidade de Serviço) são estabelecidas em acordos entre o provedor do serviço e o usuário, sendo formalizadas por meio de um SLA (*Service Level Agreement* — Acordo de Nível de Serviço). Sousa, Moreira e Machado [2011] e Moreira [2014] ainda afirmam que, este acordo consiste em contratos que definem um padrão de qualidade que deve ser mantido, estabelecendo, também, penalidades a serem aplicadas em caso de não conformidade.

Consoante a Carr et al. [2023], a ausência de desempenho adequado pode resultar em atrasos no processamento de consultas e transações, afetando negativamente a produtividade e os tempos de resposta operacionais. Tal cenário pode acarretar consequências financeiras adversas para a empresa. Dessa forma, torna-se imperativo avaliar o desempenho do SGBD (Sistema Gerenciador de Banco de Dados) para assegurar a eficiência e a eficácia no processamento de dados, especialmente em organizações de grande porte.

3 Microsserviços

3.1 Contexto

O ambiente na nuvem é composto por vários softwares que oferecem suporte aos diversos aspectos ortogonais da aplicação. Aspectos como registros (*logs*), autenticação, orquestração, elasticidade e rede são gerenciados por produtos de software fornecidos como serviços ou instalados pela equipe, conforme discutido por Muniz et al. [2022].

A arquitetura de microsserviços, como afirma Fowler e Lewis [2014], Newman [2015], comumente referida apenas como microsserviços, originou-se empiricamente a partir de padrões arquiteturais empregados no ambiente prático, onde sistemas são constituídos por serviços que interagem colaborativamente para alcançar seus objetivos, se comunicando por meio de mecanismos leves.

A IBM [2021] ressalta que, a adoção de microsserviços é apresentada como uma alternativa ao modelo monolítico de arquitetura onde um aplicativo é composto por diversos componentes independentes e com acoplamento fraco. Neste modelo, a comunicação entre os serviços ocorre por meio da integração de APIs (*Application Programming Interface* — Interface de Programação de Aplicações). A principal vantagem desse modelo reside na independência dos componentes, que podem ser desenvolvidos em diferentes linguagens de programação e até mesmo serem hospedados em servidores distintos. “A manutenção de um serviço desta arquitetura não deve causar grande impacto no sistema na totalidade” [IANKOSKI, 2022].

A Figura 3 apresenta um exemplo de arquitetura monolítica, disposta por Richardson [2015]. Na parte central, a aplicação abrange toda a lógica de negócios, a qual pode ser implementada por meio de módulos que delineiam serviços, objetos de negócios e eventos. Essa aplicação estabelece conexões com um SGBD para consultas e armazenamento de dados, além de se conectar a outros serviços externos, como, por exemplo, um serviço de consulta de CEP, como agregado por Villaça, Jr e Azevedo [2018].

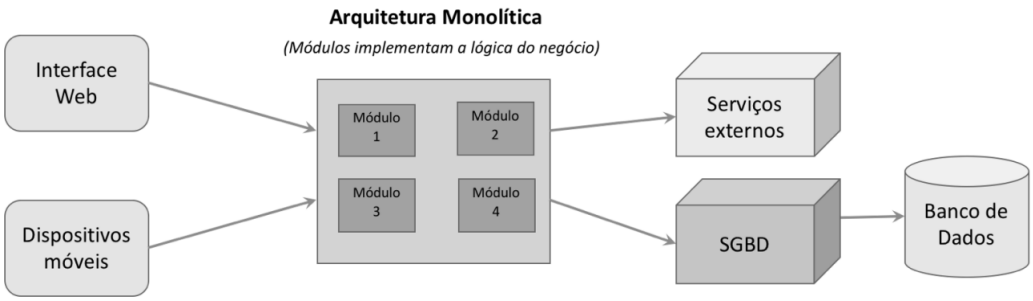

Figura 3 – Um exemplo de arquitetura monolítica.

Fonte: [VILLAçA; JR; AZEVEDO, 2018], adaptado originalmente de [RICHARDSON, 2015].

Dificuldades começam a surgir à medida que uma aplicação monolítica cresce, tornando-se excessivamente complexa. Consequentemente, a tarefa de corrigir erros *(bugs)* e implementar novas funcionalidades de maneira precisa torna-se desafiadora e consome considerável tempo. A complexidade é exacerbada quando o código é de difícil compreensão, dificultando ainda mais a evolução adequada da aplicação. Adicionalmente, aplicações de grande porte podem apresentar demora na inicialização. Caso seja necessário implantar a aplicação diversas vezes ao longo do dia, torna-se necessário reimplantar a aplicação inteira para atualizar uma parte específica do código. Esse desafio é ampliado ao empregar escalonamento em múltiplos servidores, resultando em múltiplas reinstalações de toda a aplicação. Richardson [2015] ainda afirma que, como resultado, a prática de implantação contínua torna-se consideravelmente complexa de ser executada.

Outra complicação associada às aplicações monolíticas apresentada por Richardson [2015] é que, dado que os módulos são executados no mesmo processo, uma falha em um único módulo pode resultar na queda de toda a aplicação, ou seja, de todos os seus módulos. Além disso, a introdução de novos frameworks torna-se desafiadora, pois a substituição de um framework existente por outro pode exigir a reescrita de grande parte do código que originalmente utilizava o framework anterior, especialmente quando há uma extensa base de código já implementada com o framework original.

Conforme afirmado por Villaça, Jr e Azevedo [2018], a proposta de solução para essas adversidades é adotar a arquitetura de microsserviços, uma abordagem implementada por diversas empresas notáveis, como Amazon, Netflix, The Guardian, entre outras. A essência dessa arquitetura consiste em fragmentar a aplicação em um conjunto de serviços menores interconectados. Cada microsserviço, por sua vez, tipicamente implementa um conjunto específico de funcionalidades de negócio, operando como uma mini-aplicação independente, conforme ilustrado na Figura 4.

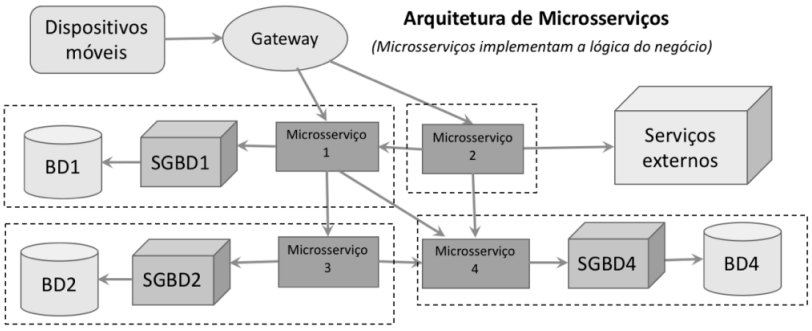

Figura 4 – Arquitetura empregando microsserviços.

Fonte: [VILLAçA; JR; AZEVEDO, 2018], adaptado originalmente de [RICHARDSON, 2015].

Villaça, Jr e Azevedo [2018] destaque que, os microsserviços interagem entre si, expondo e consumindo funcionalidades uns dos outros. Vale destacar que alguns microsserviços podem, inclusive, implementar interfaces web, possibilitando que sejam disponibilizadas como microsserviços independentes. Essa abordagem viabiliza a implantação de experiências distintas para usuários ou dispositivos específicos, assim como para casos de uso particulares.

Richardson [2015] assegura que, a arquitetura de microsserviços aborda a complexidade mediante a decomposição da aplicação monolítica em um conjunto de microsserviços. Cada microsserviço delimita-se claramente, proporcionando um nível de modularidade que, na prática, é desafiador de alcançar em uma aplicação monolítica. Como resultado, os serviços individuais tornam-se mais acessíveis para desenvolvimento, compreensão e manutenção. Equipes independentes operam em cada microsserviço, utilizando as tecnologias que julgam mais apropriadas, tanto em termos de software quanto de hardware, seguindo as diretrizes da organização a que pertencem. A implantação de cada microsserviço ocorre de maneira independente, facilitando a atualização de novas versões sem impactar a implantação de outros microsserviços.

É imperativo ressaltar que, segundo Fowler [2017], nem todas as aplicações monolíticas são necessariamente problemáticas ou enfrentam as mesmas questões mencionadas. No entanto, todas as características inerentes a uma aplicação concebida na arquitetura monolítica divergem dos princípios fundamentais de uma arquitetura de microsserviços, a qual prioriza a escalabilidade e a alta disponibilidade de suas aplicações.

Para este trabalho, adotou-se a perspectiva do NIST (*National Institute of Standards and Technology* — Instituto Nacional de Padrões e Tecnologia), conforme apresentado por Sousa, Moreira e Machado [2009]. O NIST descreve que o modelo de computação em nuvem é composto por cinco características essenciais, três modelos de serviço e quatro modelos de implantação.

De acordo com Sousa, Moreira e Machado [2009], as características essenciais da

computação em nuvem, consideradas premissas para microsserviços, incluem o *self-service* sob demanda, que permite ao usuário adquirir unilateralmente recursos computacionais conforme sua necessidade, sem a necessidade de interação humana com os provedores de serviços. A segunda premissa é o amplo acesso, onde recursos são disponibilizados através da rede e acessados por meio de mecanismos padronizados que permitem a utilização por diversas plataformas.

Pooling de recursos é retratado por Sousa, Moreira e Machado [2009] como a terceira premissa e linearmente rápida elasticidade e por fim, o serviço medido. Sousa, Moreira e Machado [2009] e Jacobs e Aulbach [2007] também definem que *pooling* de recursos trata-se de recursos computacionais do provedor organizados em um *pool* para servir múltiplos usuários usando um modelo multi-inquilino, com diferentes recursos físicos e virtuais, dinamicamente atribuídos e ajustados conforme a demanda dos usuários.

Quanto à rápida elasticidade, refere-se a recursos que podem ser adquiridos de forma rápida e elástica, em alguns casos automaticamente, caso haja a necessidade de escalar com o aumento da demanda, e finalmente, o serviço medido, seguindo as palavras de Sousa, Moreira e Machado [2009], Sousa, Moreira e Machado [2011]. O uso de recursos pode ser monitorado e controlado, possibilitando transparência para o provedor e o usuário do serviço utilizado. Para garantir a qualidade do serviço supramencionado, pode-se utilizar a abordagem baseada em acordo de nível de serviço.

3.2 Modelos de serviços

Susodito, Sousa, Moreira e Machado [2009] garantem que há três modelos de serviço, nomeados de SaaS (*Software as a Service* — Software como um Serviço), PaaS (*Plataform as a Service* — Plataforma como um Serviço) e IaaS (*Infrastructure as a Service* — Infraestrutura como um Serviço), podendo ser lida como uma hierarquia, demonstrada na Figura 5.

A Figura 6 exemplifica aplicações de cada categoria a ser apresentada, assim como uma suma comparação entre SaaS, PaaS e Iaas.

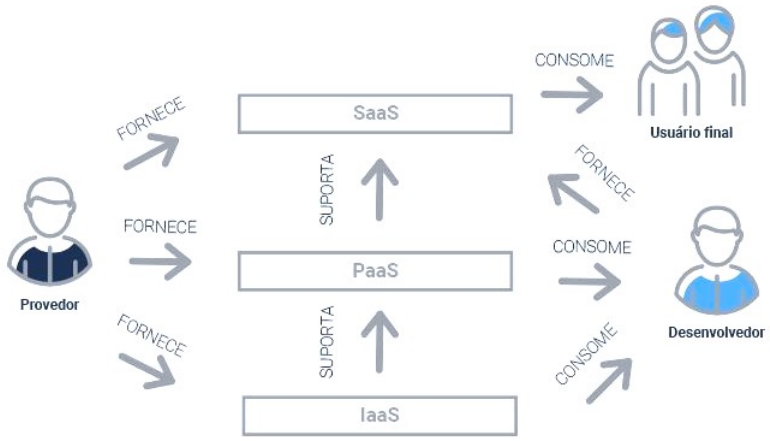

Figura 5 – Papéis na computação em nuvem.

Fonte: [CARVALHO, 2018].

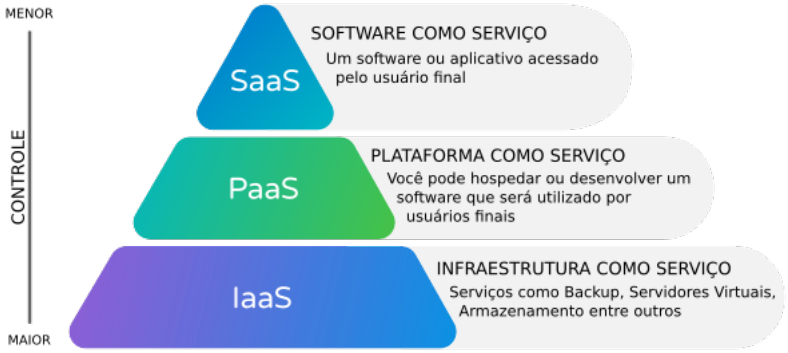

Figura 6 – Diferença de serviços de SaaS, PaaS e IaaS.

Fonte: [SILVA, 2024].

3.2.1 Software como um Serviço — SaaS

O SaaS refere-se a aplicativos de interesse comum disponibilizados na nuvem por provedores e acessados pelos clientes por meio de navegadores de internet, como descrito por Pineli e Duarte [2013].

Além disso, de acordo com Sousa, Moreira e Machado [2009], no modelo SaaS, os usuários não têm a responsabilidade de gerenciar ou controlar a infraestrutura subjacente, que engloba rede, servidores, sistemas operacionais, armazenamento, ou mesmo as características específicas da aplicação, exceto por configurações particulares. Essa abordagem permite que os desenvolvedores se concentrem na inovação em vez de na infraestrutura, resultando no desenvolvimento ágil de sistemas de software.

Ao ser hospedado na web, o software torna-se acessível para usuários em qualquer local e a qualquer momento, proporcionando uma integração mais eficiente entre unidades

de uma mesma empresa ou outros serviços de software, como destacado por Sousa, Moreira e Machado [2009], Cancian et al. [2009]. Dessa forma, a incorporação automática de novos recursos nos sistemas de software ocorre de maneira imperceptível para os usuários, tornando transparente a evolução e atualização dos sistemas.

O modelo SaaS contribui para a redução de custos, uma vez que elimina a necessidade de adquirir licenças de sistemas de software. Além disso, destaca-se a velocidade de implantação como um fator crucial. Com a dispensa de fornecimento de hardware, os clientes conseguem avaliar os produtos de forma mais rápida e ajustar facilmente suas rotinas de trabalho ao software escolhido, ainda seguindo Sousa, Moreira e Machado [2009], Cancian et al. [2009].

A utilização do SaaS possibilita que o cliente construa sua própria carteira de serviços de software de acordo com suas necessidades, podendo ser facilmente modificada para incluir novos serviços ou excluir aqueles anteriormente contratados. Essa flexibilidade é viabilizada pela natureza desacoplada e online das liberações. Esse cenário oferece diversas vantagens, permitindo que o cliente pague apenas pelo que realmente utiliza. Além disso, há a opção de experimentar um serviço por determinado período e, caso não atenda às expectativas, rescindir o contrato, uma vez que no modelo SaaS o cliente não adquire uma licença de software. Para o provedor, essa abordagem possibilita atender de maneira mais precisa às necessidades do cliente, como afirma Cancian et al. [2009].

"A negociação burocrática que envolve o uso, execução e acesso ao serviço entre cliente e provedor é regida por um contrato de nível de serviço" [CANCIAN et al., 2009]. Conforme brevemente apresentado por Kalinowski e Reinehr [2013], as solicitações podem ser categorizadas com base em classificações específicas, cada uma com seus respectivos prazos de solução, ou seja, o Acordo de Nível de Serviço (SLA), conforme ilustrado na Figura 7.

ACORDO DE NÍVEL DE SERVIÇO E MULTAS

Este acordo de nível de serviço é válido tanto para correções, quanto para mudanças e novos desenvolvimentos, solicitados a partir da data de entrada em vigor deste contrato.

