RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

ESTUDO DOS ASPECTOS PARA A DISPONIBILIZAÇÃO DE DADOS LOCAIS NA NUVEM: ESTUDO DE CASO NETDOCTOR

STUDY OF ASPECTS FOR THE AVAILABILITY OF LOCAL DATA IN THE CLOUD: NETDOCTOR CASE STUDY

Carlos Felipe Queiroz[1], Lucas Maués Calandrine Conceição[2], Marcos Vinicius Sadala Barreto[3]

Submetido em: 29/06/2021 e26474
Aprovado em: 20/07/2121 https://doi.org/10.47820/recima21.v2i6.474

RESUMO
Com a crescente utilização da internet por softwares de gestão de informação e comunicação, novas demandas se apresentam, tais como a integração de aplicações mobile com sistemas legados. Vê-se que as organizações, de modo geral, encontram problemas para manter o fluxo de informação entre a estrutura on-premises e serviços on-cloud que são necessários para distribuição de dados na internet de forma a preservar aspectos de seus sistemas legados. O objetivo deste estudo é projetar e implementar uma solução baseada em uma arquitetura multicamada utilizando chamadas de serviços web que persistem aspectos de segurança de informação de sistemas legados. O trabalho realizado é de natureza qualitativa bibliográfica exploratória, baseando-se na NBR ISO/IEC 27002:2013 que define os principais aspectos de segurança da informação e outras análises teóricas. Quanto aos procedimentos, apresenta-se a pesquisa experimental, com a elaboração de uma solução para a integração de um sistema de informação para gestão clínica (chamado NETDOCTOR), com um aplicativo para dispositivo móvel baseado na plataforma Android, através de Web Services ou API, utilizando técnicas de engenharia de software como UML para alcançar os objetivos. Os resultados evidenciam que é viável o desenvolvimento de softwares baseados em arquitetura orientada a serviços, para o intercâmbio de informações entre sistemas na área de prestação de serviços em saúde suplementar, de forma segura, ágil, escalável e com baixo custo de implementação, operação e manutenção.

PALAVRAS-CHAVE: On-premises. Cloud. Gestão de Informação e Comunicação.

ABSTRACT
With the growing use of the internet by information and communication management software, new demands arise, such as the integration of mobile applications with legacy systems. It is noticed that organizations, in general, face problems to maintain the flow of information between the local structure and the cloud services, necessary for the distribution of data on the internet, in order to preserve aspects of their legacy systems. The objective of this study is to design and implement a solution based on a multi-tier architecture using web service calls that persist information security aspects of legacy systems. The work performed have a qualitative bibliographic exploratory nature, based on the ISO / IEC 27002, 2013 standard, which defines the main aspects of information security and other theoretical analyses. As for the procedures, the experimental research is showed, with the development of a solution for the integration of an information system for clinical management (called NETDOCTOR), with an mobile device application based on the Android platform through Web Services or API, using software engineering techniques like UML to achieve the goals. The results show that software development based on a service-oriented architecture is viable, for the exchange of information between systems in the area of providing health services, in a safe, agile, scalable and low cost of implementation, operation and maintenance.

[1] Tecnólogo em Análise e Desenvolvimento de Sistemas pelo Instituo Federal do Pará - IFPA.
[2] Graduando em Análise e Desenvolvimento de Sistemas pelo Instituto Federal de Educação, Ciência e Tecnologia do Pará (IFPA)
[3] Mestrado em Engenharia Elétrica pela Universidade Federal do Pará (2007) e doutorado em Engenharia Elétrica pela Universidade Federal do Pará (2019) na área de computação aplicada.

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

KEYWORDS: *On-premises. Cloud. Information and communication management.*

INTRODUÇÃO

Empresas dos mais variados segmentos utilizam em seu cotidiano sistemas de informação para gerenciar e operacionalizar seus processos, dados, informações e serviços que dispõe. Muitas organizações utilizam em sua gestão sistemas desenvolvidos com ferramentas anteriores à popularização do desenvolvimento de sistemas para a Web. Estes sistemas, robustos e consolidados, que cresceram exponencialmente durante os anos e cujo custo de manutenção é mais viável economicamente do que a migração ou adoção de um sistema mais atual, utilizam, em sua grande maioria, data centers locais para hospedar seus Bancos de Dados, assim chamados *On-Premises*, que em tradução livre significa “nas instalações”, ou seja, a empresa instala a solução em seus servidores e infraestruturas (BÉRGAMO; OLIVEIRA, 2019).

O termo *on-premises* refere-se ao tipo de instalação de um software ou solução. Esta instalação é realizada dentro do servidor e infraestrutura de Tecnologia da Informação e Comunicação (TIC) da empresa. É o modelo tradicional de aplicativos de negócios (BHARGAVA et al., 2019).

De acordo com Jorgensen et al. (2014), com o modelo *on-premise*, a empresa é responsável pela segurança, disponibilidade e gerenciamento do software. Portanto, a empresa deve ter um departamento de sistemas que dedique parte de seus recursos à gestão da infraestrutura in loco. No entanto, o fornecedor também costuma fornecer suporte pós-venda e serviços de integração. As soluções de TIC que podem ser encontradas neste tipo de implementação, são aquelas que já existem há algum tempo no mercado, tais como gestão de documentos, sistemas "Enterprise Resource Planning" (ERP) ou sistema de gestão integrado, ou gestão de relacionamento com o cliente (CRM). Os sistemas mais recentes geralmente são oferecidos apenas em nuvem, e por isso possuem outro modelo de estrutura.

De fato, a tecnologia da informação está em permanente e rápida evolução, principalmente na área de software, que evolui tanto no caminho do aprimoramento da utilização dos dispositivos em que já estão inseridos, como também na utilização em novos dispositivos que são criados, como por exemplo os "smartphones", computadores de bolso que nos permitem ter acesso às mais diversas informações, a qualquer momento e em qualquer lugar.

Esta facilidade de acesso impulsiona uma demanda das pessoas por também terem em seus smartphones, as informações contidas nos sistemas que utilizam em seus trabalhos, que de modo geral, são sistemas de gestão integrados, mundialmente conhecidos como ERP (Enterprise Resource Planning) (PADILHA; MARTINS, 2005).

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

Um ERP (Enterprise Resource Planning) é um tipo sistema de informação com propósito gerencial e pode ser definido como “sistemas de informação adquiridos como pacotes comerciais de software que permitem a integração de dados dos sistemas de informação transacionais e dos processos de negócios ao longo de uma organização” (SOUZA e SACCOL, 2003).

As organizações que utilizam sistemas de gestão administrativa (ERP) que acessam bancos de dados locais encontram dificuldades para suprir essas demandas, pois para disponibilizar informações e serviços legados a seus colaboradores fora do ecossistema interno da empresa, é necessário que seja permitido o acesso externo à infraestrutura local e isso agrava o risco à segurança da informação, pois uma vez que a infraestrutura de seus sistemas é exposta na rede mundial de computadores, abre-se a possibilidade de acessos externos não autorizados e/ou maliciosos.

Objetivando não agravar o risco e preservar os aspectos locais de segurança da informação, vê-se a necessidade do desenvolvimento de uma arquitetura de software em nuvem que servirá de interface de comunicação entre o ERP e quaisquer sistemas e/ou aplicações externas, fazendo com que o acesso à estrutura local seja realizado de forma indireta e assim mitigando a vulnerabilidade frente à exposição à rede mundial de computadores.

A interface de comunicação que irá atuar entre os sistemas será desenvolvida a partir de um conjunto de definições e padrões utilizados no desenvolvimento e na integração de aplicações conhecida como API (*Application Programming Interface*, em tradução livre Interface de Programação de Aplicações). APIs são aplicações que atuam como interfaces entre sistemas distintos, estas permitem acesso a dados e interoperabilidade através de requisições utilizando-se de protocolos de comunicação em rede, como por exemplo o protocolo HTTP. Aplicações API podem também comumente ser denominadas de "*web services*", pois em sua grande maioria disponibilizam seus serviços através da rede mundial de computadores (*World Wide Web*).

A computação em nuvem é um conceito emergente que permite aos usuários acessar convenientemente recursos de computação remota. Como costuma acontecer nas fases iniciais de um novo conceito de tecnologia, as expectativas são altas, bem como suas projeções de crescimento. Em 2015, a computação em nuvem esteve entre as 10 principais tecnologias consideradas estratégicas para a maioria das organizações (PEREIRA et al., 2019).