Acordo

As solicitações terão os seguintes prazos para que sejam atendidas plenamente (sem apresentar defeitos):

- **Solicitações Urgentes**
  Prazo em Horas = (1,5 x esforço estimado) Horas.
  Prazo em Dias = (Prazo em Horas/8) Dias Úteis
- **Solicitações de Prioridade Alta**
  Prazo em Horas = (1,5 x esforço estimado) Horas.
  Prazo em Dias = (Prazo em Horas/8) + 1 Dias Úteis
- **Solicitações de Prioridade Média**
  Prazo em Horas = (1,5 x esforço estimado) Horas.
  Prazo em Dias = (Prazo em Horas/8) + 2 Dias Úteis|
- **Solicitações de Prioridade Baixa**
  Prazo em Horas = (1,5 x esforço estimado) Horas.
  Prazo em Dias = (Prazo em Horas/8) + 4 Dias Úteis

O Prazo em Horas será utilizado como limite do número de horas faturáveis.

O Prazo em Dias será utilizado para a aplicação de multas referentes a prazo. Sempre que este número for fracionário ele deverá ser arredondado para cima, ou seja, um prazo de 8,32 dias úteis deverá ser atendido dentro do nono dia útil.

Multas

As solicitações de mudança ou de novos desenvolvimentos que não forem atendidas dentro do acordo de nível de serviço serão automaticamente penalizadas em 30%.

As solicitações de correções, incluídas no valor mensal fixo, que não forem atendidas dentro do acordo de nível de serviço, por sua vez, serão automaticamente convertidas em multa correspondente a 30% do prazo em horas, considerando o valor hora padrão estabelecido neste contrato. Este valor será descontado do custo variável a ser faturado mensalmente.

Figura 7 – Acordo de Nível de Serviço.

Fonte: [KALINOWSKI; REINEHR, 2013].

3.2.2 Plataforma como um Serviço — PaaS

A PaaS, como Sousa, Moreira e Machado [2009] conceituam, oferece uma infraestrutura altamente integrada para implementação e teste de aplicações na nuvem. Nesse modelo, o usuário não gerencia ou controla a infraestrutura subjacente, que engloba rede, servidores, sistemas operacionais e armazenamento. No entanto, o usuário mantém controle sobre as aplicações implantadas e, possivelmente, as configurações dessas aplicações hospedadas na infraestrutura provida.

A PaaS disponibiliza um ambiente que inclui sistema operacional, linguagens de programação e ferramentas de desenvolvimento para as aplicações, facilitando a implementação de sistemas de software. Além disso, a PaaS oferece suporte a colaboração entre desenvolvedores, fornecendo ferramentas específicas para o desenvolvimento eficiente, afirma Sousa, Moreira e Machado [2009].

A diferença entre PaaS e SaaS é que, o SaaS hospeda apenas aplicativos completos em nuvem, enquanto o PaaS oferece uma plataforma de desenvolvimento para aplicativos em nuvem concluídos e em andamento. PaaS oferece um ambiente onde os desenvolvedores podem criar e implantar aplicativos sem a necessidade de saber quanto de memória e

quantos processadores seu aplicativo estará utilizando. O modelo PaaS oferece benefícios aos desenvolvedores em termos de ciclo de vida do software, desde o planejamento até o design, a construção do aplicativo, a implantação e a manutenção (tradução livre de: [RANI; RANJAN, 2014]).

Rani e Ranjan [2014] afirmam também que neste modelo, o consumidor cria o software utilizando ferramentas e/ou bibliotecas fornecidas pelo provedor. O consumidor também controla a implantação do software e as configurações. O provedor disponibiliza as redes, servidores, armazenamento e outros serviços necessários para hospedar a aplicação do consumidor. As ofertas de PaaS facilitam a implantação de aplicativos sem o custo e a complexidade de comprar e gerenciar o hardware e software subjacentes, além de provisionar capacidades de hospedagem. Existem vários tipos de fornecedores de PaaS; no entanto, todos oferecem hospedagem de aplicativos e um ambiente de implantação, juntamente com vários serviços integrados.

De maneira geral, Sousa, Moreira e Machado [2009] reiteram que, os desenvolvedores que optam pela PaaS têm acesso a ambientes escaláveis, embora estejam sujeitos a algumas restrições em relação ao tipo de software que podem desenvolver. Essas limitações abrangem desde as imposições do ambiente na concepção das aplicações até a necessidade de utilizar sistemas de gerenciamento de banco de dados do tipo chave-valor, em oposição aos SGBDs relacionais. No âmbito empresarial, a PaaS possibilita que os usuários empreguem serviços de terceiros, expandindo o uso do modelo de suporte, no qual os usuários se inscrevem para solicitar serviços de TI ou resolver problemas por meio da Web. Dessa maneira, é possível aprimorar a gestão do trabalho e as responsabilidades das equipes de TI nas empresas.

3.2.3 Infraestrutura como um Serviço — IaaS

A IaaS desempenha o papel de fornecer toda a infraestrutura necessária para suportar a PaaS e o SaaS, ressalta Sousa, Moreira e Machado [2009]. O principal propósito da IaaS é facilitar e tornar acessível à entrega de recursos essenciais de computação, como servidores, rede, armazenamento, entre outros, para construir um ambiente sob demanda, podendo incluir sistemas operacionais e aplicativos. A IaaS apresenta características distintivas, como uma interface única para a administração da infraestrutura, uma API para interação com hosts, switches, balanceadores, roteadores, e suporte para a incorporação de novos equipamentos de maneira simples e transparente.

De acordo com Rani e Ranjan [2014], o IaaS permite a acessibilidade da infraestrutura usando tecnologia da Internet, composta por servidor, armazenamento e outros dispositivos periféricos. Pode ser combinado com serviços gerenciados para suporte a sistemas operacionais e aplicativos. O modelo IaaS concentra-se em capacitar tecnologias,

oferecendo um serviço para obter um servidor virtual em poucos minutos e pagar apenas pelos recursos que utilizam. O consumidor pode utilizar diretamente os componentes da infraestrutura (armazenamento, *firewall*, rede, etc.).

O IaaS, como provedor de serviços, oferece um servidor virtual contendo uma ou mais CPUs, executando várias opções de sistema operacional, apresentando-se como uma oferta padronizada e altamente automatizada, na qual os recursos de computação, complementados por capacidades de armazenamento e rede, são de propriedade e hospedados por um provedor de serviços e oferecidos aos clientes sob demanda. Os clientes podem guarnecer essa infraestrutura por conta própria, usando uma interface gráfica baseada na web que serve como um console de gerenciamento de operações de TI para o ambiente geral. O acesso da API à infraestrutura também pode ser oferecido como opção, destaca Rani e Ranjan [2014].

De maneira concisa, Rani e Ranjan [2014] esclarecem que, o usuário não exerce a gestão nem o controle sobre a infraestrutura na nuvem, porém mantém autoridade sobre os sistemas operacionais, armazenamento e aplicativos implantados. Eventualmente, pode-se permitir a seleção de componentes de rede. O termo Infraestrutura como um Serviço refere-se a uma estrutura computacional fundamentada em técnicas de virtualização de recursos computacionais. Essa infraestrutura pode escalar dinamicamente, ajustando os recursos conforme as necessidades das aplicações. Do ponto de vista econômico e de otimização de recursos legados, em contraposição à aquisição de novos servidores e equipamentos de rede para a expansão de serviços, é possível aproveitar os recursos já disponíveis e adicionar novos servidores virtuais à infraestrutura de forma dinâmica, como Sousa, Moreira e Machado [2009] explicam.

4 API

4.1 Contexto

A API, que significa *Application Program Interface* (Interface de Programação de Aplicação), é a implementação de um conceito que permite a execução de funções ou serviços de um sistema padronizadamente, simplificando a complexidade envolvida na realização da tarefa. As APIs representam mecanismos que facilitam a comunicação entre dois componentes de software por meio de um conjunto de definições e protocolos, afirma Services [2023].

Muniz et al. [2023] assegura que, cada API oferece um contrato de comunicação que permite ao usuário executar as operações disponíveis. A concepção de uma interface que abstrai a complexidade e simplifica a integração entre os componentes de um sistema é um princípio fundamental no desenvolvimento de software. Esse paradigma de organização dos componentes promove a manutenibilidade, associada à facilidade de realizar correções e implementar evoluções no software.

Ao discutir sobre APIs, é comum abordar serviços web (*webservice*), conforme mencionado por Ferris e Farrell [2003]. Pandey [2006] acrescenta não haver uma definição universalmente aceita para serviços web, enquanto Pereira [2010] sugere que um denominador comum pode ser encontrado na comunicação entre entidades por meio de XML (*Extensible Markup Language* — Linguagem de Marcação Extensível), associado a protocolos para internet.

Uma API é uma tecnologia que facilita a troca de mensagens ou dados entre duas, ou mais aplicações. Inicialmente, as APIs eram tecnologias baseadas em simples chamadas de sub-rotinas. No entanto, ao longo do tempo, elas evoluíram para incluir novas funcionalidades, permitindo interoperabilidade e mudanças em sistemas, além de possibilitar o compartilhamento de dados entre diversas aplicações [PEREIRA, 2010].

Numerosas tecnologias possibilitam a realização dessa comunicação, de acordo com Tihomirovs e Grabis [2016], como RMI (*Remote Method Invocation* — Invocação Remota de Método), CORBA (*Common Object Request Broker Architecture* — Arquitetura Comum de Solicitação de Objeto Distribuído) ou DCOM (*Distributed Component Object Model* — Modelo de Objeto de Componente Distribuído).

Contudo, essas tecnologias frequentemente apresentam desafios significativos em termos de segurança e compatibilidade. Em vez disso, a tecnologia moderna costuma se apoiar em dois modelos emergentes: SOAP (*Simple Object Access Protocol* — Protocolo

Simples de Acesso a Objetos) e REST (*Representational State Transfer* — Transferência de Estado Representacional). Halili, Ramadani et al. [2018] ressaltam que, tanto o SOAP quanto o REST fundamentam-se na arquitetura orientada a serviços. Seu processo de desenvolvimento incorpora um elemento denominado Web API, que representa a interface para consumir o serviço.

Originária dessa semelhança, torna-se necessário esclarecer que, embora ambos os serviços sejam adequados para comunicações, há uma distinção sutil entre eles. Enquanto o SOAP segue um protocolo específico, o REST é um estilo arquitetônico para o desenvolvimento de serviços web [SONI; RANGA, 2019].

Frequentemente, um serviço web específico utiliza um formato determinado para a troca de dados. Por exemplo, ao se comunicar com REST, é provável que o serviço web utilize JSON (*JavaScript Object Notation* — Notação de Objeto Javascript) como formato de troca, enquanto o SOAP utiliza XML. No entanto, ele é realmente confrontado com a API REST, que interage com o servidor usando JSON. Ao utilizar a arquitetura RESTful com o formato de transferência de objeto JSON em vez de XML, a carga no canal de comunicação é reduzida, resultando em uma maior produtividade devido à significativa redução do tamanho do corpo da resposta do serviço web (tradução livre de: [MELNICHUK; KORNIENKO; BOYTSOVA, 2018]). Nas Figuras 8, 9 e 10 são ilustrados exemplos dos dois modelos de dados, JSON e XML.

```
{
    "email":"email@domain.com",
    "first_name":"Ivan",
    "last_name":"Ivanov"
}
```

Figura 8 – Representação do objeto “estudante” em JSON.

Fonte: [MELNICHUK; KORNIENKO; BOYTSOVA, 2018].

```
<student>
    <email>email@domain.com</email>
    <firstName>Ivan</firstName>
    <lastName>Ivanov</lastName>
</student>
```

Figura 9 – Primeira representação do objeto “estudante” em XML.

Fonte: [MELNICHUK; KORNIENKO; BOYTSOVA, 2018].

```
<student email="email@domain.com" firstName="Ivan" lastName="Ivanov">
</student>
```

Figura 10 – Segunda representação do objeto “estudante” em XML.

Fonte: [MELNICHUK; KORNIENKO; BOYTSOVA, 2018].

4.2 Tipos de APIs

4.2.1 SOAP

Silva et al. [2016] afirma que, o SOAP é um protocolo baseado em XML para mensagens e RPCs (*Remote Procedure Call* — Chamadas de Procedimentos Remotos). Em vez de definir um novo protocolo de transporte, o SOAP opera em protocolos de transferência existentes, como HTTP, SMTP e MQSeries. A especificação do SOAP define um modelo que dita como os destinatários devem processar as mensagens SOAP. O modelo de mensagem também inclui atores, que indicam quem deve processar a mensagem. Uma mensagem pode identificar atores que indicam uma série de intermediários que processam as partes da mensagem destinadas a eles e encaminham o restante.

Sendo uma tecnologia muito abrangente, SOAP funciona em várias plataformas, sido fornecida por diversos fabricantes e contando com o suporte de uma ampla gama de empresas. Assume-se dois tipos de documentos SOAP: o SOAP simples, o qual é uma mensagem XML pura, e o SOAP com anexos, que permite a transmissão de vários tipos de dados com uma mensagem SOAP (tradução livre de: [SINGH, 2004]).

Singh [2004] afirma que o SOAP é apresentada como uma tecnologia única para uso em computação distribuída, tendo algumas semelhanças com outros mecanismos, como RMI, mas também possui várias características distintas. Essas diferenças derivam principalmente do fato de que o SOAP é projetado como um protocolo simples, foi deliberadamente projetado para ser leve e, por esse motivo, não oferece suporte a recursos como a passagem de objetos por referência.

O SOAP também é projetado para ser extensível, e esses objetivos de simplicidade e extensibilidade significam que, o uso do SOAP não implica em nenhum modelo de programação específico, a semântica de uma implementação específica é extremamente flexível. Existem diversos modelos possíveis para a troca de mensagens, bem como mecanismos para a troca de dados de tipos definidos pela aplicação e para a realização de chamadas e respostas de procedimentos remotos (tradução livre de: [SINGH, 2004]).

As mensagens SOAP não precisam estar vinculadas a um protocolo específico, o HTTP é uma escolha óbvia para muitas situações, mas existem muitas outras possibilidades. Da mesma forma, o modelo para a troca de mensagens é essencialmente uma transmissão unidirecional entre dois pares, onde o remetente envia uma mensagem para o receptor. No entanto, esse modelo pode ser estendido, "solicitação-resposta" é uma possibilidade óbvia, assim como transmissões de um-para-muitos e transmissão ou recepção assíncrona através de algum mecanismo de fila (tradução livre de: [SINGH, 2004]).

Diniz e Silva [2021] ressaltam que SOAP, em si, é apenas um protocolo e não uma arquitetura completa de objetos distribuídos. Por outro lado, arquiteturas completas de

objetos distribuídos são projetadas em torno dos seus protocolos para maior eficiência.

Belqasmi et al. [2012] defende que, uma arquitetura de *web services* baseada em SOAP é composta de três componentes:

1. O provedor de serviços, responsável pela publicação do *web service*;
2. O registro do servidor, componente que documenta todas as operações disponibilizadas;
3. O consumidor, que faz a conexão com o serviço e o utiliza para realizar requisições.

"Uma mensagem SOAP consiste em um envelope, baseado em XML. Esse envelope tem um cabeçalho opcional e um corpo obrigatório. O cabeçalho contém os metadados da mensagem, e o corpo define os dados enviados pela mensagem" [SUNGKUR; DAIBOO, 2015].

4.2.2 REST

Tanto Belqasmi et al. [2012] quanto Kumari e Rath [2015] conceituam REST (*Representational State Transfer* — Transferência de Estado Representacional) como um estilo arquitetural baseado em uma arquitetura cliente-servidor. Ele é utilizado em conjunto com o HTTP (*Hypertext Transfer Protocol* — Protocolo de Transferência de Hipertexto). "*Web services* que atendem as restrições impostas pelo estilo REST são chamadas de RESTful APIs" [KUMARI; RATH, 2015].

Já Marques [2018] afirma que, o REST figura como uma das arquiteturas mais prevalentes para o desenvolvimento de serviços distribuídos, possibilitando uma implementação simplificada de APIs para a comunicação entre diversos componentes do sistema, notadamente entre o servidor e os distintos clientes. O REST proporciona um conjunto de diretrizes essenciais para o desenvolvimento de um serviço coeso, escalável e com alto desempenho. Além disso, é independente da plataforma e da linguagem utilizada.

Segundo Fielding [2000], este estilo de arquitetura baseia-se nos princípios básicos a seguir:

1. Cliente-Servidor;
2. *Stateless*;
3. *Cacheable*;
4. Interface uniformizada;
5. Sistema em camadas.

Onde, ainda seguindo Fielding [2000], REST deve separar a arquitetura e as responsabilidades em dois ambientes, cliente e servidor, tornando-se independente e consequentemente, mais escalável. O consumidor do serviço preocupa-se apenas com a interface, enquanto o fornecedor de serviço (servidor) é responsável por devolver uma resposta ao cliente através da execução de um pedido enviado pelo cliente.

Stateless, como afirma Marques [2018], garante que o servidor não armazene o estado do cliente, e cada pedido é tratado de forma independente, transmitindo apenas informações essenciais. Para otimizar o desempenho, as respostas do servidor são temporariamente armazenadas em cache, apresentando o conceito *cacheable*, evitando processamento redundante para pedidos semelhantes de vários clientes. No entanto, para garantir a confiabilidade dos dados, pode ser necessário implementar um cache de *gateway* que armazena e reutiliza respostas, reduzindo as interações diretas com o servidor.

A comunicação entre clientes e servidor segue regras claras, promovendo uniformidade e normalização, conforme o quarto princípio apresentado por Fielding [2000]. Alguns fundamentos incluem identificar recursos por URI, representação de ações por métodos HTTP, escolha de formatos de retorno como JSON ou XML, definir o formato de comunicação no Content Type e incluir informações metadata nos pedidos e respostas, como códigos HTTP e Content-Type. O objetivo é proporcionar ao cliente todas as informações necessárias para navegar e acessar os recursos da aplicação de maneira eficiente, como ressaltam Marques [2018], Fielding [2000].

Quatro operações principais definem a interface REST: criar, ler, atualizar e deletar, implementadas usando, respectivamente, *POST, GET, PUT* e *DELETE*. *POST* é utilizado para a criação de novos dados. *GET* é utilizado para a recuperação de dados. *PUT* é utilizado para a atualização de dados e *DELETE* é utilizado para deletar dados. Cada requisição REST contém toda a informação necessária para o servidor realizar o seu processamento, não havendo nenhuma dependência entre requisições diferentes, explica Fielding e Taylor [2002].

Referente ao sistema de camadas, Marques [2018] declara que a aplicação deve adotar uma arquitetura em camadas que permita fácil modificação, adição ou remoção. Cada camada processa informações entre cliente e servidor de maneira específica. O acesso direto do cliente ao servidor deve ocorrer por meio de um *middleware*, como um balanceador de carga, garantindo distribuição eficiente de pedidos e proporcionando uma estrutura flexível, melhorando desempenho, escalabilidade, simplicidade, flexibilidade, visibilidade, portabilidade, confiabilidade e segurança.

5 Gestão de TI

5.1 Contexto

Biancolino et al. [2011] afirma que, a problemática associada à mensuração do valor da TI no contexto empresarial torna-se consideravelmente mais complexa quando tecnologias específicas são examinadas em suas diversas dimensões e em diferentes estágios de sua evolução histórica. O ERP (*Enterprise Resource Planning System* — Sistema de Planejamento de Recursos Empresariais) é um exemplar dessas tecnologias. Esse tipo de aplicativo não apenas é intrinsecamente complexo em sua constituição tecnológica, mas também provoca inúmeros impactos em diversos processos de negócio das organizações, manifestando-se de maneiras distintas e em momentos diversos ao longo da cadeia informacional.

Biancolino et al. [2011] ainda ressalta que, a mudança da natureza do valor dos benefícios gerados pelos ERP's com o passar do tempo, fato este associado à sua constante evolução (provocadas pelo surgimento de novas versões, novas funcionalidades e novas tecnologias), fazem com que os sistemas ERP sejam precursores de contínuas mudanças no formato de gestão dos processos de negócios das empresas. A medida em que o valor dos ERP's tende a se dissipar com o tempo devido, não só às próprias mudanças notadas nos ERP's, mas também impulsionados por frequentes mudanças nos ambientes interno e externo das organizações que usam os aplicativos ERP's para aquisição e manutenção de vantagens competitivas associadas à inovação.

Segundo Mata, Fuerst e Barney [1995], a competência primordial para gerar uma vantagem competitiva sustentável reside na gestão capacitada em TI. As competências associadas a essa gestão incluem:

1. A capacidade dos gestores de TI em compreender as necessidades informacionais do negócio;

2. A habilidade da empresa em conciliar interesses e integrar capacidades de gestores, clientes e fornecedores para desenvolver soluções eficazes de TI;

3. A competência de coordenar as atividades de TI de modo a oferecer suporte a gestores internos e à cadeia de TI;

4. A mestria de antecipar as futuras necessidades dos gestores internos e demais atores na cadeia de TI.

"A gestão das atividades de TI foi examinada no conceito de maturidade de seus processos. Indicia-se três áreas de preocupação para os executivos de TI. Uma se refere à gestão das demandas, isto é, à gestão das solicitações das áreas de negócio. A segunda, à gestão da produção ou operações das atividades e projetos de TI. Por fim, a terceira, à gestão de ativos, incluindo base de dados e infraestrutura" [RODRIGUES; MACCARI; SIMõES, 2009].

Referente aos modelos estruturais de gestão da TI utilizados, Rodrigues, Maccari e Simões [2009] em sua dissertação apresenta o ITIL (*Information Technology Infrastructure Library* — Biblioteca de Infraestrutura de Tecnologia da Informação) como o modelo estrutural ou sistema de padronização de processos e atividades de TI mais adotado pelos executivos de sua pesquisa.

5.2 ITIL

"O gerenciamento estratégico contempla a implantação de modificações em aspectos da maneira como a empresa compete e sobrevive no mercado, controlando ações e comportamentos necessários para implantar as mudanças" [STEELE, 1989]. A ITIL (*Information Technology Infrastructure Library* — Biblioteca de Infraestrutura de Tecnologia da Informação) é introduzido como um modelo de referência, que contém melhores práticas para o gerenciamento de serviços de TI, organizadas sob a lógica do ciclo de vida do serviço, de acordo com Fernandes e Abreu [2008].

Ramalho [2022] assume que, há também disponíveis diferentes ferramentas úteis ao processo de gestão de TI, entre as quais se destacam as ferramentas de ITSM (*Information Technology Service Management* — Gerenciamento de Serviços de TI), as quais são responsáveis por suportar os processos da operação de TI, considerando as boas práticas propostas pela ITIL. Um exemplo básico e comum deste serviço é uma central de serviços de TI, ou o conhecido *Help Desk*, sendo sistemas que registram problemas enfrentados pelos usuários de TI, que possuem inteligência para automatizar processos de solicitações e gerenciam os ativos do departamento, entre outras aplicações.

Magalhães e Pinheiro [2007] retrata a posição do ITIL no englobo de TI, como mostrada na Figura 11 e Assis e Laurindo [2010] afirmam que os principais modelos de referência da Governança de TI são o COBIT (*Control Objectives for Information and Related Technology* — Objetivos de Controle para Informação e Tecnologias Relacionadas) e o padrão ISO/IEC 385000 (*Corporate Governance of Information Technology* — Governança Corporativa de Tecnologia da Informação), entre outros.

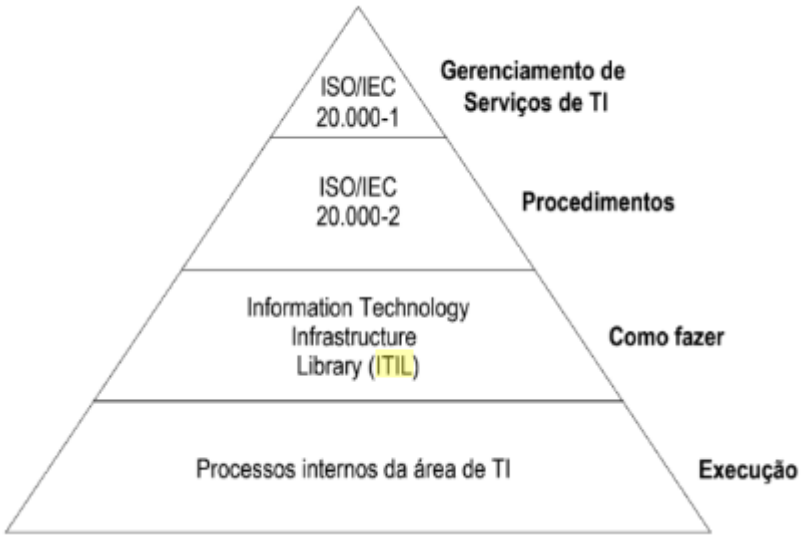

Figura 11 – Estratégia de implementação do Gerenciamento de Serviços de TI.

Fonte: [MAGALHãES; PINHEIRO, 2007].

Brito et al. [2020] assegura que, até o início da década de 90, a ITIL tornou-se ainda mais robusta e passou a ser adotada por empresas do setor privado, como a Microsoft e a IBM, que perceberam os benefícios que essas boas práticas estavam proporcionando ao departamento de TI. Desde seu lançamento, passou por quatro versões, sendo que a ITIL v3 evoluiu para v4 em resposta à quarta revolução industrial. O advento da globalização em massa, *smartphones* e o processamento em larga escala de dados são características marcantes dessa revolução. "Atualmente, a ITIL não se concentra apenas na infraestrutura de TI, mas abrange processos e serviços de TI na indústria" [RAMALHO, 2022].

Ramalho [2022] resume as características e história da ITIL em alguns dos seguintes pontos, podendo também ser visualizada na Figura 12:

- Estabelecida por usuários, fornecedores e consultores;
- Utilização comprovada como padrão de melhores práticas no gerenciamento de serviços de TI;
- ITIL é de domínio público, ou seja, não proprietária;
- Está em constante desenvolvimento;
- Oferece certificação para profissionais em geral;
- Reconhece que uma das maiores preocupações das empresas de TI é o gerenciamento dos serviços prestados;

- Não consente em substituir completamente métodos já existentes no gerenciamento de serviços;
- Através do ITSMF (*IT Service Management Forum* — Fórum de Gerenciamento de Serviços de TI) possui seu próprio fórum internacional de gerenciamento de serviços de TI;
- Não impõe regras rígidas, por depender de cada organização.

Figura 12 – Linha do tempo ITIL.

Fonte: [RAMALHO, 2022].

A Figura 13 apresenta os assuntos discutidos na ITIL e, em seguida, Ramalho [2022] define cada etapa do ciclo do Gerenciamento de Serviço.

- Estratégia de serviço: É a definição de estratégias para beneficiar a organização, incluindo a identificação de necessidades, demandas e recursos de infraestrutura necessários para atingir os objetivos estabelecidos. Essa fase alinhará os serviços de TI com as metas organizacionais;
- Desenho de serviço: Delineiam-se detalhes para a implementação, especificando os serviços, recursos e capacidades de TI necessários para atender aos requisitos do negócio. Este processo visa mitigar os riscos associados a uma implementação inadequada;
- Transição de serviços: Determina-se quais novos serviços devem ser implementados e quais devem ser descontinuados;
- Operação de Serviço: assegura-se que os serviços entregues atendam aos níveis pre-definidos. Esta fase envolve a gestão de incidentes e o controle de acessos aos serviços;
- Melhoria Contínua de Serviço: Refere-se ao processo constante de aprimoramento dos serviços, por meio da implementação de pequenas melhorias incrementais.

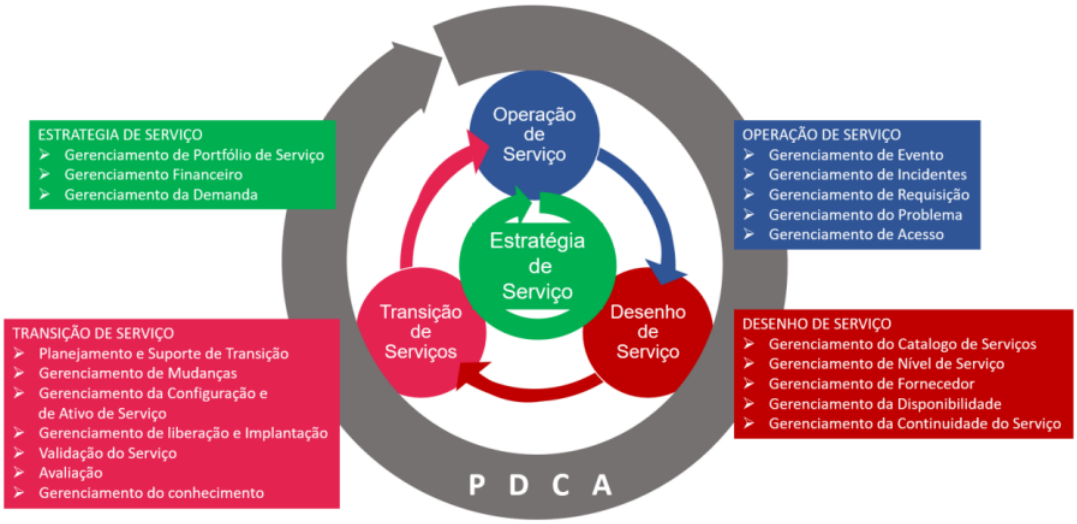

Figura 13 – Gerenciamento de serviços de tecnologia com qualidade.

Fonte: [RAMALHO, 2022].

5.3 ServiceNow

A ServiceNow (SNow) foi fundada em 2003 por Fred Luddy, sob o nome de Glidesoft Inc, complementando a afirmação de Costa [2022b], a ferramenta SNow fornece um modelo Saas, que se refere a uma padronização de entrega de software, onde o software é fornecido como um serviço online.

No entanto, Ramalho [2022] afirma que, apesar de o ServiceNow oferecer uma plataforma para o desenvolvimento de aplicativos e automação de processos, sua principal oferta é uma solução de gerenciamento de serviços na nuvem e outras soluções correlatas, entregues como um serviço online. Por conseguinte, é categorizado como SaaS, embora suas capacidades adicionais possam ser classificadas também como PaaS devido à sua natureza de plataforma para o desenvolvimento de aplicativos e automação.

A SNow propõe fluxos de trabalho para alavancar a produtividade das organizações, conforme o próprio site da empresa descreve, ServiceNow [2023a]. Os produtos oferecidos possuem foco em: serviços de *cloud*, serviços de gerenciamento de TI, gerenciamento de operações de TI, gerenciamento de negócios de TI, segurança, serviços do cliente, entrega de serviços de Recursos Humanos e desenvolvimento de aplicações, como afirma Ramalho [2022]. A ferramenta contempla todos os módulos do gerenciamento de operações conforme a ITIL. Esta ferramenta possui atualizações de versão semestral de modo a agregar novidades, melhorias e inovação nos processos de seus clientes.

Como ServiceNow [2023a] apresenta, a ferramenta ServiceNow é uma solução moderna, de fácil utilização e gestão de serviços em nuvem, permitindo que as organizações automatizem processos manuais e repetitivos, padronizem a entrega de serviços e foquem

em suas atividades principais. A plataforma ServiceNow oferece todas essas funcionalidades por meio de uma interface de usuário configurável baseada na web, construída sobre um modelo de dados flexível. Especializada em ITSM (*IT Service Management* — Gestão de Serviços de TI), ITOM (*IT Operations Management* — Gestão de Operações de TI) e ITBM (*IT Business Management* — Gestão de Negócios de TI), as aplicações que operam na plataforma Now utilizam um único sistema de registro e um modelo de dados comum para consolidar os processos empresariais das organizações.