Os analistas projetam que o mercado de serviços em nuvem crescerá de US $55 bilhões em 2017 para US $220 bilhões em 2025 e esperam que quase todos os recursos de computação migrem para a nuvem. Enquanto as ofertas de nuvem, como a Elastic Compute Cloud da Amazon e o Google Apps, estão ganhando rapidamente uma grande base de usuários, a migração de software empresarial para a nuvem ainda está em sua gênese (SANTOS, 2018).

A migração relativamente lenta para soluções em nuvem no domínio do software corporativo pode ser explicada por diferentes fatores: pelo lado da demanda, os usuários de negócios têm necessidades de tecnologia da informação (TI) mais complexas do que os usuários privados, uma vez

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

que usam software corporativo, como recursos corporativos sistemas de planejamento (ERP) ou gestão de relacionamento com o cliente (CRM), para apoiar suas operações de negócios centrais. Dada a criticidade de negócios desses sistemas, as empresas estão mais preocupadas com questões de segurança e desempenho, mas também enfrentam custos de troca significativos ao migrar para novas tecnologias (MAESTRI, 2018).

Consequentemente, a perspectiva do usuário de negócios foi extensivamente investigada em estudos recentes sobre os drivers e desafios da adoção da nuvem, bem como abordagens para tal migração. Contudo, está lenta migração para o software empresarial em nuvem também pode ser explicada do lado da oferta, onde os fornecedores, de modo geral, estão relutantes em apresentar ofertas de nuvem. Para eles, a computação em nuvem implica mudanças profundas: não apenas os força a entregar suas soluções pela Internet e a substituir seu modelo de licença de software por taxas de serviço; também exige que eles trabalhem novamente suas soluções para se tornarem totalmente habilitadas para a web e atender a vários clientes com a mesma instância, enquanto os aplicativos tradicionais são instalados e muitas vezes amplamente personalizados para um único cliente (FISHER et al., 2018).

Ao disponibilizar informações que estão em um banco de dados local para acesso fora do ambiente de Tecnologia da Informação da empresa (seus servidores), nos deparamos com algumas questões críticas como a segurança da informação.

Neste sentido, deve-se buscar ao máximo não elevar o risco à segurança da informação, mantendo os aspectos de segurança já existentes da estrutura de tecnologia da informação local. A partir do momento que os dados estarão disponíveis para acesso externo, estes necessitam estar disponíveis pelo maior tempo possível, uma vez que novos sistemas estarão acessando estes dados. Com o aumento do número de sistemas, possivelmente ocorrerá aumento no número de requisições aos dados e, portanto, a solução deve ser capaz de atender a este aumento de carga computacional.

De forma a atender a estas premissas, uma das abordagens possíveis é desenvolver um serviço em nuvem para receber estes dados do sistema local e então disponibilizá-los na nuvem para que estes sejam acessados por aplicações externas.

Para que isso seja possível, podemos basear o desenvolvimento desta solução nas definições de segurança da informação presentes na NBR ISO 27002:2013, que define a proteção da informação de vários tipos de ameaças para garantir a continuidade do negócio, maximizar o retorno sobre os investimentos e as oportunidades de negócio, minimizar o risco ao negócio e estabelece como objetivo da segurança da informação a preservação da confidencialidade, da integridade e da disponibilidade da informação (NBR ISO 27002, 2013).

Ainda segundo a NBR ISO 27002:2013, a segurança da informação é obtida a partir da implementação de um conjunto de controles adequados, incluindo políticas, processos, procedimentos, estruturas organizacionais e funções de *software* e *hardware*.

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

Esses termos estão definidos na NBR ISO 27002:2013 como: I) Confidencialidade: propriedade de que a informação não esteja disponível ou revelada a indivíduos, entidades ou processos não autorizados; II) Integridade: propriedade de salvaguarda da exatidão e completeza de ativos; III) Disponibilidade: propriedade de estar acessível e utilizável sob demanda por uma entidade autorizada.

Portanto, uma vez que estes aspectos estão presentes em ERPs *on-premises*, a preservação destes aspectos em nuvem também é necessária.

Assim, estudos sobre como o fluxo de informação entre a estrutura *on-premises* e a estrutura on-cloud é construído preservando os aspectos de segurança da informação de sistemas legados são importantes para a segurança da informação e aspectos ACID[1].

Em termos de competências e recursos, a distribuição de sistemas em nuvem implica que os fornecedores de *software* operem centros de dados e gerenciem aplicativos, além de suas atividades tradicionais de desenvolvimento de *software*. Ao disponibilizar informações que estão em um banco de dados local para acesso fora do ambiente de Tecnologia da Informação da empresa (seus servidores), tem-se algumas questões críticas como a segurança.

Em meados da década de 1990, a ISO (*International Organization for Standardization*) e a IEC (*International Electrotechnical Commission*) criaram um arcabouço de normas consolidando as diretrizes relacionadas à Segurança da Informação, através da série 27000.

Nesta série, temos a NBR ISO/IEC 27002, norma internacional que estabelece as melhores práticas para a implantação do SGSI (Sistema de Gestão de Segurança da Informação) nas organizações, sejam elas públicas ou privadas, independente do porte.

Sua última atualização foi realizada em 2013, trazendo um aumento no número de seções e eliminação no número de controles, assim como alterações de terminologias, buscando refletir a evolução de práticas de gestão de segurança da informação nos últimos anos.

A adoção das políticas e normas relacionadas à segurança da informação definidas na NBR ISO 27002 no processo de desenvolvimento de software, proporciona inúmeras vantagens para os sistemas e empresas que os adotam, tais como: I) Identificação e correção de possíveis ameaças à integridade II) Oferece a possibilidade de redução de custos relacionados a possíveis incidentes de segurança; III) Possibilita a maior conformidade com a legislação e outras regulamentações quanto a segurança.

O tema Segurança da Informação vem se popularizando nos últimos anos principalmente devido ao grande aumento no número de incidentes de segurança, que vem ocorrendo em diversas esferas em todo o planeta. Uma vez que o acúmulo e a exposição das informações aumentam a cada dia, alavancado pela Internet, é natural que o número de incidentes aumente.

[1] ACID é uma sigla que se refere a quatro aspetos de transação de um sistema de banco de dados: Atomicidade, Consistência, Isolamento e Durabilidade

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

Os problemas causados por estas ocorrências podem ser os mais variados, como danos à imagem da organização, descumprimento de obrigações legais, vazamento de informações confidenciais e até mesmo interrupção ou mal funcionamento de operações, acarretando em prejuízos financeiros diretos ou indiretos.

Tendo em vista que a NBR ISO 27002:2013 foi projetada "como uma referência na seleção de controles dentro do processo de implementação de um sistema de gestão da segurança da informação … ou como um documento de orientação para as organizações implementarem controles de segurança da informação comumente aceitos (NBR ISO 27002:2013)." É de vital importância a adoção e aplicação desta norma ou de outra equivalente, no intuito de mitigar os riscos à segurança da informação e evitar os danos e prejuízos que possam vir a ocorrer.

Nossa pesquisa aborda o ambiente de software empresarial e de que forma ocorre o fluxo de informação entre a estrutura *on-premises* e a estrutura on-cloud no âmbito da segurança da informação. Com a necessidade de se atualizar perante as tecnologias atuais, mantendo os serviços ofertados, empresas que utilizam sistemas ERP com servidores de bancos de dados locais necessitam disponibilizar dados e funcionalidades em nuvem para que possam expandir o alcance de seus serviços mantendo os principais aspectos relacionados à segurança da informação.

O objetivo principal é projetar e implementar uma solução baseada em uma arquitetura multicamada utilizando requisições de serviços web que persistem aspectos de segurança de informação de sistemas legados que possuem a NBR ISO 27002:2013, tendo como os objetivos específicos:

- Estudo dos aspectos de segurança de informação;
- Estudo dos aspectos de estruturas *on-premises* e on-cloud
- Teorização da segurança de informação em sistemas *on-premises* legados, bem como a segurança de informação em sistemas on-cloud;
- Desenvolver um estudo de caso persistindo os aspectos estudados de segurança informação;
- Demonstrar a viabilidade de disponibilizar dados de sistemas legados em nuvem, respeitando os aspectos de segurança de informação.

O trabalho realizado é de natureza qualitativa bibliográfica exploratória, baseando-se na NBR ISO 27002:2013 que define os principais aspectos de segurança da informação e outras análises teóricas com o objetivo de aplicar as teorias analisadas no estudo de caso NETDOCTOR utilizando técnicas de engenharia de software como UML para alcançar os objetivos definidos.

O presente trabalho está dividido em 3 tópicos, sendo o primeiro quanto a fundamentação teórica dos aspectos de segurança da informação definidos pela NBR ISO 27003; a teorização destes aspectos aplicados a sistemas *on-premise* e on-cloud assim como também a análise dos conceitos referentes às tecnologias empregadas a fim de alcançar a solução proposta, assim como são relatados trabalhos e soluções relacionados ao mesmo problema deste trabalho. O segundo tópico apresenta a estrutura do projeto, relacionando os requisitos técnicos e as funcionalidades do sistema

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

NETDOCTOR que são alvo da modelagem da solução, que estarão presentes na API e apresenta o desenvolvimento do projeto e sua implementação como um estudo de caso para demonstrar o uso de Serviços Web baseados na arquitetura REST para a resolução do tema proposto. No terceiro são apresentados os resultados obtidos com a análise do referencial teórico e a implementação da solução proposta para o estudo de caso, assim como é realizada a análise da solução desenvolvida perante os aspetos definidos para a segurança da informação.

1 FUNDAMENTAÇÃO TEÓRICA

Este capítulo apresenta os aspectos teóricos da segurança da informação e a base das tecnologias de Serviços Web, mostrando conceitos importantes das tecnologias que serão empregadas, assim como são relatadas algumas investigações e soluções relacionados ao mesmo problema deste trabalho.

1.1 Aspectos de segurança de informação NBR ISO/IEC 27002:2013

De acordo com a NBR ISO 27002: A segurança da informação é alcançada pela implementação de um conjunto adequado de controles, incluindo políticas, processos, procedimentos, estrutura organizacional e funções de *software* e *hardware*. Os 3 principais aspectos abordados na segurança da informação a serem observados, segundo a norma são: I) Confidencialidade; II) Integridade e: III) Disponibilidade (NBR ISO 27002, 2013).

O conceito de confidencialidade se refere à proteção de informações que não devem ser acessadas por indivíduos não autorizados. Isso significa dizer que determinadas informações são confidenciais e só dispõe de seu acesso aqueles que possuem autorização para tal. Portanto, este aspecto tem como principal objetivo assegurar a proteção das informações de sistemas de cunho confidencial e sigiloso (NBR ISO 27002, 2013).

Para a integridade, seu conceito permeia a segurança da informação e está relacionado à plenitude do armazenamento dos dados. Isto é, da mesma forma que as informações são fornecidas, elas devem ser armazenadas, sem qualquer alteração em seu conteúdo. Portanto, o princípio de integridade garante que todas as informações estejam em seu formato original e verdadeiro, a fim de servir para os propósitos para o qual foram designadas. Em outras palavras, elas devem permanecer íntegras (NBR ISO 27002, 2013).

No que tange à disponibilidade, seu conceito diz respeito ao acesso dos dados, sempre que este for necessário. Significa a garantia de que as informações estão sempre disponíveis. Esse princípio está diretamente relacionado à eficácia do sistema e do funcionamento da rede para que, consequentemente, a informação possa ser acessada quando for necessário (NBR ISO 27002, 2013).

1.2 Políticas de segurança de informação frente aos ativos de uma organização

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

As facilidades para se conectar às redes aumentaram; além disso, os aplicativos e *softwares* estão cada vez mais amigáveis e acessíveis, desta forma todos tendem a se conectar em rede para compartilhar recursos, mas essa facilidade de conexão também representa um aumento nos riscos de que as informações e recursos de uma organização possam ser comprometidas (OLIVEIRA, 2017).

Por este motivo, medidas de segurança devem ser implementadas para proteger as informações da empresa. Segurança significa ter os meios para reduzir, tanto quanto possível, a vulnerabilidade das informações e recursos. Em função das várias maneiras nas quais as ameaças podem se aproveitar das vulnerabilidades para causar dano à organização, os riscos de segurança da informação estão sempre presentes. Uma segurança da informação eficaz reduz estes riscos, protegendo a organização das ameaças e vulnerabilidades e, assim, reduzindo o impacto aos seus ativos[2] (NBR ISO 27002, 2013).

Hackers e crackers estão constantemente monitorando redes para encontrar vulnerabilidades e fragilidades em um sistema de informação, o desenvolvimento de software tornou a configuração e o uso cada vez mais fáceis, a Internet também permite a conectividade de qualquer pessoa, desta forma as ameaças à segurança das informações tendem a ter uma maior probabilidade de ocorrência. A qualquer momento um servidor ou dispositivo de rede pode ser atacado com propósitos negativos à imagem, funcionalidade ou outros aspectos da organização ou instituição (CHEROBINI, 2017).

A informação, nos dias atuais, constitui um dos principais recursos (ativos2) de uma organização, portanto deve ser protegida, através de um conjunto de atividades, controles e políticas de segurança que devem ser implementadas com base em recursos humanos, *hardware* e *software* (SILVA, 2019).

A segurança da informação depende de uma gestão e procedimentos adequados, dos colaboradores da organização, fornecedores, clientes, acionistas e do nível de segurança dos meios técnicos.

Os principais ativos associados aos sistemas de informação de uma organização podem ser classificados de acordo com os (SILVA, 2019):

I. recurso de informação, que são Bancos de dados, manuais de usuário, procedimentos operacionais ou de suporte, planos de continuidade, informações arquivadas ou provisões de emergência para recuperação de informações;
II. softwares, que são considerados aplicativos, sistemas operacionais, ferramentas de desenvolvimento e utilitários;

[2] Ativos - Conjunto dos bens de direito e de propriedade da empresa, mensuráveis monetariamente, que tenham a capacidade de gerar benefícios presentes ou futuros.

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

III. equipamentos, sendo estes servidores, computadores, roteadores, switches, hubs, equipamentos de energia, ar condicionado, equipamentos de comunicação, dentre outros;

IV. serviços, que podem ser considerados como comunicações, processamento de informática, dentre outros serviços;

A segurança deve garantir as seguintes características das informações (SILVA, 2019):

I. Confidencialidade, ou seja, que as informações são conhecidas apenas por pessoas autorizadas;

II. Integridade, o que significa que seu conteúdo não é alterado, a menos que seja modificado por pessoal autorizado;

III. Disponibilidade, ou seja, a possibilidade de estar sempre disponível para ser processado por pessoas autorizadas;

IV. Controle, pois apenas pessoas autorizadas podem decidir quando e como acessar as informações;

V. Autenticidade, uma vez que a informação só se torna válida e utilizável quando a fonte de informação é válida;

VI. Proteção de repetição: a transação ocorre apenas uma vez, a menos que esperado de outra forma;

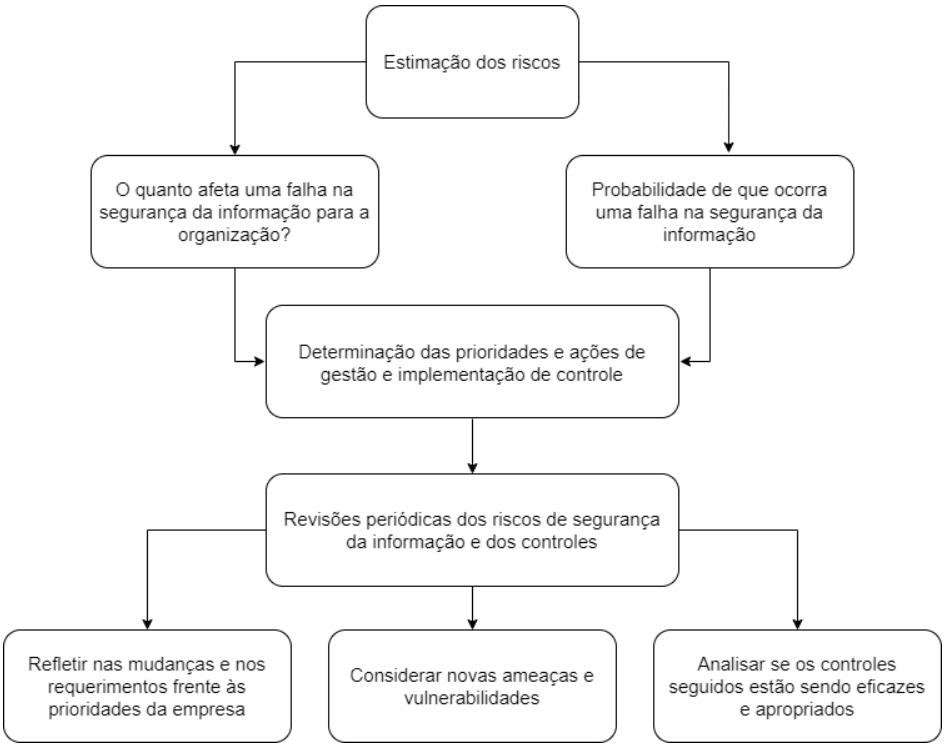
VII. Irrecusabilidade: Para evitar que uma entidade que recebeu ou enviou informações alegue que não o fez.