Como afirma Silva [2022], as aplicações entregues pelo ServiceNow estão divididas em quatro fluxos de trabalho diferentes:

- Fluxos de trabalho em TI: integram tecnologia, gestão de riscos e operações de segurança, proporcionando serviços modernos e resilientes alinhados às prioridades do cliente, otimizando operações, alinhando investimentos e gerenciando eficazmente riscos, segurança e custos;
- Fluxos de trabalho dos empregados: possibilitam à organização, melhorar o engajamento e a produtividade dos funcionários, gerenciando um ambiente de trabalho seguro e eficiente, além de aumentar a eficiência operacional;
- Fluxo de trabalho do cliente: possibilitam a expansão dos serviços para além do centro de contato e das operações de escala, automatizando processos em todos os departamentos. Por meio desses fluxos, é viável capacitar os clientes com autosserviço personalizado, proporcionar visibilidade aos agentes para antecipar as necessidades dos clientes e otimizar os serviços de campo;
- Fluxos de trabalho do criador: permitem elaborar experiências intuitivas que conquistam os usuários. Isso envolve a ativação de aplicativos interempresariais e de baixo código, proporcionando serviços ágeis em uma escala ampla.

5.3.1 Arquitetura

Como relatado no manual ServiceNow [2023b], a SNow é uma plataforma empresarial baseada em nuvem, que segue uma arquitetura de múltiplas instâncias, o que significa que cada cliente obtém sua própria instância dedicada da plataforma, permitindo uma maior personalização e controle. A arquitetura básica segue os princípios de ITSM, sendo o aprimoramento da agilidade, otimização de custos, melhoria na gestão de incidentes de TI, simplificação da conformidade regulatória e elevação da qualidade de serviços a fim de construir as diversas camadas que formam a arquitetura total, podendo ser visualizada na Figura 14.

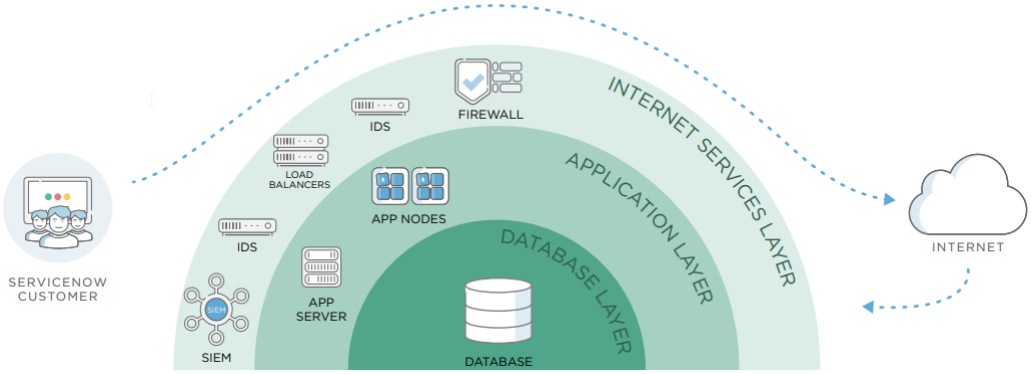

Figura 14 – Visão Geral da Arquitetura Lógica.

Fonte: [SERVICENOW, 2023b].

No manual da SNow, disposto por ServiceNow [2023b] e por Goldar [2023], é apresentado três camadas principais da arquitetura, sendo elas:

1. Camada de serviços de internet (camada proxy)

   Os clientes e serviços web se conectam à nuvem privada do ServiceNow via HTTPS usando TLS 1.2 para comunicação de e para uma instância da Plataforma Now. Todas as atividades interativas do usuário final são realizadas usando um navegador web padrão. Não há necessidade de os clientes instalarem qualquer software cliente em qualquer desktop, laptop, tablet ou smartphone para acessar suas instâncias da Plataforma Now. No entanto, para maior conveniência, o ServiceNow oferece aplicativos móveis nativos para iOS e Android.

   A camada proxy encaminha solicitações feitas pelos usuários finais dos clientes ou integrações para a instância do cliente relevante. Este primeiro nível da arquitetura do aplicativo inclui roteadores de rede, switches, balanceadores de carga, firewalls e sistemas de detecção de intrusões. Todos são implantados em uma base mínima de 2N (um sistema espelhado totalmente redundante com dois sistemas de distribuição independentes). A tradução de Identificadores de Recursos Universais (URIs) para endereços IP internos do ServiceNow é realizada nesta camada.

2. Camada de aplicação

   Neste segundo nível estão os servidores de aplicação em um segmento de rede discreto acessado apenas via camada proxy e não diretamente acessível da internet. Esses servidores hospedam nós de aplicação agrupados para cada instância do cliente. As instâncias da Plataforma Now são o ponto de término para todas as solicitações de entrada feitas pelos usuários finais dessas instâncias. As solicitações são recebidas e processadas pelos nós do aplicativo (incluindo serem escapadas ou codificadas conforme necessário) antes de passar para o serviço de banco de dados relevante na

camada do servidor de banco de dados. Servidores de aplicação e banco de dados têm proteções, como regras de firewall de negação padrão baseadas em host e ao nível de rede, em vigor para evitar que o tráfego de nível de host alcance a Internet.

3. Camada de banco de dados

   A terceira e última camada da arquitetura do ServiceNow compreende servidores de banco de dados, localizados em um segmento de rede isolado e não roteável pela internet. As solicitações de usuários finais ou integrações não podem acessar diretamente a camada de banco de dados, sendo emitidas exclusivamente a partir da instância de cada cliente. Cada instância é associada a um banco de dados individual, operado em servidores que executam múltiplos serviços de banco de dados discretos e segregados, assegurando a completa separação dos dados entre diferentes instâncias e clientes.

5.3.2 Camada de Banco de Dados do ServiceNow

Aprofundando, como afirma ServiceNow [2023b], a camada de banco de dados do ServiceNow consiste em servidores de banco de dados instalados em um segmento de rede discreto e não roteável pela internet. As solicitações dos usuários finais ou integrações não podem ser feitas diretamente à camada de banco de dados, sendo estas mediadas pela instância da Plataforma Now do cliente. Cada instância possui um único banco de dados, presente em um servidor que executa múltiplos serviços de banco de dados discretos e segregados. Esta configuração garante que não haja mistura de dados entre diferentes clientes: se um cliente possui quatro instâncias do ServiceNow, ele terá quatro bancos de dados e serviços de banco de dados completamente separados, cada um exclusivo para sua respectiva instância, podendo ser visualizado na Figura 15.

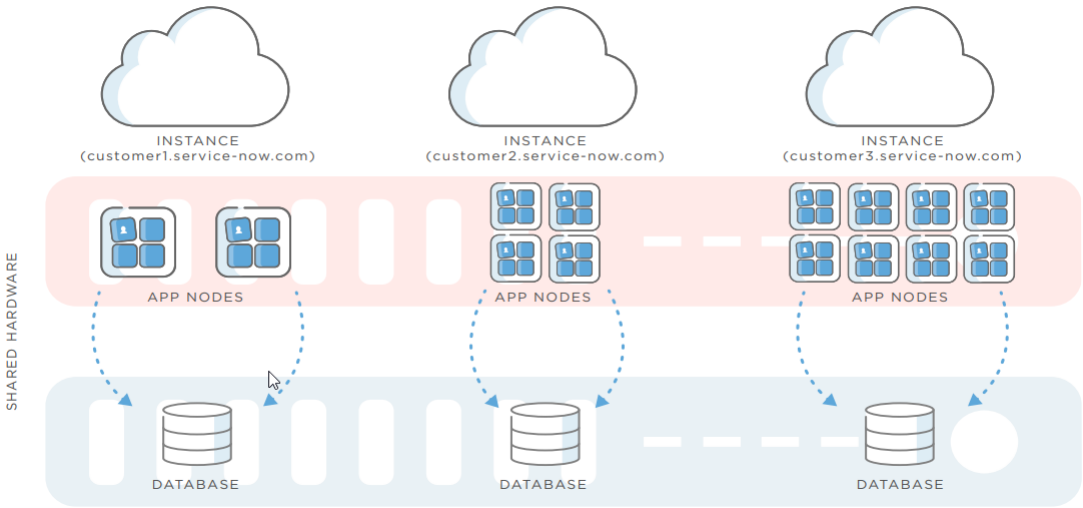

Figura 15 – Camada de Banco de Dados.

Fonte: [SERVICENOW, 2023b].

Esta abordagem de segregação total dos dados oferece vantagens significativas em termos de segurança e integridade. A arquitetura multi-instância do ServiceNow assegura que os dados de cada cliente são mantidos isolados, evitando problema comuns em ambientes *multi-tenant*, onde a separação de dados é feita por meio de marcação ou etiquetagem em um banco de dados compartilhado (tradução livre de: [SERVICENOW, 2023b]).

Além disso, esta estrutura permite uma administração precisa e eficiente dos dados, facilitando a manutenção, a realização de backups, e a recuperação de desastres sem impactar outras instâncias. A capacidade de realizar atualizações e manutenção de forma independente para cada instância contribui para a alta disponibilidade e resiliência do serviço (tradução livre de: [SERVICENOW, 2023b]).

Um benefício significativo da arquitetura da ServiceNow é a criação de uma fronteira lógica muito distinta entre os dados de cada cliente, o que não depende de abordagens comuns de separação de dados utilizadas por muitos provedores de SaaS, como a marcação de dados em bancos de dados compartilhados para identificar a qual cliente pertencem (tradução livre de: [SERVICENOW, 2023b]).

ServiceNow [2023b] declara ainda que, arquitetura única de múltiplas instâncias da ServiceNow possibilita um inventário altamente preciso da localização exata dos dados de um cliente específico a qualquer momento. Os clientes podem acessar essas informações diretamente por meio do portal de suporte ao cliente da ServiceNow a qualquer momento. Conhecer exatamente onde todos os dados hospedados de um cliente estão localizados também permite que a ServiceNow exclua de maneira confiável e segura todos os dados desse cliente, se necessário.

O modelo de múltiplas instâncias também facilita a transferência suave entre instâncias do cliente de um servidor de aplicativos para outro em um único centro de dados, assim como a rápida transição de instâncias de um centro de dados para outro dentro da mesma região, e por fim, a capacidade de realizar atualizações e manutenções individualmente sem impactar as instâncias de outros clientes, proporcionando uma disponibilidade excepcional das instâncias (tradução livre de: [SERVICENOW, 2023b]).

5.3.3 Comunicação do ServiceNow — MID Server

O Servidor de Gerenciamento, Instrumentação e Descoberta (MID Server) da ServiceNow é um componente opcional e gratuito oferecido pela ferramenta. Ele facilita a comunicação de dados entre as instâncias dos clientes e aplicativos externos, fontes de dados e serviços. Os MID Servers são utilizados pelos clientes em conjunto com suas instâncias para monitoramento de aplicativos e serviços empresariais, integração, orquestração e descoberta de ativos (tradução livre de: [SERVICENOW, 2023b]).

Conforme o material informativo ServiceNow [2023b]), o MID Server constitui uma aplicação Java disponibilizada aos clientes mediante um link de download dentro de sua instância pessoal ou corporativa. Sua instalação pode ser realizada pelo cliente em um sistema hospedeiro apropriado dentro de seu ambiente, podendo ter sistemas operacionais Windows ou Linux. Durante o processo de instalação, os MID Servers são emparelhados criptograficamente com uma instância específica e necessitam da aprovação por parte dos administradores do ServiceNow por parte dos clientes antes de serem habilitados para uso.

Em um intervalo definido pelo cliente, um MID Server inicia seguramente uma sessão de saída para a instância do cliente por meio do protocolo HTTPS usando TLS 1.2, procurando por atividades a serem executadas. A atividade é recuperada e executada, e qualquer saída ou dados resultantes são retornados à instância de origem. Essa abordagem de saída, ou *"pull"*, elimina a necessidade de permitir acesso de entrada através da perícia do cliente ou firewalls diretamente para a internet (tradução livre de: [SERVICENOW, 2023b]).

Para melhor compreensão, é possível visualizar as Figuras 16 e 17, onde a Figura 16 representa o diagrama técnico disponibilizado pelo manual da ServiceNow e a Figura 17 é uma exemplificação mais sucinta do fluxo de lógica de comunicação do MID Server com outros repositórios.

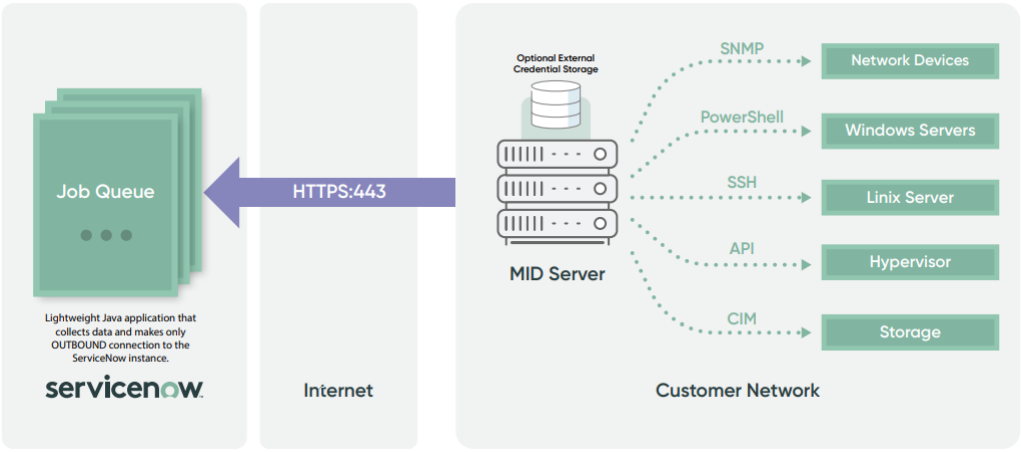

Figura 16 – Fluxo Técnico do MID Server.

Fonte: [SERVICENOW, 2023b].

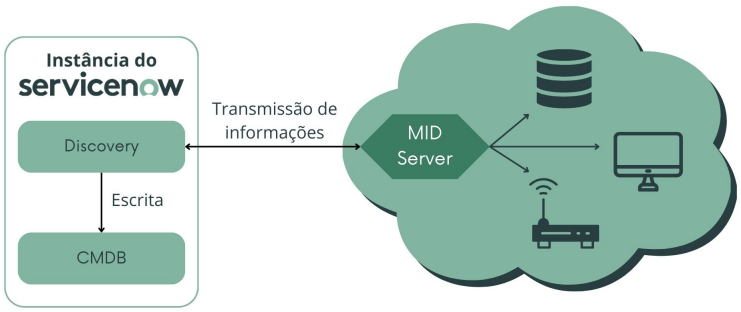

Figura 17 – Relação SNow, Mid Server e bancos diversos.

Fonte: Adaptado de [MIRANDA, 2022].

Tal como afirmado por ServiceNow [2023b], a SNow oferece suporte a serviços web (*webservices*) por meio dos protocolos SOAP e REST para fins de integração, sendo que todo o tráfego é submetido à criptografia TLS, a qual não será abordada nessa síntese. A segurança dos serviços web é implementada mediante a utilização combinada de autenticação básica em formato desafio/resposta e acesso ao nível de sistema, empregando segurança contextual. Adicionalmente, são atribuídos conjuntos específicos de funções *(roles)* destinadas aos usuários de serviços web. Para solicitações SOAP recebidas, há suporte para WS-Security 1.1, incluindo o perfil de token X.509 e o perfil de token de nome de usuário WSS. Neste contexto, "recebido" refere-se a solicitações direcionadas a um recurso de serviços web em uma instância do ServiceNow do cliente.