Observa-se que as redes de dados devem ser protegidas, uma vez que existem várias ameaças. Deve ser constantemente realizada uma avaliação dos ativos e determinada a sua importância, bem como o risco a que estão sujeitos.

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

1.3 Ameaças à segurança da informação

Figura 1 - Estimação dos Riscos



A seguir, são destacadas as ameaças mais frequentes, conforme relatam Cipriano (2021) e Santos e Marconi (2020):

Desastres naturais: esses tipos de ameaças geralmente causam a interrupção dos serviços, afetando principalmente a disponibilidade de informações, exemplos desses tipos de ameaças são as causadas pela natureza: inundações, terremotos, tornados, dentre outros (CIPRIANO, 2021).

Ameaças físicas: relacionadas ao acesso físico aos recursos, podem resultar em roubo, danos físicos ao equipamento, sabotagem. O acesso não autorizado, mas que é conseguido através de engenharia social, explorando a confiança dos colaboradores de uma organização.

Fraude informática: representada por enganar clientes na venda de produtos e serviços por meio de promoções e agências que não existem.

Intrusões: acesso não autorizado aos sistemas de comunicação, aos servidores de uma organização, com o objetivo de prejudicar a imagem ou obter benefícios econômicos indevidos (CIPRIANO,2021).

Erros humanos: como o próprio nome indica, eles resultam da ação humana, tais como: senhas facilmente vulneráveis, backup de sistemas malfeitos, interrupção de serviços, configurações de dispositivos incompletos (SILVA, 2019).

Software ilegal: as consequências da cópia de software ilegal levam a vulnerabilidades nos sistemas do computador, uma vez que não há atualizações fornecidas pelos desenvolvedores, o

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

software ilegal também possui outras ameaças, como códigos maliciosos (SANTOS: MARCONI, 2020).

Código malicioso: é qualquer programa ou parte do programa (software) que causa problemas nos sistemas informáticos, como vírus, *trojans*, *worms*, *backdoors*, quando são ativados nos sistemas finais. Esse tipo de ameaça evoluiu devido ao aumento de conectividade da Internet e aos recursos enganosos usados pelos invasores (SANTOS; MARCONI, 2020).

Como observado acima, é necessário estimar os riscos aos quais a rede, os servidores e os dispositivos de rede estão sujeitos. Embora seja difícil fazer uma avaliação exata das informações, pode-se tentar avaliá-las assumindo sua perda ou alteração. Na figura 1 é exposta uma possível metodologia para implementar um sistema de segurança.

1.4 Política de segurança na implementação de um sistema de segurança

A implementação de um sistema de segurança deve ser complementada com políticas de segurança. A política de segurança requer não apenas conhecer as ameaças às quais as informações e recursos de uma organização estão expostos, mas também estabelecer sua origem, que pode ser interna ou externa à organização (SILVA, 2019).

Seria inútil proteger a empresa de usuários externos se também houvesse ameaças internas. Por exemplo, se um usuário usa um pendrive que contém um vírus, ele pode se propagar por toda a rede interna da empresa.

Uma política de segurança é a declaração de regras que devem ser respeitadas para acessar informações e recursos. Os documentos de uma política de segurança devem ser dinâmicos, ou seja, devem ser continuamente ajustados e aprimorados de acordo com as mudanças que ocorrem nos ambientes onde foram criados (TSAREGORODTSEV, 2018).

As políticas de segurança devem ser conhecidas por todos os funcionários de uma organização. No conteúdo dos documentos deve ser claramente estabelecido: o objetivo, os responsáveis pelo cumprimento e as medidas a serem aplicadas em caso de não cumprimento (RMACHANDRA et al., 2017). Os documentos incluem o seguinte:

I. administração de usuários que regulará o acesso aos recursos pela equipe da organização;
II. cópias de segurança: neste, descreve-se os passos a seguir para garantir uma recuperação adequada das informações, por meio de cópias de segurança;
III. tratamento da informação: define-se claramente os tipos de informação que são tratados por pessoas autorizadas dentro da organização;
IV. software legal: define-se o uso de software na empresa com licenças de uso legais;
V. uso do serviço de internet e e-mail: descreve-se a proteção das informações por meio do uso de e-mail e do serviço de Internet;
VI. ambientes de processamento: define-se de que forma deve ser o uso de ambientes de processamento de informações;

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

VII. segurança nas comunicações: descreve-se a proteção da informação durante os processos de transmissão e recepção de dados em redes internas e externas;

VIII. auditorias de sistemas: permite-se um controle dos eventos de segurança dos sistemas;

IX. continuidade do processamento: as atividades relacionadas com a recuperação da informação em casos críticos serão definidas e regulamentadas através de metodologia adequada;

X. proteção física: define-se a proteção física do equipamento, processamento, armazenamento e transmissão de informações;

XI. sanções por descumprimento: este documento contemplará as medidas que serão aplicadas em caso de descumprimento das regras definidas (RAMACHANDRA et. al., 2017).

Compreende-se, portanto, que a informação é um recurso muito importante para a organização e deve ser protegida através da implementação de medidas de segurança baseadas em hardware, software e recursos humanos, mas também complementada com políticas de segurança adequadas e conhecidas do pessoal da organização a todos os níveis. O pessoal da organização deve se identificar totalmente com os objetivos de proteção e segurança buscados pela empresa. Neste sentido, vê-se que a segurança da informação é tarefa de todos: funcionários da empresa, parceiros, acionistas, clientes.

1.5 Segurança de informação em sistemas *on-premises* legados

Nas redes físicas existe um controle total da informação por parte do prestador que a implementa, exigindo a adoção de políticas de segurança e uso de técnicas e ferramentas tradicionais na rede física. Na nuvem quem realiza o controle total da informação é o provedor ou há uma divisão das responsabilidades, onde o provedor e o contratante se responsabilizam pela segurança da informação. Assim, o provedor também utiliza de políticas de segurança dentro de seu datacenter, a diferença é que essas políticas são aplicadas pelo provedor e não pelo cliente. No entanto, o contratante deve manter as boas práticas de segurança dentro de sua organização.

Considerando o setor de Tecnologia da Informação, nas redes físicas, ele pode não receber a devida atenção pelo fato de não ser o foco da organização e isso implica vulnerabilidade. Já os provedores de nuvem trabalham exclusivamente com isso e possuem todo aparato necessário para administrar a segurança dos dados do cliente (PALANIMALAI, 2018).

Trabalhar com sistemas legados às vezes é uma necessidade. Isso pode ser imposto pelos custos de desenvolvimento de uma nova plataforma. Mas também porque o tempo necessário para substituí-lo implica que o existente deve ser mantido por muito tempo.

Quer a extensão da vida útil do software ou hardware seja motivada pelos custos, como se fosse uma necessidade temporária, é importante notar que os sistemas legados apresentam alguns riscos. Isso deve ser levado em consideração, tanto ao decidir se realmente é um sistema que ainda é válido, quanto ao calcular os custos totais de não o renovar. Nesse caso, a manutenção que pode ser muito cara será incluída (PALANIMALAI, 2018).

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

Um dos principais riscos é que um sistema desatualizado tende a falhar, e resolver seus problemas pode ser muito caro. Tanto no horário de trabalho, que às vezes pode ser muito especializado e difícil de encontrar, quanto nos momentos em que o sistema legado não estará em serviço.

Associado a este problema está o desempenho de sistemas desatualizados. É possível que tenham um desempenho muito bom quando foram desenvolvidos. Mas hoje em dia, os tempos de espera podem ser muito mais lentos do que em aplicativos recentes. Além de trazer transtornos aos usuários, pode ser uma barreira para, por exemplo, apresentar informações em tempo real.

Mas mesmo que a plataforma e o software sejam estáveis e continuem a funcionar perfeitamente em condições normais, o risco de segurança sempre estará lá. Muito provavelmente, do sistema operacional ao aplicativo e outros componentes de software terão parado de receber atualizações. Isso significa que qualquer brecha de segurança em potencial detectada desde que o suporte cessou é difícil ou impossível de mitigar (NAKKEERAN et al. 2020).

Nestes casos, é comum que se tenham ações em prol de estabelecer camadas de segurança em torno do problema. Mas isso não impede que um invasor explore a vulnerabilidade. Isso produz uma falsa sensação de segurança, que pode ser tão perigosa quanto a brecha a ser fechada.

Além disso, em uma época em que os aplicativos estão interconectados, tanto dentro da empresa operacional quanto com ferramentas de terceiros, as limitações técnicas podem levar a problemas intransponíveis. Entre eles, um sistema legado vital para a empresa que não se conecta a aplicativos de faturamento ou sistemas de pagamento ou continue pagando pelo armazenamento local de arquivos quando seria mais interessante levar essas informações para a nuvem. Este último se deve, em muitos casos, ao fato de que, ao usar software desatualizado, não é possível se conectar com AWS, Azure ou Google Cloud (NAKKEERAN et al., 2020).

Nesses casos, a incompatibilidade de software de sistemas legados pode ser substancial e, por conseguinte, ser oneroso à empresa. Em outros, pode-se limitar diretamente o desenvolvimento do negócio, impedindo o uso de certas ferramentas de terceiros. Esse tipo de incompatibilidade pode vir de qualquer lugar. Assim, o sistema operacional pode ter uma pilha TCP/IP que não permite o aproveitamento de certas funções de qualidade de serviço (QoS) ou a linguagem de programação escolhida pode não ter bibliotecas com as versões mais atuais do TLS, impedindo a conexão com certas APIs externas. Além disso, o servidor web pode não permitir o uso de certos tipos de criptografia (NAKKEERAN et al., 2020).

Em alguns destes casos, o software pode ser desenvolvido para servir como intermediário entre o sistema legado e o resto do mundo. Nestes casos, um sistema inseguro pode ser isolado do resto do mundo e parte de seus problemas podem ser resolvidos. Para isso, é possível instalar um servidor web moderno que atua como proxy do antigo e até desenvolver um adaptador que se conecta à API de um serviço em nuvem e interage com o aplicativo pegado (WAN et al., 2017).

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

No entanto, outros problemas têm uma solução mais complicada. O desempenho, por exemplo, pode ser resolvido em alguns casos virtualizando o software antigo em uma plataforma moderna. Mas nem todos os sistemas legados são virtualizados, seja porque são hardwares diferentes, ou porque as licenças de software disponíveis não permitem isso (NAKKEERAN et al., 2020).

A solução costuma ser enfrentar o temido momento de lançar um novo projeto para substituir o sistema legado. Sem dúvidas, os custos podem ter uma limitação e o projeto também pode se prolongar no tempo. Mas ter mais tempo para iniciá-lo tem duas vantagens: a primeira é que mais tempo pode ser gasto na análise e garantir a qualidade do novo produto. Aproveita-se, também, para fazer todas as melhorias necessárias, treinar as pessoas que utilizam a ferramenta, dentre outros (WAN et al., 2017). Além disso, um projeto de longo prazo executado por uma equipe pequena, e mesmo por equipe técnica já contratada pela empresa pode ser mais sustentável financeiramente do que outro em que recursos humanos necessários precisam ser superdimensionados para chegar à sua conclusão dentro do prazo esperado.

Nesse sentido, vale a pena considerar outras desvantagens dos sistemas legados. Em geral, os profissionais que trabalham com eles são altamente qualificados, que aprenderam a fazê-lo quando estavam com a tecnologia mais recente. Atualmente se torna difícil contratar profissionais com experiência com sistemas antigos. Enfrentar uma migração, com todos os inconvenientes que ela acarreta, é uma boa oportunidade para a equipe técnica atualizar os seus conhecimentos.

1.6 Segurança de informação em sistemas on-cloud

A segurança na nuvem se refere à prática de proteger a integridade de aplicativos, dados e infraestrutura baseada em nuvem virtual. O termo se aplica a todos os modelos de implantação de nuvem (nuvem pública, nuvem privada, nuvem híbrida, nuvem múltipla) e a todos os tipos de serviços e soluções sob demanda baseados em nuvem (IaaS, PaaS, SaaS). De modo geral, com serviços baseados em nuvem, o provedor de nuvem é responsável por proteger a infraestrutura subjacente, enquanto o cliente é responsável por proteger aplicativos e dados na nuvem (FISHER et al., 2018).

De acordo com Pereira et al., (2016 p15.),

> "Os serviços implantados nesses sistemas são armazenados e processados em um ou mais servidores de data centers, acessados remotamente. Como esses serviços passam a ser executados nas máquinas dos data centers, a tarefa de manutenção do serviço é deslocada dos clientes, que estão pagando por esse serviço, para os gestores dos data centers, agora denominados provedores de serviços..." (PEREIRA et al., 2016 p. 15)

A seguir, evidenciam-se os modelos de implantação de nuvem e os tipos de serviços conforme Fischer et al., (2018):

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

Quadro 1 - Modelos de implantação de nuvem e tipos de serviços

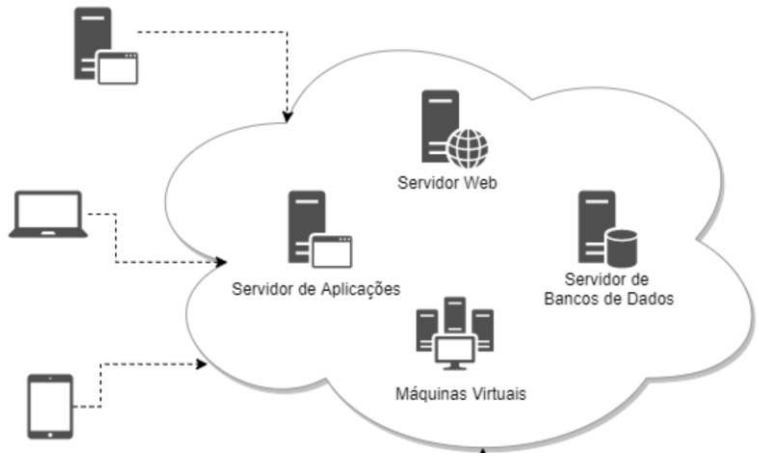
Modelos de implantação de nuvem	
Nuvem pública	Uma oferta pública de multilocação, como Amazon Web Services (AWS), Microsoft Azure ou Google Cloud Platform (GCP)
Nuvem privada	Um ambiente de nuvem dedicado a uma única entidade de negócios (mas normalmente compartilhado por muitas organizações dentro dessa entidade)
Nuvem híbrida	Uma mistura de serviços de nuvem pública e privada no local
Multicloud	Uma combinação de serviços em nuvem; geralmente inclui vários tipos de serviços (computação, armazenamento, etc.) hospedados em várias nuvens públicas e privadas
Tipos de serviços em nuvem	
Infraestrutura como serviço (IaaS)	Serviços de computação, armazenamento e rede subjacentes sob demanda
Plataforma como serviço (PaaS)	Ambientes e estruturas de desenvolvimento de aplicativos baseados em nuvem
Software como serviço (SaaS)	Soluções sob demanda, como Salesforce ou Office 365, oferecidas como aplicativos baseados em nuvem com modelos de licenciamento baseados em assinatura

Fonte: Adaptado de Fischer et al., (2018)

Compreende-se que as cargas de trabalho em nuvem são vulneráveis a uma variedade de ameaças. Os recursos e cargas de trabalho em nuvem estão expostos a uma ampla variedade de ameaças à segurança cibernética, incluindo violações de dados, *ransomware*, ataques *DDoS* e ataques de *phishing* (TAURION, 2009).

Os atacantes cibernéticos podem explorar vulnerabilidades de segurança na nuvem, usando credenciais roubadas ou aplicativos comprometidos para realizar ataques, interromper serviços ou roubar dados confidenciais. Fortes sistemas e práticas de segurança em nuvem são essenciais para manter disponíveis os aplicativos essenciais aos negócios, protegendo informações confidenciais e garantindo a conformidade regulatória. A seguir, tem-se a Figura 2, que apresenta uma ilustração do processo de computação em nuvem.