Aplicações como o *Discovery* são responsáveis pela descoberta e mapeamento dos serviços, infraestrutura de TI e as relações existentes entre os mesmos, como alega Miranda [2022]. O Discovery é um *plugin* disponibilizado pelo ServiceNow, que possui um conjunto de scripts como os *probes* e *sensors* escritos em JavaScript, e *patterns* escritos em Nebula Discovery Language.

Por fim, a CMDB (*Configuration Management Database* — Banco de Dados de Gerenciamento de Configuração) é um repositório de dados que guarda informação sobre os ativos de software e hardware, assim como os dispositivos existentes numa organização. É neste repositório que ficam registados os recursos de infraestrutura de TI do *cloud provider*. Os dados são registados na CMDB consoante a categoria à qual pertencem, por exemplo, máquinas virtuais, rede, entre outros.

6 Plano de Pesquisa

Esta seção abrange a estruturação completa do ensaio a ser realizado, bem como o escopo das atividades executadas e suas implicações.

6.1 Planejamento e Cronograma de Execução

O desenvolvimento deste estudo seguiu um fluxo contínuo de atividades previamente validadas. As atividades foram, por vezes, síncronas, mas particionadas em tarefas, seguindo o cronograma apresentado na Tabela 1. Como atividades principais, têm-se:

1. Revisão bibliográfica: Pesquisa e estudo de material necessário para conhecer todas as limitações do estudo a ser feito, além de recolher informações necessárias para embasar as melhores escolhas para descrição dos casos de uso;
2. Tutorial do TCC: Suma apresentação do fluxo do estudo para uma banca de no mínimo dois professores, com duração máxima de 10 minutos;
3. Desenvolvimento da monografia: A monografia será redigida ao decorrer do estudo e das tarefas previstas no cronograma;
4. Apresentação final: Realização dos materiais a serem utilizados na apresentação aos membros da banca, que terá dia e horários combinados, com duração máxima de 30 minutos;

Referente às tarefas específicas, têm-se:

1. Elicitação dos casos de uso: será definido o que os casos simulados deverão ter e o limite de sua cobertura. Embasado na revisão bibliográfica e nos requisitos dispostos pelos representantes da Itaipu Binacional nessa pesquisa;
2. Estudo das tecnologias: será definido como serão realizadas as medições e como as informações geradas serão colhidas, assim como o levante das limitações perante ao ServiceNow;
3. Implementação dos casos de uso: preparação e implementação dos casos de uso definidos;
4. Experimentos e Validação: realização dos testes com o ServiceNow, conforme casos previamente elicitados e validação dos dados coletados.

Atividades	**Período**							
	Out	Nov	Dez	Jan	Fev	Mar	Abr	Mai
ATV 1 - Revisão bibliográfica	•	•	•					
TASK 1 - Elicitação dos casos de uso		•	•	•	•	•		
TASK 2 - Estudo das tecnologias	•	•	•	•	•	•		
TASK 3 - Implementação dos casos de uso				•	•	•	•	
TASK 4 - Testes e Validação				•	•	•	•	•
ATV 2 - Tutorial do TCC				•	•	•		
ATV 3 - Desenvolvimento da monografia	•	•	•	•	•	•	•	•
ATV 4 - Apresentação final							•	•

Tabela 1 – Cronograma das Atividades.

6.2 Material e Método

Contribuindo valiosamente, a Itaipu Binacional fornecerá os dados aos quais os casos de uso deverão ser redigidos. Estes casos devem retratar ambientes onde o usuário final, por exemplo, faça uma consulta com um perfil de usuário não contido no banco em nuvem, mas sim, em um banco terceiro hospedado em um servidor local. Para prover as consultas aos sistemas internos, a Superintendência de Informatica da IB disponibilizará todo o ambiente de *hardware* e *software*, tais como os respectivos *webservices*, APIs e acessos às bases de dados, assim como as licenças necessárias.

Com isso, o estudo será realizado na plataforma em nuvem da ServiceNow, utilizando como intermediário o Mid Server, destacando que será instanciado um Mid Server dedicado somente a este experimento, onde será configurado os tipos de conexões e requisições objetos de estudo deste ensaio. Considerando o fato que a ServiceNow (SNow) é uma plataforma low/no-code, ou seja, sem codificação direta, há poucas circunstâncias onde será necessária codificação em linguagem de programação, mas aos momentos excepcionais, fica estritamente compulsório o uso da linguagem JavaScript, por meio de *scripts* que serão incluídos na plataforma.

Como o SNow dispõe da ferramenta de *"roles"*, ou seja, papéis que podem ser agregados aos perfis dos usuários para limitar os serviços disponíveis ao mesmo, os experimentos serão realizados com a *role* "admin". Ao todo, estão agregadas 71 *roles* no perfil a ser utilizado para acesso, mas a fim de sumarizar, a role "admin" é tratada como administrador de alta prioridade, permitindo todos os acessos necessários para implementação de todo o fluxo a ser desenvolvido neste ensaio.

Como todos os recursos computacionais estão disponíveis na nuvem e as máquinas dos usuários não necessitam ter altos recursos computacionais, diminuindo o custo na aquisição de máquinas [SOUSA; MOREIRA; MACHADO, 2009], a ficha técnica de *hardware* das máquinas a serem utilizadas durante esse estudo não tem relevância, mas a fim de esclarecimento, o computador utilizado para a execução dos testes possui as seguintes configurações:

- Sistema Operacional Windows 11 Home;
- Processador Intel(R) Core(TM) i7-1355U CPU @ 1.70 GHz;
- 12 GB de memória RAM.

Referente ao Mid Server, ele deve ser hospedado em um servidor, o qual a IB implementou em uma configuração disposta de:

- Sistema Operacional Windows Server 2019 Standard;
- Processador Intel(R) Xeon(R) CPU E5-2267 v2 @ 2.70GHz (4 processors);
- 12 GB de memória RAM.

Ao levante dos casos de uso, serão analisadas circunstâncias nas quais a quantia de dados a serem retornados diante uma consulta seja de densidade baixa, média e alta. Posteriormente, durante a etapa de elicitação dos casos de uso, serão definidas as quantias de dados a serem analisados para cada densidade.

Os instantes a serem implantados os observadores para coleta do tempo de resposta, posteriormente, serão elaborados de forma mais consistente na etapa de elaboração dos casos de uso. Entretanto, parte-se da premissa de duas possíveis situações:

- Tempo total da execução da busca pela API na rede interna;
- Tempo total da execução da busca pela API no ambiente do ServiceNow;

O ensaio segue um fluxo contínuo de testes e coleta de informações, podendo ser exemplificado na Figura 18, onde será implementado o método de integração ao mesmo banco e com o mesmo conjunto de dados, consultado pela rede interna e o consumo da API SOAP disponibilizada pela equipe da Itaipu para uso no ServiceNow. O tempo de resultado será colhido ponderando o método e o tempo de execução total, assim, o mesmo delimitará a aplicabilidade de integrações síncronas, ou seja, em tempo real. Derivado dessa análise, o principal resultado será o delineamento de melhores práticas de desenvolvimento para a integração do SNow com sistemas internos.

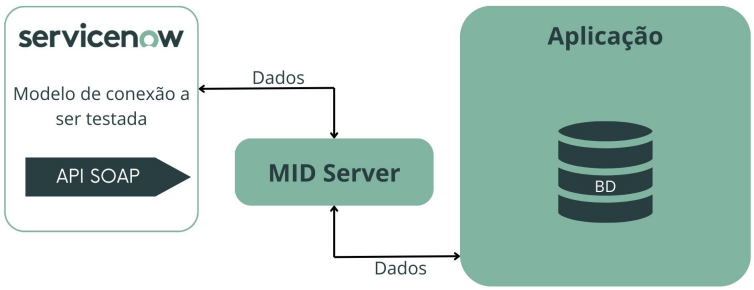

Figura 18 – Fluxo de testes. Fonte: Autora.

A IB dispõe de três instâncias da SNow, sendo elas: desenvolvimento, homologação e produção. Todos os ambientes possuem os mesmos dados, entretanto o ambiente de desenvolvimento refere-se à instância utilizada para elaboração de novos projetos, podendo ocorrer avarias nos dados, mas sem interferência nos dados dos outros ambientes.

Uma vez que o projeto é categorizado como pronto no ambiente de desenvolvimento, ele é enviado para o ambiente de homologação, onde serão realizadas sequências de testes e validação do produto, prevendo eventos inoportunos e *bugs*. Quando todas as contrariedades presentes no projeto em ambiente de homologação são revisadas e corrigidas, seu *status* é dito como pronto e, então, enviado para o ambiente final, a produção.

O ambiente de produção é o mais alto escalão dos ambientes, refere-se ao produto final utilizado pelos usuários, onde não deverão ocorrer complicações. A fim de não imiscuição nos dados e projetos presentes nos ambientes de homologação e produção, esse trabalho será realizado no ambiente de desenvolvimento da SNow da IB, utilizando busca pela API aos dados em produção da aplicação utilizada para estudo.

6.3 Critérios de Avaliação

Durante o perpassar desta pesquisa, as inquirições a serem solucionadas terão como métrica principal o aumento percentual do tempo total de execução, sendo esse o responsável pelo apontamento final do melhor caso a ser implementada em futuros planejamentos de novos serviços de conexões.

Ao coletar os tempos obtidos em cada cenário de consumo da API, serão cometidos a uma tabela onde poderá visualizar-se os instantes resultantes e com essas informações, será obtido o tempo mínimo, médio e máximo obtido para cada modelo, assim como o desvio padrão, para por fim, categorizar os objetos de estudo do melhor indicado para performance ao pior.

Como há diferentes tipos de consultas, há também a necessidade de manter o comparativo entre os cenários o mais simétrico possível. Por isso, é de suma importância uma definição equitativa, ou seja, ao comparar diferentes cenários, é importante garantir que as condições sejam equivalentes. Isso significa que os sistemas integrados devem operar sob circunstâncias semelhantes, evitando vantagens ou desvantagens injustas.

Todas essas informações resultarão na comparação final do método, podendo identificar os parâmetros necessários para categorizar e indicar as melhores práticas a serem abordadas pelos profissionais ao integrar uma base externa com o SNow.

7 Desenvolvimento

Este capítulo discute o estudo, juntamente com os principais detalhes do procedimento adotado para obtenção dos resultados. Para tanto, são delineadas as etapas fundamentais conforme apresentadas no capítulo anterior, abordando as tecnologias envolvidas, o processo de desenvolvimento e, por fim, os resultados alcançados.

7.1 Tecnologias Envolvidas

O objeto de estudo, como supramencionado, advém da baixa performance de uma tecnologia estar entre os principais fatores técnicos que podem pesar para que profissionais deixem de usar um *framework*, conforme afirma Moströmm e Ryrberg [2022]. Isso se dá pelo fato da performance interferir diretamente na relação do sistema com o usuário, e ter o risco de aumentar conforme a aplicação escala os fatores de tamanho e complexidade [PINTO; HOFFMANN; URIARTE, 2023].

Com base nessas considerações, foi observado que no ambiente do ServiceNow, a documentação vaga sobre os empregos das formas de implementação deixa à equipe de implementação a responsabilidade de escolher o modelo a ser adotado. No entanto, essa atribuição à equipe pode sobrecarregá-la, comprometendo a eficiência e eficácia operacional.

Apesar das inovações do ServiceNow como uma ferramenta para melhorar a eficiência operacional e oferecer suporte eficaz de TI, potenciais gargalos ainda podem surgir devido a escolhas mal feitas e integrações inadequadas. Para mitigar essa responsabilidade atribuída à equipe de integrações, foi considerado o cenário de importação de dados por meio da funcionalidade chamada *"Fix Script"* (script de correção). Uma ferramenta do ServiceNow utilizada para incluir scripts de correção que fazem alterações necessárias para manter a integridade dos dados ou a estabilidade do produto de um aplicativo [SERVICENOW, 2022b].

De acordo com ServiceNow [2022b], um *script* de correção é um código JavaScript executado no servidor após a instalação ou atualização de um aplicativo, mas também pode ser utilizado em contextos de *script* único para verificação e/ou obtenção de informações, como no cenário deste estudo.

Após a seleção do modelo a ser testado, foi decidido quais dados ele trataria. Em colaboração com representantes da Itaipu Binacional, optou-se por testar a integração do ServiceNow com o aplicativo terceiro VetorH, sendo este, apresentado por SOUZA et al. [2017], como uma a solução tecnológica que aplica conceitos modernos e inovadores no

gerenciamento estratégico de Recursos Humanos, orientado à gestão do Capital Humano.

Conseguinte a definição dos dados e modelo, realizou-se o experimento e posteriormente, a coleta dos dados. Nesta etapa foi utilizado o software Excel para centralizar os resultados obtidos, assim como para a análise estatística dos mesmos, visto que é um software de planilha eletrônica produzido pela Microsoft para computadores que utilizam o sistema operacional Microsoft Windows, além de computadores Macintosh da Apple Inc. e dispositivos móveis. Suas funcionalidades incluem uma interface intuitiva e capacitadas ferramentas de cálculo e de construção de gráficos que tornaram o Excel um dos mais populares aplicativos de computador até hoje, de acordo com Microsoft [2024].

Observa-se que, para atender a um dos cenários do estudo, foram realizadas consultas através do SOAP UI, uma ferramenta reconhecida como "uma líder mundial em testes funcionais de API" [SOAPUI, 2024a]. O SOAP UI é uma ferramenta de código aberto, construído em Java e usa Swing para a interface do usuário, que oferece suporte a diversos protocolos, incluindo SOAP, REST, HTTP, JMS, AMF e JDBC [SOAPUI, 2024b, 2024b]. Essa ferramenta foi selecionada devido à sua robustez e capacidade de testar diferentes tipos de APIs, garantindo a precisão e confiabilidade dos resultados obtidos durante o estudo. Além disso, a decisão foi influenciada pelo fato de que a Itaipu já a utiliza regularmente, oferecendo um suporte adicional e familiaridade com a ferramenta, facilitando a integração e o desenvolvimento das atividades de teste.

É fundamental ressaltar que todas as tecnologias empregadas neste estudo foram utilizadas conforme seus propósitos específicos, como descrito anteriormente. Para facilitar a compreensão da interconexão entre essas tecnologias, foi criado dois diagramas de fluxo ilustrativos para cada cenário, que representam visualmente o fluxo do estudo e suas interações. A Figura 19 demonstra o fluxo no SOAP UI e a Figura 20 o fluxo no ServiceNow. Esses diagramas visam oferecer uma visão mais clara e abrangente do contexto técnico do estudo, demonstrando como cada componente se relaciona e contribui para os objetivos e resultados da pesquisa.

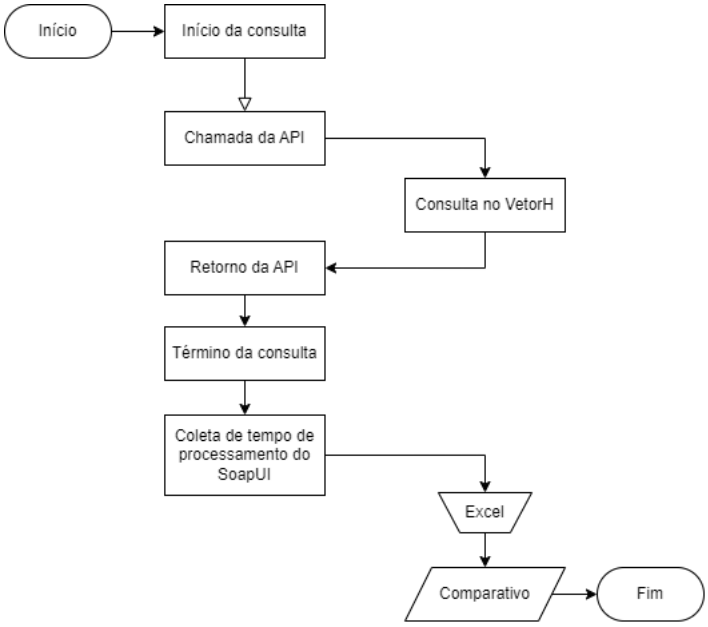

Figura 19 – Diagrama de fluxo do cenário SOAP UI.