Figura 2 - Processo geral da computação em nuvem



RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

1.7 Arquitetura cliente-servidor

A seguir, são apresentadas as arquiteturas cliente-servidor mais populares, sendo estas a arquitetura cliente-servidor de duas camadas, e a arquitetura cliente-servidor de três camadas.

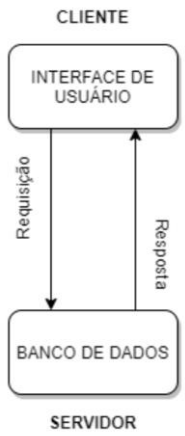
1.7.1 Arquitetura cliente-servidor de duas camadas

Na arquitetura cliente-servidor de duas camadas, compreende-se que consiste em uma camada de apresentação e lógica do aplicativo e o outro da base de dados. Normalmente esse tipo de arquitetura é empregado nas seguintes situações (TEIXEIRA, 2019):

I. quando é pouco necessário o processamento de dados na organização;
II. quando se tem um banco de dados centralizado em um único servidor;
III. quando o banco de dados é relativamente estático ou;
IV. quando se tem manutenção mínima.

Na arquitetura de duas camadas, a interface do usuário é armazenada na máquina cliente e o banco de dados é armazenado em um servidor. A lógica de negócios e a lógica do banco de dados são arquivadas no servidor ou cliente, mas devem ser mantidas em boas condições. Na arquitetura de duas camadas, o servidor e o cliente devem ser incorporados diretamente. Neste caso, por exemplo, se um cliente fornece alguma entrada para um servidor, não deve haver nenhum meio-termo. Geralmente é feito para obter resultados rápidos e evitar confusão entre vários clientes (COELHO et al., 2000). A seguir, ilustra-se a arquitetura cliente-servidor de duas camadas (ver Fig. 3).

Figura 3 - arquitetura cliente-servidor de duas camadas



RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

1.7.2 Arquitetura cliente servidor de três camadas

Na arquitetura cliente-servidor de três camadas, esta consiste em uma camada de apresentação, outra camada de lógica do aplicativo e outra camada do banco de dados. Normalmente essa arquitetura é empregada nas seguintes situações (SCOBA et al., 2019):

I. quando há muito processamento de dados no aplicativo;
II. em aplicações onde a funcionalidade está em constante mudança;
III. quando os processos não estão intimamente relacionados aos dados;
IV. quando é necessário isolar a tecnologia do banco de dados para que seja fácil a mudança;
V. quando é necessário separar o código do cliente para a manutenção ou;
VI. quando é muito adequado para uso com tecnologia orientada ao objeto.

Consiste, portanto, na camada de apresentação que é a camada de interface do usuário, uma camada de aplicativo que é uma camada de serviço, que executa o processamento detalhado, e uma camada de dados que consiste no servidor de banco de dados, que armazena informações. Funciona das seguintes maneiras: o sistema do Cliente lida com a camada de apresentação; o servidor de aplicativos cuida da camada de aplicativos e o sistema do servidor monitora a camada de banco de dados. A seguir, ilustra-se a arquitetura cliente-servidor de três camadas (ver Fig. 4).

Figura 4 - Arquitetura cliente-servidor de três camadas

1.7.3 Sistemas cliente-servidor

Quanto aos sistemas cliente-servidor, tem-se uma classificação de acordo com o nível de abstração do serviço que é oferecido. Na representação distribuída, a interação com o usuário ocorre no servidor, o cliente atua como um gateway entre o usuário e o servidor. Na representação remota, a lógica do aplicativo e do banco de dados estão no servidor. O cliente recebe e formata os dados para interagir com o usuário. Na lógica distribuída, o cliente é responsável pela interação com o usuário e algumas funções triviais da aplicação. Por exemplo, controles de intervalo de campo, campos obrigatórios, dentre outros. Enquanto o restante do aplicativo, assim como o banco de dados, está no servidor. No gerenciamento remoto de dados, o cliente interage com o usuário e executa a aplicação e o servidor é quem trata os dados. No banco de dados distribuído, o cliente realiza a interação com o usuário, executa a aplicação, deve conhecer a topologia da rede, bem como a disposição e localização dos dados. Para o servidor-cliente em três níveis, o cliente lida com a interação com o

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

usuário, enquanto o servidor de lógica do aplicativo e o banco de dados podem estar em outro servidor (SCOBA et al., 2019).

1.7.4 Protocolo

Protocolo é um conjunto bem conhecido de regras e formatos que são usados para a comunicação entre processos que realizam determinada tarefa. Duas partes são necessárias, sendo estas a especificação da sequência de mensagens a serem trocadas e a especificação do formato dos dados nas mensagens.

Um protocolo permite que componentes heterogêneos de sistemas distribuídos sejam desenvolvidos de forma independente e, por meio dos módulos de software que compõem o protocolo, logo, existe uma comunicação transparente entre os dois componentes. Vale ressaltar que esses componentes do protocolo devem estar tanto no receptor quanto no transmissor.

Para exemplos de protocolos usados em sistemas distribuídos, tem-se (GOMEZ et al., 2018):

I. IP: Protocolo de Internet. - Protocolo da camada de rede, que permite definir a unidade básica de transferência de dados e é responsável pelo endereçamento da informação, de forma que ela chegue a seu destino na rede;
II. TCP: Protocolo de controle de transmissão. - Protocolo de Camada Transporte, o que permite dividir e ordenar as informações a serem transportadas em pacotes menores para transporte e recepção;
III. HTTP: Protocolo de transferência de hipertexto. - Protocolo da camada de aplicação que permite o serviço transferência de páginas de hipertexto entre o cliente de rede e servidores;
IV. SMTP: Simple Mail Transfer Protocol. - Protocolo de Camada aplicativo, que permite o envio de Correio eletrônico através da rede;
V. POP3: Protocolo Escritório de Email. - Protocolo da camada de aplicação, que permite a gestão dos correios eletrônicos na Internet, ou seja, permite que uma estação de trabalho recupere os e-mails armazenados em um servidor.

1.8 Application Programming Interface (API/ Web Service)

A Application Programming Interface - API é um software intermediário que permite que sistemas interajam entre si e disponibiliza interfaces de comunicação através de pontos de acesso. Logo, compreende-se que a API é um conjunto de definições e protocolos usados no desenvolvimento e na integração de software de aplicações (SAHIN; AKAY, 2021).

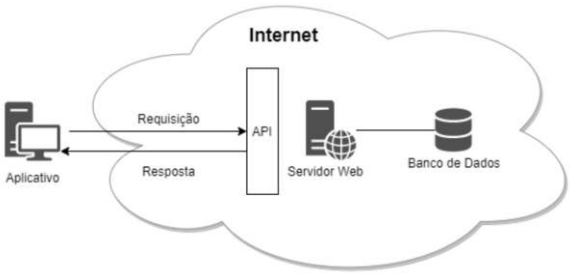
As APIs costumam ser vistas como contratos, com documentações que representam um acordo entre as partes interessadas. Se uma dessas partes enviar uma solicitação remota estruturada de uma forma específica, isso determinará como o software da outra parte responderá (SAHIN; AKAY, 2021).

Libera-se o acesso aos seus recursos, sem abrir mão da segurança e do controle. As APIs web normalmente usam o protocolo HTTP para mensagens de solicitação e fornecem uma definição da estrutura das mensagens de resposta. Essas mensagens de resposta geralmente têm o formato

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

de arquivo XML ou JSON. Tanto XML quanto JSON são formatos de preferência porque apresentam os dados de forma simplificada, o que facilita a manipulação por outras aplicações. A seguir, tem-se uma ilustração da estrutura API (GRAHL et al., 2017).

Figura 5 - Estrutura API



Na arquitetura cliente-servidor todos os dados são armazenados nos servidores, que possuem controles de segurança muito maiores do que a maioria dos clientes. Os servidores podem controlar melhor o acesso aos serviços, por exemplo, garantindo que apenas os clientes com credenciais válidas possam utilizar os serviços oferecidos;

Em nosso projeto, a API utiliza um Banco de Dados relacional na nuvem para persistir os dados. Desta forma, o projeto tem em sua arquitetura, a API (e seu Banco de Dados) atuando como servidor, e atuando como clientes, o sistema legado enviando os dados para a API e os aplicativos nos smartphones consumindo estes dados também através da API.