Fonte: Autora.

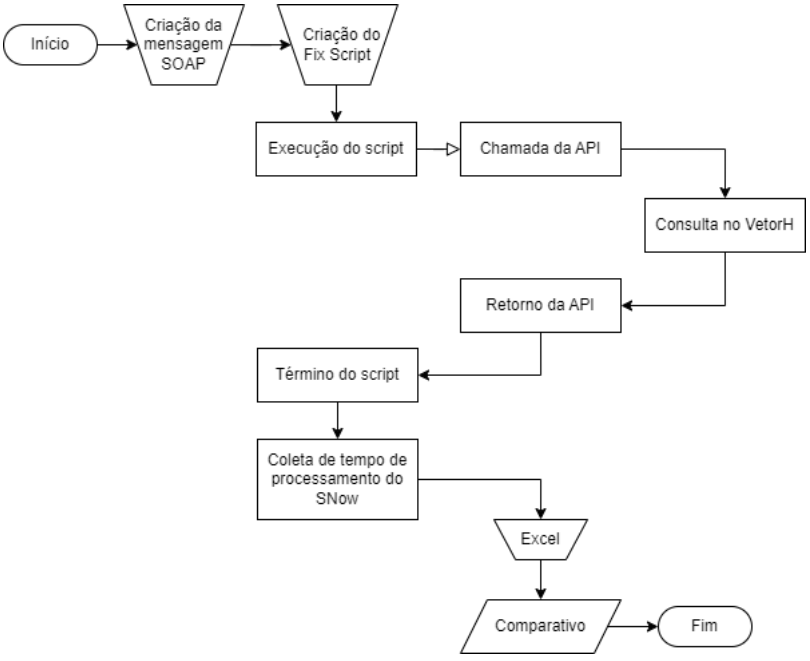

Figura 20 – Diagrama de fluxo do cenário ServiceNow.

Fonte: Autora.

7.2 Processo de Desenvolvimento

A escolha da integração entre VetorH e ServiceNow foi motivada pela suspeita de possíveis gargalos de desempenho identificados previamente. Para focar o escopo do estudo, a tabela R034FUN foi selecionada, a qual armazena dados de funcionários e

terceiros vinculados a contratos na Itaipu, provenientes do banco de dados Oracle da aplicação VetorH.

Para investigar as origens dos gargalos de desempenho, foi realizada a atividade 1 (TASK 1) conforme o cronograma apresentado na Tabela 1, que consistiu na elicitação dos casos de uso.

Três escopos de testes foram identificados para coletar dados de tempo visando futuras comparações:

1. Consulta de dados sem filtros específicos;
2. Consulta de um dado único através do registro de matrícula de um colaborador;
3. Consulta de dados através do registro de contrato, com limitação de retorno para 10, 50, 100 e 200 registros.

As delimitações evidenciadas acima foram escolhidas devido ao interesse em oferecer um comparativo do possível tempo significativo que o Mid Server agrega a uma consulta, sendo ela simples ou robusta. Estas consultas foram realizadas tanto no ServiceNow quanto pelo SOAP UI na rede interna em períodos de baixo tráfego, especificamente entre as 20h e 22h.

No quadro do SOAP UI, por ser uma ferramenta de teste de API, tem suas limitações reduzidas e sua mensagem de busca estruturada como na Figura 21. A mensagem é estruturada contendo os parâmetros de filtragem dos dados, onde são garantidos os três escopos de testes já mencionados.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:ser="http://services.senior.com.br">
   <soapenv:Header/>
   <soapenv:Body>
      <ser:dados>
         <user></user>
         <password></password>
         <encryption></encryption>
         <parameters>
            <!--Optional:-->
            <contrato></contrato>
            <!--Optional:-->
            <flowInstanceID></flowInstanceID>
            <!--Optional:-->
            <flowName></flowName>
            <!--Optional:-->
            <matricula></matricula>
            <!--Optional:-->
            <qregistros></qregistros>
         </parameters>
      </ser:dados>
   </soapenv:Body>
</soapenv:Envelope>
```

Figura 21 – Modelo de mensagem no cenário SOAP UI.

Fonte: Autora.

Os parâmetros relevantes utilizados foram o número do contrato, identificado na tag <`contrato`>; o número de matrícula, identificado na tag <`matricula`>; e por fim, a quantidade limitadora de registros a serem retornados, identificado na tag <`qregistros`>. Observa-se que, todos esses parâmetros são opcionais e quando não informados, é retornado todos os dados contidos na tabela, atendendo ao escopo de teste 1. Quanto ao escopo de teste 2, é afirmado quando informado o parâmetro matricula, e o escopo de teste 3 quando informados dois parâmetros, contrato e qregistros, de modo a garantir as particularidades desse escopo.

Quanto ao quadro do ServiceNow, por ser necessária a integração via Mid Server, foi criado um novo registro de Fix Script e SOAP Message. Uma mensagem SOAP é usada para enviar solicitações SOAP, cada registro de mensagens SOAP, armazena as informações necessárias para enviar solicitações SOAP, incluindo um ponto de extremidade para a solicitação, o formato necessário da solicitação como um arquivo de linguagem de descrição de serviços da web (WSDL), informações de autenticação e uma lista de funções que podem ser executadas contra o ponto de extremidade, como explicado por ServiceNow [2022a]. No caso desse estudo, criou-se um novo registro SOAP Message contendo a mesma estruturação do código descrito na Figura 21, mas com a inclusão dos identificadores para substituição de variáveis “nr_contrato”, “nr_matricula” e “qtd_registro”, mostrado na Figura 22, que foi utilizado no script criado no Fix Script.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:ser="http://services.senior.com.br">
   <soapenv:Header/>
   <soapenv:Body>
      <ser:dados>
         <user></user>
         <password></password>
         <encryption></encryption>
         <parameters>
            <!--Optional:-->
            <contrato>${nr_contrato}</contrato>
            <!--Optional:-->
            <flowInstanceID></flowInstanceID>
            <!--Optional:-->
            <flowName></flowName>
            <!--Optional:-->
            <matricula>${nr_matricula}</matricula>
            <!--Optional:-->
            <qregistros>${qtd_registro}</qregistros>
         </parameters>
      </ser:dados>
   </soapenv:Body>
</soapenv:Envelope>
```

Figura 22 – Modelo de mensagem no cenário do ServiceNow.

Fonte: Autora.

A substituição de variáveis nas mensagens SOAP do ServiceNow permite a inserção de valores dinâmicos em suas mensagens SOAP. Em vez de codificar valores específicos em suas mensagens, pode-se usar variáveis substituídas por valores reais quando a mensagem é enviada. Para usar a substituição de variáveis, é necessário definir os valores das variáveis no script antes de enviar a mensagem. A Figura 22 mostra no script como é

feita essa substituição. Por fim, são executados os scripts no ambiente do SNow e obtidos os resultados, para então iniciar o processo de comparação para fins de avaliação.

Em ambos os quadros, as consultas foram realizadas 10 vezes para obter-se dados confiáveis de média e desvio padrão, necessários para uma análise coesa, além de outros indicativos.

7.3 Resultados

Após a conclusão do desenvolvimento, iniciou-se a coleta dos tempos obtidos tanto no SOAP UI quanto no ServiceNow. Durante a execução dos três escopos de testes estipulados, registrou-se no SOAP UI o tempo de processamento em milissegundos do webservice, correspondente ao tempo total de cada consulta. No ServiceNow, foram registrados o tempo total de processamento de cada consulta.

Para ser estabelecido a significância do volume de dados, foi definido na etapa de elicitação dos casos de uso, que os testes seriam cometidos as consultas que tivessem retornos classificadas como muito baixo, baixo, médio, alto, muito alto e extremamente alto. Definido o retorno de 1 dado para muito baixo, retorno de 10 dados para baixo, 50 para médio, 100 para alto, 200 para muito alto e acima de 1.500 para extremamente alto.

Para cada conjunto de dados obtido, calculou-se a média; o desvio padrão, uma medida de variabilidade em um conjunto de dados; e o coeficiente de variação (CV), definido como a relação entre o desvio padrão e a média, frequentemente expresso em porcentagem [NETO, 2002], podendo ser visualizado na Expressão 7.1.

$$\mathrm{CV} = \frac{\mathrm{DP}}{\mathrm{Média}} \times 100 \tag{7.1}$$

Assim, foi calculado também a diferença do coeficiente de variação nas consultas que detinham as mesmas estruturações, ou seja, as consultas que alterassem apenas a quantidade de registros obtidos.

Os tempos de processamento (em milissegundos) do *webservice* obtidos no SOAP UI para consultas sem parâmetros, utilizando o parâmetro de matrícula (selecionado aleatoriamente pela autora) e com o retorno de dados limitado a 10, 50, 100 e 200 registros, estão detalhados nas figuras correspondentes, sendo estas as Figuras 23, 24, 25, 26, 27 e 28, respectivamente.

Sem Parâmetros	
Experimento	**Tempo (ms)**
0	7232
1	7211
2	7265
3	7434
4	7510
5	7311
6	7479
7	7732
8	7472
9	7794
Média (ms)	**7444**
DP	**153,4**
CV (%)	**2,06**

Figura 23 – Tempos de processamento (em ms) do webservice obtidos no SOAP UI para consulta sem parâmetros.

Fonte: Autora.

Matrícula	
Experimento	**Tempo (ms)**
0	165
1	175
2	177
3	169
4	179
5	177
6	174
7	170
8	173
9	160
Média (ms)	**171,9**
DP	**4,72**
CV (%)	**2,75**

Figura 24 – Tempos de processamento (em ms) do webservice obtidos no SOAP UI para consulta por matrícula.

Fonte: Autora.

Contrato 10 registros	
Experimento	**Tempo (ms)**
0	262
1	250
2	200
3	243
4	244
5	229
6	244
7	280
8	274
9	245
Média (ms)	**247,1**
DP	**15,52**
CV (%)	**6,28**

Figura 25 – Tempos de processamento (em ms) do webservice obtidos no SOAP UI para consulta por contrato limitado a 10 registros.

Fonte: Autora.

Contrato 50 registros	
Experimento	**Tempo (ms)**
0	396
1	401
2	426
3	368
4	424
5	389
6	384
7	424
8	456
9	423
Média (ms)	**409,1**
DP	**21,5**
CV (%)	**5,26**

Figura 26 – Tempos de processamento (em ms) do webservice obtidos no SOAP UI para consulta por contrato limitado a 50 registros.

Fonte: Autora.

Contrato 100 registros	
Experimento	**Tempo (ms)**
0	695
1	755
2	778
3	771
4	798
5	691
6	792
7	778
8	793
9	690
Média (ms)	**754,1**
DP	**37,26**
CV (%)	**4,94**

Figura 27 – Tempos de processamento (em ms) do webservice obtidos no SOAP UI para consulta por contrato limitado a 100 registros.

Fonte: Autora.

Contrato 200 registros	
Experimento	**Tempo (ms)**
0	1396
1	1322
2	1345
3	1378
4	1307
5	1332
6	1277
7	1300
8	1410
9	1280
Média (ms)	**1334,7**
DP	**38,04**
CV (%)	**2,85**

Figura 28 – Tempos de processamento (em ms) do webservice obtidos no SOAP UI para consulta por contrato limitado a 200 registros.

Fonte: Autora.

Observa-se que o coeficiente de variação do tempo altera entre as consultas, refletindo diferentes níveis de variação nos tempos de processamento. Por exemplo, para

consultas com um número crescente de registros, o coeficiente de variação tende a diminuir na consulta por contrato e limitador de registros, indicando uma maior consistência nos tempos de processamento à medida que o volume de dados aumenta, podendo ser verificar esta afirmação ao observar a Figura 29.

Consulta	Média (ms)	DP	CV (%)	Variação CV em relação ao anterior
Sem Parâmetros	7444	153,4	2,06	*
Matrícula	171,9	4,72	2,75	*
Contrato 10 registros	247,1	15,52	6,28	*
Contrato 50 registros	409,1	21,5	5,26	**-16%**
Contrato 100 registros	754,1	37,26	4,94	**-6%**
Contrato 200 registros	1334,7	38,04	2,85	**-58%**

Figura 29 – Comparativo entre as consultas realizadas pelo SOAP UI.

Fonte: Autora.

Na sequência, foram realizados os testes no ambiente do ServiceNow, abrangendo a medição do tempo total de execução, englobando desde a instanciação da mensagem SOAP até o término do script, este agregado ao tempo de processamento do Mid Server. Isso se deve ao fato de que, devido a algumas restrições, não foi possível obter o tempo isolado do Mid Server, uma vez que o acesso ao servidor onde está implementado é restrito à equipe da Itaipu Binacional. Esse obstáculo delimitou um limite para o ensaio, obrigando a análise apenas do tempo total de execução obtido do script.

Da mesma forma que na coleta realizada no SOAP UI, durante a execução dos três escopos de testes estipulados, registrou-se no ServiceNow o tempo de processamento em milissegundos do script criado, correspondente ao tempo total de cada consulta com a inclusão do tempo de processamento do Mid Server.

Para cada conjunto de dados obtido, também foram calculadas a média, o desvio padrão e o CV (Coeficiente de Variação), assim como sua variação nas consultas que possuíam as mesmas estruturações, variando apenas a quantidade de registros obtidos.

Os tempos de processamento (em milissegundos) do script para consultas sem parâmetros, utilizando o parâmetro de matrícula (selecionado aleatoriamente pela autora) e com o retorno de dados limitado a 10, 50, 100 e 200 registros, estão detalhados nas figuras correspondentes, sendo estas as Figuras 30, 31, 32, 33, 34 e 35, respectivamente.

Sem Parâmetros	
Experimento	**Tempo (ms)**
0	17761
1	17603
2	15820
3	16188
4	16168
5	17049
6	18675
7	16528
8	15539
9	14369
Média (ms)	**16570**
DP	**961,6**
CV (%)	**5,80**

Figura 30 – Tempos de processamento (em ms) do script no SNow para consulta sem parâmetros.

Fonte: Autora.

Matrícula	
Experimento	**Tempo (ms)**
0	2538
1	1034
2	3034
3	1803
4	1030
5	1041
6	1036
7	3691
8	1030
9	1034
Média (ms)	**1727,1**
DP	**831,52**
CV (%)	**48,15**

Figura 31 – Tempos de processamento (em ms) do script no SNow para consulta por matrícula.

Fonte: Autora.

Contrato 10 registros	
Experimento	**Tempo (ms)**
0	3709
1	1069
2	1063
3	1066
4	1066
5	3269
6	2069
7	2113
8	5781
9	2081
Média (ms)	**2328,6**
DP	**1154,64**
CV (%)	**49,59**

Figura 32 – Tempos de processamento (em ms) do script no SNow para consulta por contrato limitado a 10 registros.

Fonte: Autora.

Contrato 50 registros	
Experimento	**Tempo (ms)**
0	2905
1	2209
2	3228
3	2233
4	1201
5	4973
6	1203
7	2193
8	4992
9	2210
Média (ms)	**2734,7**
DP	**1031,84**
CV (%)	**37,73**

Figura 33 – Tempos de processamento (em ms) do script no SNow para consulta por contrato limitado a 50 registros.

Fonte: Autora.

Contrato 100 registros	
Experimento	**Tempo (ms)**
0	4802
1	2298
2	2992
3	2371
4	2342
5	2378
6	4100
7	2353
8	3070
9	3431
Média (ms)	**3013,7**
DP	**669,64**
CV (%)	**22,22**

Figura 34 – Tempos de processamento (em ms) do script no SNow para consulta por contrato limitado a 100 registros.