1.8.1 Arquitetura REST

A arquitetura REST mudou completamente a engenharia de software a partir dos anos 2000. Essa nova abordagem para o desenvolvimento de projetos e serviços web foi definida por Roy Fielding, o pai da especificação HTTP e uma das referências internacionais em tudo relacionado à Arquitetura de Rede, em sua dissertação ”Estilos Arquitetônicos e o projeto de arquiteturas de software baseadas em rede”. No campo das APIs, REST (Representational State Transfer) é, hoje, referência para o desenvolvimento de serviços de aplicação (FERREIRA; OLIVEIRA, 2017).

Atualmente, não existe nenhum projeto ou aplicativo que não possua uma API REST para a criação de serviços profissionais a partir desse software. Twitter, YouTube, sistemas de identificação com Facebook, dentre outros, são várias organizações que geram negócios graças às APIs REST e REST. Sem eles, todo crescimento horizontal seria praticamente impossível. Isso porque o REST é o padrão mais lógico, eficiente e comum na criação de APIs para serviços de Internet (FERREIRA; OLIVEIRA, 2017).

Procurando uma definição simples, REST é qualquer interface entre sistemas que usa HTTP para obter dados ou gerar operações nesses dados em todos os formatos possíveis, como XML e JSON. É uma alternativa crescente a outros protocolos de troca de dados padrão como o SOAP

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

(Simple Object Access Protocol), que tem uma grande capacidade, mas também muita complexidade. Às vezes, uma solução de manipulação de dados mais simples como REST é preferível.

1.8.2 Recursos REST

Para os recursos REST, tem-se o protocolo cliente-servidor sem estado, neste, cada solicitação HTTP contém todas as informações necessárias para executá-la, o que permite que nem o cliente nem o servidor precisem se lembrar de qualquer estado anterior para satisfazê-la. Embora seja esse o caso, alguns aplicativos HTTP incorporam cache. O que é conhecido como protocolo cliente-cache-servidor *stateless*, no qual existe a possibilidade de definir algumas respostas a solicitações HTTP específicas como armazenáveis em cache, com o objetivo de que o cliente possa executar a mesma resposta para solicitações idênticas no futuro. Em qualquer caso, o fato de existir a possibilidade não significa que seja a mais recomendada (FERREIRA; OLIVEIRA, 2017).

Os objetivos em REST são sempre manipulados com base em URI. O URI e nenhum outro elemento é o identificador exclusivo de cada recurso nesse sistema REST. O URI facilita o acesso às informações para modificação ou exclusão, ou, por exemplo, para compartilhar a localização exata com terceiros.

Outro recurso é sua interface uniforme, logo, para transferência de dados em um sistema REST, aplica ações específicas (POST, GET, PUT e DELETE) sobre os recursos, desde que sejam identificados com um URI. Isso facilita a existência de uma interface uniforme que sistematiza o processo com as informações.

Outro recurso é o uso de hipermídia, a hipermídia é uma extensão do conceito de hipertexto. Esse conceito que levou ao desenvolvimento de páginas web é o que permite ao usuário navegar pelo conjunto de objetos por meio de links HTML. No caso de uma API REST, o conceito de hipermídia explica a capacidade de uma interface de desenvolvimento de aplicativos fornecer ao cliente e ao usuário os links apropriados para executar ações específicas nos dados (SAKAMOTO et al., 2018).

1.8.3 API Restful

Compreende-se que a arquitetura REST funciona no protocolo HTTP. Portanto, os procedimentos ou métodos de comunicação são os mesmos do HTTP, sendo os principais: GET, POST, PUT, DELETE. Outros métodos usados na API RESTful são OPTIONS e HEAD. Este último serve para passar parâmetros de validação, autorização e tipo de processamento, entre outras funções.

Outro componente de uma API RESTful é o "Código de status HTTP", que informa ao cliente ou consumidor da API o que fazer com a resposta recebida. Trata-se de uma referência universal de resultados, ou seja, ao projetar uma API RESTful, leva-se em consideração o uso correto do “Código de Status”. Compreende-se que a adoção dos padrões se espalhou rapidamente pela Web e abriu o caminho para seu crescimento contínuo e ampla utilização (PEREIRA et al., 2018).

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

2 O SISTEMA NETDOCTOR

O sistema NetDoctor é um sistema de informação responsável pela gestão de clínicas médicas. Sua operação abrange diversos módulos, que atendem desde o agendamento de pacientes até o faturamento de contas médicas para convênios, passando pelo atendimento médico e ambulatorial, controle de estoque, etc.

Para a integração com o aplicativo móvel, utilizaremos apenas o módulo de agendamento de pacientes, responsável pelo gerenciamento das informações a serem utilizadas no aplicativo.

Pelo sistema NetDoctor o atendente da clínica agenda a consulta do paciente da seguinte forma: selecionando o paciente (cadastrando se for um novo paciente), selecionando um dos profissionais de saúde que atende na clínica, o tipo de atendimento (consulta, procedimento, terapia) e escolhendo a data e hora do agendamento.

As informações necessárias para o registro de um agendamento são:

- Data do Agendamento, Hora do Agendamento, Paciente, Profissional e Tipo de Atendimento. O campo Observação é opcional e é utilizado para que o profissional consulte no momento da consulta. O campo código é um número gerado de forma automática e incremental pelo sistema para cada novo registro incluído e é a chave primária da tabela de agendamentos.
- O campo Hora de Chegada é registrado pelo sistema quando o usuário indica que determinado paciente chegou e o campo Hora de Atendimento é registrado pelo sistema quando o profissional indica que iniciou o atendimento.

As informações relativas ao paciente como: Telefone e Convênio estão atrelados ao paciente através do cadastro de pacientes, que por não ser objeto do estudo, não será abordado.

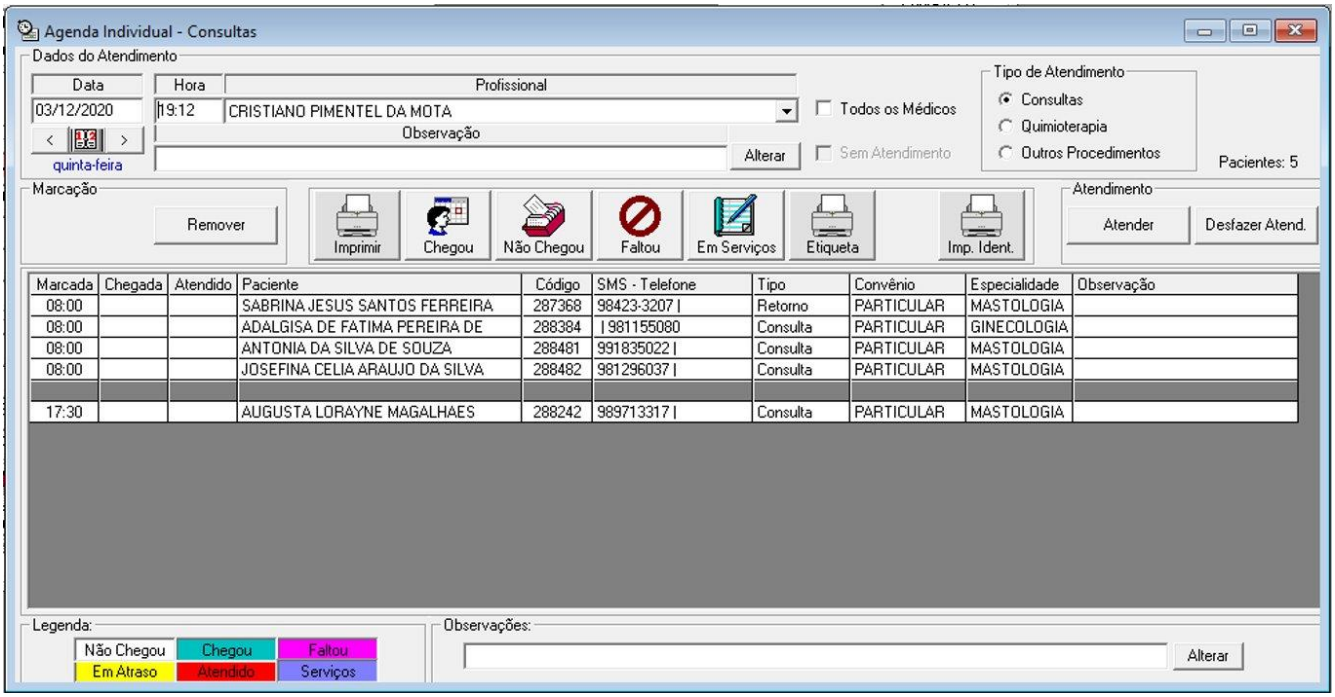
A especialidade do profissional, da mesma forma, está atrelada ao profissional através do cadastro de profissionais, o qual também não é objeto do estudo.