Fonte: Autora.

Contrato 200 registros	
Experimento	**Tempo (ms)**
0	5213
1	2675
2	5194
3	2666
4	5330
5	2653
6	5394
7	2725
8	5318
9	2736
Média (ms)	**3990,4**
DP	**1299,4**
CV (%)	**32,56**

Figura 35 – Tempos de processamento (em ms) do script no SNow para consulta por contrato limitado a 200 registros.

Fonte: Autora.

Similarmente ao observado no SOAP UI, no escopo do ServiceNow também foi constatada variação no coeficiente de variação do tempo entre as consultas, refletindo diferentes níveis de variação nos tempos de processamento. No entanto, ao contrário do primeiro cenário, observou-se uma variação nos tempos de processamento em um tipo de consulta no escopo do ServiceNow, podendo ser interpretada como uma oscilação comum sem real significância, já que se observado mais atentamente, o padrão apresentado no primeiro cenário ainda é seguido.

No contexto do ServiceNow, um coeficiente de variação elevado para um tipo específico de consulta pode indicar que o tempo de processamento para essa consulta é inconsistente e pode variar amplamente. Essa variação pode ser atribuída a vários fatores, incluindo a carga do servidor no momento da consulta, a complexidade da consulta,

a modelagem inadequada da API utilizada, entre outros. Uma forma de investigar profundamente o agente causador seria a constatação do tempo atrelado ao Mid Server, visto a indisponibilidade dessa informação, a motivação ficará implícita ao mesmo, já que se sabe que sua utilização agrega de fato algum custo de tempo junto ao tempo restante do processo.

Por exemplo, ao analisar consultas com um número crescente de registros, observou-se que o coeficiente de variação tende a diminuir nas consultas sem parâmetros, ou seja, consulta com retorno de volume de dados extremamente alto. Já nas consultas por contrato com limitador de registros, o coeficiente de variação apresentou uma instabilidade singular, já que se for observado atentamente, no contexto de todos os volumes, o padrão é mantido. Essa tendência pode ser verificada ao observar a Figura 36.

Consulta	Média (ms)	DP	CV (%)	Variação CV em relação ao anterior
Sem Parâmetros	16570	961,6	5,80	*
Matrícula	1727,1	831,52	48,15	*
Contrato 10 registros	2328,6	1154,64	49,59	*
Contrato 50 registros	2734,7	1031,84	37,73	**-24%**
Contrato 100 registros	3013,7	669,64	22,22	**-41%**
Contrato 200 registros	3990,4	1299,4	32,56	**+46%**

Figura 36 – Comparativo entre as consultas realizadas pelo script no ServiceNow.

Fonte: Autora.

Inicialmente, nota-se um aumento de desempenho em relação ao coeficiente de variação na consulta de 50 registros em comparação com a consulta de 10 registros, com uma redução de aproximadamente 24% no coeficiente de variação entre as duas consultas. Essa melhoria persiste quando comparadas as consultas de 50 e 100 registros, com o coeficiente de variação diminuindo em 41%.

No entanto, ao observar as consultas de 100 e 200 registros, ocorre uma inversão na tendência. Esperava-se um maior ganho de desempenho na consulta com 200 registros em relação à de 100 registros, porém, ela apresenta um aumento de 46% no coeficiente de variação, contrariando as expectativas em comparação realizada no SOAP UI (Figura 29). Todavia, se observadas a consulta sem parâmetro e a consulta com retorno de 200 registros, o decaimento do CV ainda é constatado.

A análise dos dados, conforme representado na Figura 37, permite destacar o aumento percentual entre o ponto inicial, caracterizado pelo escopo do webservice na rede interna, simbolizado como SOAP UI, e o ponto final, correspondente ao escopo do ServiceNow. Este valor de aumento percentual é calculado como a diferença entre a média de tempos de consulta pela rede interna e a média de tempos de consulta no ServiceNow,

expressa como uma fração do valor inicial. Além disso, essa metodologia é aplicada também para o cálculo da diferença percentual no desvio padrão entre os dois escopos, substituindo-se as médias pelos respectivos desvios padrão.

	SOAP UI			ServiceNow						
Consulta	**Média (ms)**	**DP**	**CV (%)**	**Média (ms)**	**DP**	**CV (%)**	**Diferença da média (ms)**	**Aumento % da média**	**Diferença DP**	**Aumento % do DP**
Sem Parâmetros	7444	153,4	2,06	16570	961,6	5,80	9126,0	123%	808,2	527%
Matrícula	171,9	4,72	2,75	1727,1	831,52	48,15	1555,2	905%	826,8	17517%
Contrato 10 registros	247,1	15,52	6,28	2328,6	1154,64	49,59	2081,5	842%	1139,1	7340%
Contrato 50 registros	409,1	21,5	5,26	2734,7	1031,84	37,73	2325,6	568%	1010,3	4699%
Contrato 100 registros	754,1	37,26	4,94	3013,7	669,64	22,22	2259,6	300%	632,4	1697%
Contrato 200 registros	1334,7	38,04	2,85	3990,4	1299,4	32,56	2655,7	199%	1261,4	3316%

Figura 37 – Comparativo entre os aumentos percentuais no escopo do webservice e no escopo do ServiceNow.

Fonte: Autora.

O cálculo do aumento percentual segue a abordagem descrita por Paraíba [2015], que envolve a diferença entre dois valores, inicial e final, dividida pelo valor inicial. No contexto deste estudo, a fórmula adotada é visualizada na Expressão 7.2.

$$\text{Aumento Percentual} = \frac{\text{Média}_{\text{SNow}} - \text{Média}_{\text{SoapUI}}}{\text{Média}_{\text{SoapUI}}} \times 100 \tag{7.2}$$

Observa-se que essa mesma fórmula é utilizada para calcular o aumento percentual do desvio padrão, substituindo-se os valores de média pelos desvios padrão, conforme Expressão 7.3.

$$\text{Aumento Percentual} = \frac{\text{DP}_{\text{SNow}} - \text{DP}_{\text{SoapUI}}}{\text{DP}_{\text{SoapUI}}} \times 100 \tag{7.3}$$

A Figura 37 revela a diferença de média para cada consulta, variando entre 1.555,2 ms e 9.126ms, enquanto o desvio padrão oscila entre 632,4ms e 1.261,4ms. Os resultados indicam um aumento percentual de 123% no tempo médio ao utilizar o ServiceNow em comparação com o SOAP UI na consulta sem parâmetros. O aumento percentual do desvio padrão é ainda mais significativo, alcançando 527%.

Para a consulta de matrícula, observa-se o maior aumento percentual calculado no tempo médio ao utilizar o ServiceNow em comparação com o SOAP UI, atingindo 905%. O desvio padrão também apresenta um aumento percentual substancial de 17.517%. Esses resultados indicam um aumento expressivo tanto no tempo médio quanto no desvio padrão ao utilizar o ServiceNow em comparação com o SOAP UI, na consulta de um dado único.

No caso da consulta de contrato limitada a 10 registros, observa-se um aumento considerável, embora menor que o anterior, de 842% no tempo médio para o ServiceNow

em relação ao SOAP UI. O desvio padrão também apresenta um aumento percentual expressivo de 7.340%.

Para consultas com contrato limitado a 50 registros, o tempo médio no ServiceNow aumenta em 568%, acompanhado por um aumento correspondente no desvio padrão de aproximadamente, 4.699%.

Na consulta de contrato limitada a 100 registros, observa-se um aumento de cerca de 300% no tempo médio ao usar o ServiceNow em comparação com o SOAP UI, enquanto o desvio padrão aumenta em torno de 1.697%.

Por fim, na consulta de contrato limitada a 200 registros, há um aumento de cerca de 199% no tempo médio ao utilizar o ServiceNow em comparação com o SOAP UI. Embora ainda seja menor que o aumento observado na consulta anterior, nota-se um aumento significativo no desvio padrão, atingindo aproximadamente 3.316%.

Ao comparar consultas sem parâmetros e consultas de 200 registros, onde a diferença percentual mantém-se entre 100% e 200%. Essa tendência é ainda mais acentuada ao comparar consultas com volume de retornos de dados distintos, como a busca por contrato limitada a 10 elementos e a limitada a 200, onde o aumento percentual diminui drasticamente de 842% para 199%.

Essa análise é complementada pela observação dos dados de aumento percentual do desvio padrão. Na consulta de baixo volume, como exemplificado na busca por 10 registros, o aumento percentual do desvio padrão é de 7.340%, enquanto na consulta de alto volume, como na busca por 200 registros, essa porcentagem diminui para 3.316%.

É possível constatar esse padrão também ao comparar consultas sem parâmetros, que retorna todos os dados da tabela, e consultas com retornos únicos. Apresentando um aumento percentual da média de 123%, a consulta sem parâmetro, contrastada com a consulta que retorna um único registro, que enuncia um aumento de 905%. Esses resultados são acompanhados por aumentos correspondentes nos desvios padrão, de 527% para a consulta sem parâmetros e 17.517% para a consulta com retorno único. Esses resultados sugerem que, à medida que o volume de dados aumenta, a diferença no aumento percentual do tempo médio de execução entre ServiceNow e webservice na rede local diminui.

Portanto, esses resultados sugerem uma relação entre o volume de dados e o tempo de execução, contradizendo a premissa comum de que “quanto menos dados, melhor o tempo de consulta”. No contexto do ServiceNow, um alto volume de dados é percebido como benéfico, pois estatisticamente é menos oneroso em termos de desempenho. Isso pode ser atribuído ao modelo de negócios do próprio Mid Server, que pode lidar eficientemente com consultas robustas. No entanto, quando usado para consultar um único dado via script, pode não ser a solução mais adequada.

Essa análise enfatiza a necessidade de considerar a complexidade das operações e a arquitetura do sistema ao avaliar o impacto do volume de dados no desempenho. Aspectos como a eficiência da API empregada em consultas, a qualidade do banco de dados disponível para consulta e outros elementos estruturais do sistema devem ser considerados. Embora o ServiceNow demonstre uma capacidade robusta de lidar com grandes volumes de dados, especialmente por meio do Mid Server, a eficácia da plataforma pode variar segundo a natureza específica da consulta realizada. Por exemplo, consultas que buscam recuperar um único dado por meio de script podem não se beneficiar da mesma otimização que consultas envolvendo grandes volumes de dados. Nesses cenários, é fundamental avaliar a adequação da abordagem adotada em relação aos requisitos particulares da aplicação.

Além disso, é importante ressaltar que a eficácia da plataforma ServiceNow não reside apenas na sua capacidade de processar grandes volumes de dados, mas também na sua flexibilidade e adaptabilidade às necessidades dos usuários e dos sistemas. Portanto, ao considerar o desempenho do ServiceNow em relação ao volume de dados, é essencial considerar não apenas os aspectos técnicos, mas também as nuances operacionais e contextuais que podem influenciar os resultados. Isso inclui considerações sobre a configuração do sistema, a otimização de consultas e a manutenção de boas práticas de desenvolvimento, a fim de garantir a eficiência e a escalabilidade da plataforma em diferentes cenários de uso.

8 Conclusões

Este capítulo encerra o estudo com as considerações finais e as etapas futuras para continuidade do trabalho.

8.1 Considerações Finais

O principal objetivo deste estudo foi desenvolver uma proposta de guia de melhores práticas e decisões para a otimização de integrações de bases de dados externas de serviços corporativos com o ambiente em nuvem do ServiceNow. Para atingir esse objetivo, foi realizada uma revisão da literatura a fim de fundamentar os temas relevantes relacionados ao contexto deste estudo.

A delimitação do escopo do estudo foi estabelecida com base nos casos de uso fornecidos pela Itaipu Binacional, visando representar situações reais de interesse em um ambiente controlado para o ensaio. Todas as licenças necessárias foram gentilmente cedidas por esta instituição, incluindo a instância do ServiceNow utilizada para a condução dos testes.

O estudo em questão apresentou uma análise do tempo de execução de consultas a uma API SOAP, mediada pelo Mid Server, por meio de um Fix Script no ServiceNow, comparada o tempo de execução isolado do webservice na rede interna. A partir dessa análise, observou-se um padrão que indica uma diminuição na diferença do aumento percentual do tempo médio de execução entre o ServiceNow e o Webservice à medida que o volume de dados aumenta.

Essa avaliação sugere uma relação entre o volume de dados e o tempo de execução, indicando que, no contexto do ServiceNow, um alto volume de dados é percebido como benéfico quando consultado via Fix Script, uma vez que estatisticamente é menos custoso em termos de desempenho. No entanto, para consultas de dados individuais, essa abordagem pode não ser a mais adequada.

O cenário ideal para a utilização da integração via Fix Script pode ser descrito como situações que exigem a importação de grandes volumes de dados, que podem ser processados em segundo plano. Apesar de constatado um melhor desempenho com grandes volumes de dados, a integração via Fix Script ainda pode apresentar um tempo de execução relativamente alto.

8.2 Trabalhos Futuros

O ServiceNow oferece uma ampla gama de formas de implementações de integrações de aplicações não abordadas nesse ensaio, que podem ser classificadas como sugestões de soluções apropriadas aos casos de uso de cada problema a ser resolvido.

Segue abaixo a sequência de trabalhos propostos como atividades a serem realizadas posteriormente para dar continuidade a este estudo:

1. **Análise de Desempenho Detalhada**
   Realizar uma análise mais aprofundada do desempenho do ServiceNow em diferentes cenários de consulta, considerando uma variedade de métricas, como tempo de resposta de cada parte do escopo, assim como utilização de recursos do sistema e escalabilidade.

2. **Otimização de Consultas**
   Identificar e implementar estratégias de otimização de consultas no ServiceNow, visando melhorar o desempenho em consultas específicas, reduzindo tempo de resposta e recursos necessários.

3. **Estudo de Caso Adicional**
   Realizar estudos de caso adicionais em ambientes de produção ou simulações mais próximas da realidade, a fim de validar os resultados obtidos e fornecer insights adicionais sobre o desempenho do ServiceNow em contextos específicos de uso.

4. **Avaliação de Alternativas**
   Investigar e comparar o desempenho do ServiceNow com outras plataformas ou soluções disponíveis no mercado para operações semelhantes, a fim de identificar possíveis alternativas ou complementos que melhorem a eficiência do sistema.

5. **Comparativo entre Integração via *Fix Script, Import Set* e *Integration Hub***
   Realizar um estudo comparativo entre esses três métodos de integração, considerandos critérios como facilidade de implementação, flexibilidade, desempenho, manutenção e escalabilidade, visando identificar as vantagens e desvantagens de cada abordagem de integração e fornecer recomendações para empresas que estão planejando implementar integrações no ServiceNow.

Referências

ABADI, D. J. Data management in the cloud: Limitations and opportunities. *IEEE Data Eng. Bull.*, v. 32, p. 3–12, 2009. Citado na página 8.

ABINADER, J. A.; LINS, R. D. *WEB SERVICES EM JAVA*. [S.l.]: Brasport, 2006. Citado 3 vezes nas páginas 4, 5 e 1.

ALVES, W. P. *Banco de dados*. [S.l.]: Saraiva Educação SA, 2014. Citado 2 vezes nas páginas 5 e 6.

ASSIS, C. B.; LAURINDO, F. J. B. Governança de ti e seu impacto na gestão da ti. In: *ENEGEP 2010 - XXX Encontro Nacional de Engenharia de Produção*. São Carlos: [s.n.], 2010. Citado na página 24.

BELQASMI, F. et al. Soap-based vs. restful web services: A case study for multimedia conferencing. *IEEE Internet Computing*, v. 16, n. 4, p. 54–63, 2012. Citado na página 21.

BIANCOLINO, C. A. et al. A gestão de ti e o valor de uso dos erp's em sua perspectiva de pós implementação. *Revista Eletrônica de Ciência Administrativa*, v. 10, n. 2, p. 5–19, 2011. Citado na página 23.

BRITO, C. R. et al. Plenary: The challenges of education in engineering, computing and technology without exclusions: Innovation in the era of the industrial revolution 4.0. In: *2020 IEEE World Conference on Engineering Education (EDUNINE)*. [S.l.: s.n.], 2020. p. 1–3. Citado na página 25.

BUYYA, R.; RANJAN, R.; CALHEIROS, R. N. Modeling and simulation of scalable cloud computing environments and the cloudsim toolkit: Challenges and opportunities. *CoRR*, abs/0907.4878, 2009. Disponível em: ⟨https://arxiv.org/abs/0907.4878⟩. Citado 2 vezes nas páginas 7 e 8.

BUYYA, R. et al. Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility. *Future Gener. Comput. Syst.*, v. 25, n. 6, p. 599–616, 2009. Citado na página 7.

CANCIAN, M. H. et al. *Uma proposta de guia de referência para provedores de software como um serviço*. Dissertação (B.S. thesis) — Universidade Federal de Santa Catarina, 2009. Citado na página 14.

CARR, C. N. et al. Oracle database: Uma análise aprofundada do sgbd que domina o mundo corporativo. *Research, Society and Development*, v. 12, n. 5, p. e0812541323–e0812541323, 2023. Citado na página 8.

CARVALHO, A. *Modelos de entrega de serviços em nuvem*. 2018. ⟨https://blog.binario.cloud/modelos-de-entrega-de-servicos-em-nuvem/⟩. Acesso em: 15 de mai de 2023. Citado na página 13.

COSTA, A. B. M. d. *Qualidade da arquitetura empresarial: estudo de caso envolvendo o LeanIX e o ServiceNow CMDB*. Tese (Doutorado) — Universidade do Minho (Portugal), 2022. Citado na página 1.

COSTA, A. B. M. da. *Qualidade da Arquitetura Empresarial: Estudo de Caso Envolvendo o Leanix e o Servicenow CMDB*. Tese (Doutorado) — Universidade do Minho, Portugal, 2022. Citado na página 27.

COSTA, L. et al. Grandes massas de dados na nuvem: Desafios e técnicas para inovação. *Minicursos do Simpósio Brasileiro de Redes de Computadores-SBRC*, p. 58, 2012. Citado na página 7.

DATE, C. J. *Introdução a sistemas de bancos de dados*. [S.l.]: Elsevier Brasil, 2004. Citado 3 vezes nas páginas 5, 6 e 7.

DINIZ, D. B.; SILVA, R. Ferreira da. Interoperabilidade de aplicaÇÕes com soap (simple object access protocol). Universidade Estadual de Goiás (UEG), 2021. Citado na página 20.

FERNANDES, A. A.; ABREU, V. F. *Implantando a Governança de TI: da estratégia à gestão dos processos e serviços*. 2. ed. Rio de Janeiro: Brasport, 2008. 444 p. Citado na página 24.

FERRIS, C.; FARRELL, J. What are web services? *Communications of the ACM*, v. 46, n. 6, p. 31, 2003. Citado na página 18.

FIELDING, R. T. *Architectural Styles and the Design of Network-based Software Architectures*. Tese (Doutorado) — University of California, Irvine, 2000. Citado 2 vezes nas páginas 21 e 22.

FIELDING, R. T.; TAYLOR, R. N. Principled design of the modern web architecture. *ACM Transactions on Internet Technology*, v. 2, n. 2, p. 115–150, 2002. Citado na página 22.

FOWLER, M.; LEWIS, J. *Microservices*. 2014. ⟨https://martinfowler.com/articles/microservices.html⟩. Acesso em: 21 de nov de 2023. Citado na página 9.

FOWLER, S. *Microsserviços prontos para a produção: Construindo sistemas padronizados em uma organização de engenharia de software*. Novatec Editora, 2017. ISBN 9788575226216. Disponível em: ⟨https://books.google.com.br/books?id=c0U4DwAAQBAJ⟩. Citado na página 11.

GOLDAR, H. *ServiceNow Developer Forum — Architecture of ServiceNow*. 2023. ⟨https://www.servicenow.com/community/developer-forum/architecture-of-servicenow/m-p/2518555⟩. Acesso em: 06 de dez de 2023. Citado na página 29.

HALILI, F.; RAMADANI, E. et al. Web services: a comparison of soap and rest services. *Modern Applied Science*, Canadian Center of Science and Education, v. 12, n. 3, p. 175, 2018. Citado na página 19.

IANKOSKI, J. *Aplicação do conceito de microsserviços e web APIs: construção de um sistema de coleta de opinião pública como estudo de caso*. Dissertação (B.S. thesis) — Universidade Tecnológica Federal do Paraná, 2022. Citado na página 9.

IBM. *Microsserviços*. 2021. ⟨https://www.ibm.com/brpt/cloud/learn/microservices⟩. Acesso em: 21 de nov de 2023. Citado na página 9.

JACOBS, D.; AULBACH, S. Ruminations on multi-tenant databases. In: *BTW*. [S.l.]: GI, 2007. (LNI, v. 103), p. 514–521. Citado na página 12.

KALINOWSKI, M.; REINEHR, S. Estruturando desenvolvimento de software como um serviço de ti: Uma experiência prática. In: SBC. *Anais do XII Simpósio Brasileiro de Qualidade de Software*. [S.l.], 2013. p. 318–325. Citado 2 vezes nas páginas 14 e 15.

KUMARI, S.; RATH, S. K. Performance comparison of soap and rest based web services for enterprise application integration. In: *2015 International Conference on Advances in Computing, Communications and Informatics, ICACCI 2015*. [S.l.: s.n.], 2015. p. 1656–1660. Citado na página 21.

MAGALHãES, I. L.; PINHEIRO, W. B. *Gerenciamento de serviços de TI na prática: uma abordagem com base na ITIL: inclui ISO/IEC 20.000 e IT Flex*. [S.l.]: Novatec Editora, 2007. Citado 2 vezes nas páginas 24 e 25.

MANOVICH, L. Banco de dados. *Revista ECO-Pós*, v. 18, n. 1, p. 7–26, 2015. Citado 2 vezes nas páginas 5 e 6.

MARQUES, A. I. A. *Desenvolvimento de API para aplicação cloud*. Tese (Doutorado) — Escola Superior de Tecnologia e Gestão, 2018. Citado 2 vezes nas páginas 21 e 22.

MATA, F. J.; FUERST, W. L.; BARNEY, J. B. Information technology and sustained competitive advantage: A resource-based analysis. *MIS Quarterly*, Management Information Systems Research Center, University of Minnesota, v. 19, n. 4, p. 487–505, December 1995. Citado na página 23.

MELNICHUK, M.; KORNIENKO, Y.; BOYTSOVA, O. Web-service. restful architecture. *Automatizácia tehnologičeskih i biznes-processov*, Odesa National Academy of Food Technologies, v. 10, n. 1, 2018. ISSN 2312-3125. Citado na página 19.

MENéNDEZ, A. I. M. *Uma ferramenta de apoio ao desenvolvimento de Web Services*. Dissertação (Mestrado) — niversidade Federal de Campina Grande, 2002. Citado 3 vezes nas páginas 4, 5 e 2.

MICROSOFT. *Excel help & learning*. 2024. ⟨https://support.microsoft.com/en-us/excel⟩. Acesso em: 15 de mai de 2023. Citado na página 40.

MIRANDA, C. M. d. S. *Orquestração multi-cloud em mercados regulados de serviços financeiros*. Tese (Doutorado) — Universidade do Minho, Portugal, 2022. Citado na página 33.

MOREIRA, L. O. Abordagem para qualidade de serviço em banco de dados multi-inquilinos em nuvem. 2014. Citado na página 8.

MOSTRöMM, M.; RYRBERG, S. *How to choose a web development framework: Analyzing best practices on the adoption of web frameworks*. Dissertação (Mestrado) — Linköping University, 2022. Citado na página 39.

MUNIZ, A. et al. *Jornada API na prática: unindo conceitos e experiências do Brasil para acelerar negócios com a tecnologia*. [S.l.]: Brasport, 2023. Citado na página 18.

MUNIZ, A. et al. *Jornada Microsserviços: do zero ao avançado somando conceitos e práticas*. [S.l.]: Brasport, 2022. Citado na página 9.

NETO, P. L. d. O. C. *Estatística*. [S.l.]: Editora Blucher, 2002. Citado na página 44.

NEWMAN, S. *Building Microservices: Designing Fine-Grained Systems*. O'Reilly Media, 2015. ISBN 9781491950333. Disponível em: ⟨https://books.google.com.br/books?id=jjl4BgAAQBAJ⟩. Citado na página 9.

OLIVEIRA, S. S. d. Bancos de dados não-relacionais: um novo paradigma para armazenamento de dados em sistemas de ensino colaborativo. *Revista da Escola de Administração Pública do Amapá*, v. 2, n. 1, p. 184–194, 2014. Citado na página 7.

PANDEY, D. *Sun Microsystems. JCP Specification, 2006*. 2006. ⟨http://jcp.org/en/jsr/detail?id=109⟩. Acesso em: 05 de dez de 2023. Citado na página 18.

PARAíBA, U. F. da. Estudando o percentual. *Oficina sobre porcentagem*, 2015. Citado na página 52.

PEREIRA, A. A. Segurança em web services para dispositivos móveis. 2010. Citado na página 18.

PINELI, J. C.; DUARTE, M. Ambiente de desenvolvimento de software em nuvem. *Revista Eletrônica e-Fatec*, v. 3, n. 1, p. 14–14, 2013. Citado na página 13.

PINTO, L. A.; HOFFMANN, S.; URIARTE, L. R. Análise comparativa entre as tecnologias de front-end react, angular e vue. *REVISTA DE EXTENSÃO E INICIAÇÃO CIENTÍFICA DA UNISOCIESC*, v. 10, n. 1, 2023. Citado na página 39.

RAMALHO, D. C. B. *Impactos da mudança de ferramenta de ITSM em uma central de serviços de TI em uma mineração: estudo de caso*. Tese (Doutorado) — Universidade Federal de Ouro Preto, 2022. Citado 4 vezes nas páginas 24, 25, 26 e 27.

RANI, D.; RANJAN, R. K. A comparative study of saas, paas and iaas in cloud computing. *International Journal of Advanced Research in Computer Science and Software Engineering*, v. 4, n. 6, 2014. Citado 2 vezes nas páginas 16 e 17.

RICHARDSON, C. *Introduction to Microservices*. 2015. ⟨https://www.nginx.com/blog/introduction-to-microservices/⟩. Acesso em: 20 de nov de 2023. Citado 3 vezes nas páginas 9, 10 e 11.

RODRIGUES, L. C.; MACCARI, E. A.; SIMõES, S. A. O desenho da gestão da tecnologia da informação nas 100 maiores empresas na visão dos executivos de ti. *JISTEM-Journal of Information Systems and Technology Management*, SciELO Brasil, v. 6, p. 483–506, 2009. Citado na página 24.

SERVICENOW. *ServiceNow Documentation: SOAP message*. 2022. ⟨https://docs.servicenow.com/bundle/tokyo-application-development/page/integrate/outbound-soap/concept/c_SOAPMessage.html⟩. Acesso em: 06 de dez de 2023. Citado na página 43.

SERVICENOW. *ServiceNow Product Documentation: Fix Script*. 2022. Acesso em 14 de maio de 2023. Disponível em: ⟨https://docs.servicenow.com/bundle/tokyo-application-development/page/build/applications/concept/c_FixScripts.html⟩. Citado na página 39.

SERVICENOW. *ServiceNow Product Documentation: Mid Server*. 2022. Acesso em 7 de setembro de 2023. Disponível em: ⟨https://docs.servicenow.com/pt-BR/bundle/tokyo-platform-security/page/product/mid-server/concept/mid-server-landing.html⟩. Citado na página 2.

SERVICENOW. *About ServiceNow — Who We Are and What We Do*. 2023. ⟨https://www.servicenow.com/company.html⟩. Acesso em: 06 de dez de 2023. Citado na página 27.

SERVICENOW. *Securing the Now Platform: An overview of the ServiceNow security program*. [S.l.], 2023. Citado 6 vezes nas páginas 28, 29, 30, 31, 32 e 33.

SERVICES, A. W. *Amazon API Gateway*. 2023. ⟨https://aws.amazon.com/pt/what-is/api/⟩. Citado na página 18.

SILVA, C. C. de M. et al. Design and evaluation of a services interface for the internet of things. *Wireless personal communications*, Springer US, New York, v. 91, n. 4, p. 1711–1748, 2016. ISSN 0929-6212. Citado na página 20.

SILVA, C. P. L. G. *Monitorização contínua dos controlos*. Tese (Doutorado) — Universidade do Minho, Portugal, 2022. Citado na página 28.

SILVA, W. *Diferença entre IaaS, PaaS e SaaS*. 2024. Acesso em: 15 de mai de 2023. Disponível em: ⟨https://www.linkedin.com/pulse/voc%C3%AAs-sabem-diferen%C3%A7a-entre-iaas-paas-e-saas-walaks-silva-seiff/⟩. Citado na página 13.

SINGH, M. Basic standards for web services. In: *Service-Oriented Computing*. [S.l.]: John Wiley & Sons, Ltd, 2004. p. 19–47. ISBN 0470091487. Citado na página 20.

SOAPUI. *SoapUI Docs*. 2024. ⟨https://www.soapui.org/docs/⟩. Acesso em: 15 de mai de 2023. Citado na página 40.

SOAPUI. *SoapUI Docs: API Docs*. 2024. ⟨https://www.soapui.org/docs/api-docs/⟩. Acesso em: 15 de mai de 2023. Citado na página 40.

SONI, A.; RANGA, V. Api features individualizing of web services: Rest and soap. *International Journal of Innovative Technology and Exploring Engineering*, v. 8, n. 9, p. 664–671, 2019. Citado na página 19.

SOUSA, F. R.; MOREIRA, L. O.; MACHADO, J. C. Computação em nuvem: Conceitos, tecnologias, aplicações e desafios. *II Escola Regional de Computação Ceará, Maranhão e Piauí (ERCEMAPI)*, p. 150–175, 2009. Citado 10 vezes nas páginas 1, 8, 11, 12, 13, 14, 15, 16, 17 e 35.

SOUSA, F. R.; MOREIRA, L. O.; MACHADO, J. C. Sladb: Acordo de nível de serviço para banco de dados em nuvem. In: *SBBD (Short Papers)*. [S.l.: s.n.], 2011. p. 155–162. Citado 2 vezes nas páginas 8 e 12.

SOUZA, D. et al. Análise do uso da tecnologia nos processos de recursos humanos: estudo de caso em uma universidade privada. *XIV SEGeT–Simpósio de Excelência em Gestão e Tecnologia*, 2017. Citado na página 39.

STEELE, L. W. *Managing Technology: The Strategic View*. New York: McGraw-Hill Inc, 1989. 356 p. Citado na página 24.

SUKHIJA, N. et al. Event management and monitoring framework for hpc environments using servicenow and prometheus. In: *Proceedings of the 12th international conference on management of digital ecosystems*. [S.l.: s.n.], 2020. p. 149–156. Citado na página 1.

SUNGKUR, R. K.; DAIBOO, S. Sorest, a novel framework combining soap and rest for implementing web services. In: *The Second International Conference on Data Mining, Internet Computing, and Big Data*. Reduit, Mauritius: [s.n.], 2015. p. 22–34. Citado na página 21.

TIHOMIROVS, J.; GRABIS, J. Comparison of soap and rest based web services using software evaluation metrics. *Riga Technical University*, v. 19, p. 92–97, 2016. Citado na página 18.

VILLAçA, L. H.; JR, A. F. P.; AZEVEDO, L. G. Construindo aplicações distribuídas com microsserviços. *Tópicos em Sistemas de Informação: Minicursos XV Simpósio Brasileiro de Sistemas de Informação — SBC*, 2018. Citado 3 vezes nas páginas 9, 10 e 11.