Os tipos de atendimento podem ser: Consulta, Quimioterapia ou Outros Procedimentos.

A figura 6 exibe a agenda diária de um profissional, com seus respectivos agendamentos de pacientes.

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

Figura 6 - Agenda diária do profissional no sistema de gestão médico/hospitalar NetDoctor



O sistema legado que pretender usar a solução, em nosso caso de uso o sistema NetDoctor, deverá enviar para a Web API um arquivo no formato JSON, através do método POST, contendo os dados mínimos necessários e seguindo os padrões definidos pela API, como apresentado no capítulo 5, figura 6. São eles: Data do Agendamento, Hora do Agendamento, Nome do Paciente, Convênio, Profissional e Tipo de Atendimento.

2.1 A solução proposta

A partir do que foi apresentado e analisado, é proposto a implementação de uma Web API Restful que servirá de provedor de informações para um aplicativo mobile.

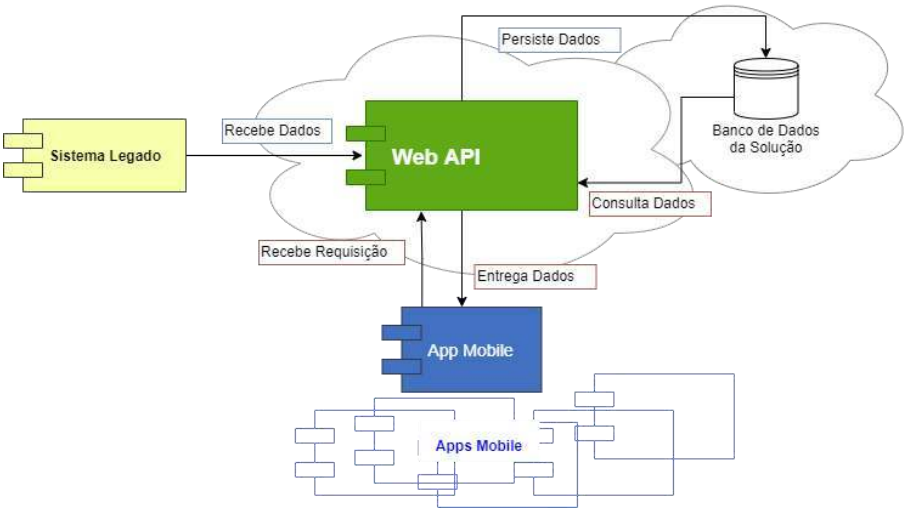
O sistema NETDOCTOR se comunica com a API via requisições HTTP enviando informações via método post e API se encarregará de persistir estes dados de forma segura em um banco de dados on-cloud e proverá os mesmos também via HTTP para o aplicativo mobile.

Este Capítulo mostra o projeto e a implementação da solução que serve de estudo de caso para demonstrar o uso de serviços web, utilizando a plataforma Android no lado cliente e na API Restful implementada em ASP.NET com C# no lado do servidor.

Foi utilizado o Diagrama de componentes definido pela UML (Unified Model Language) para a documentação relacionada à engenharia de software do projeto e para proporcionar uma melhor comunicação entre os membros da equipe (ver fig. 7).

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

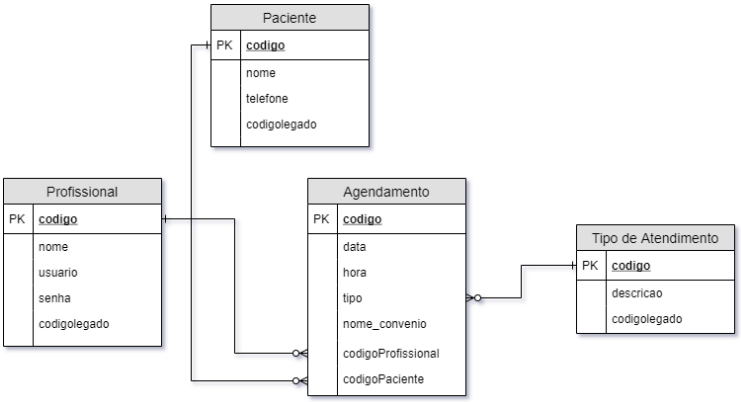
Figura 7 – Diagrama de componentes



No diagrama de componentes (fig. 7) podemos observar a arquitetura da solução e a e os módulos ou componentes que fazem dele, assim como seus relacionamentos.

O componente principal é a Web API, hospedada na nuvem, que é utilizada pelo sistema legado e pelo aplicativo mobile para acessar os dados. A origem dos dados provém do sistema legado foco do estudo, dados estes que são enviados através de requisições autenticadas para a Web API, que por sua vez, realiza o processamento dos dados e então os persiste em um sistema gerenciador de banco de dados hospedado na nuvem. A API também é responsável por entregar as informações para os requisitantes externos, como por exemplos usuários em sistemas móveis.

Figura 8 – Diagrama Entidade Relacionamento do Banco de Dados na estrutura on-cloud.



De todo o banco de dados do sistema legado, foram utilizados no estudo de caso somente os dados necessários para a requisição do objeto do estudo, o App Mobile, que apresenta a Agenda do Profissional. No Diagrama de Entidade Relacionamento visualizamos a estrutura do Banco de Dados Relacional onde serão persistidos os dados.

Três Tabelas armazenam os dados das Entidades envolvidas no agendamento:

- Paciente;

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

- Profissional;
- Tipo de Atendimento;
- Uma tabela armazena a transação “Agendamento”

2.2 Levantamento de Requisitos

Esta parte do documento explica os conceitos de termos importantes que serão citados no decorrer deste documento, conforme descrito na tabela abaixo.

Quadro 2 – Levantamento de Requisitos.

Termo	Descrição
Requisitos funcionais	Requisitos de software que compõem o sistema, descrevendo ações que o sistema deverá executar quando solicitado.
Requisitos não funcionais	Requisitos de software que compõem o sistema, descrevendo atributos de qualidade que o sistema deve possuir, ou restrições que ele deve satisfazer.

No levantamento de requisitos funcionais, foi definido que:

I. Aplicativo deverá possuir uma tela de identificação do usuário, com nome ou código de identificação e senha.

II. O aplicativo deverá armazenar um *token* gerado de acordo com as informações de usuário e senha para que realize a autenticação em cada requisição, desobrigando o profissional a sempre digitar seu usuário e senha a cada execução do aplicativo.

III. O aplicativo deverá ter uma opção para que o aplicativo “esqueça” os dados armazenados de usuário e senha, de forma a obrigar a digitação destes dados na próxima execução.

IV. Após a autenticação do usuário, o aplicativo deverá exibir a tela de agendamentos na data de hoje.

2.3 Tela de Agendamento

Quanto a requisitos funcionais baseados na tela de agendamento, foi definido que:

I. Deverá exibir no topo da tela as seguintes informações:

II. Nome do Profissional (usuário), Data do Agendamento (em formato dd/mm/aaaa e Dia da Semana).

III. A tela de agendamento deverá exibir as seguintes informações em formato de lista:

i. Hora do Agendamento

ii. Nome do Paciente

iii. Convênio

iv. É Retorno

IV. Na tela de agendamento o aplicativo deve exibir o total de agendamentos do dia.

V. O Aplicativo deverá permitir que o usuário selecione qualquer data através de um calendário.

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

VI. O Aplicativo deverá permitir que o usuário avance ou retroceda em um dia com apenas um clique em botões exibidos à esquerda e à direita da data atualmente selecionada.

VII. Deverá haver um botão para atualizar as informações da tela.

VIII. Caso não haja agendamentos no dia selecionado, o aplicativo deverá exibir a mensagem “Não há pacientes nesta data”

2.4 Requisitos Não Funcionais

Quanto a análise de requisitos não funcionais, foi definido que:

I. O aplicativo deverá ter uma interface limpa, destacando as informações mais relevantes através de fontes com cores e tamanhos diferenciados.

II. Durante a carga de dados da tela de agendamento do dia, o aplicativo deverá apresentar um ícone de “carregamento” informando ao usuário que os dados estão sendo atualizados.

III. O aplicativo deverá ser leve o suficiente para ser executado sem lentidão nos aparelhos com as configurações mais comuns do mercado.

IV. O fundo da agenda deverá ser branco para facilitar a leitura das informações

2.5 A Implementação da Web Api Restful

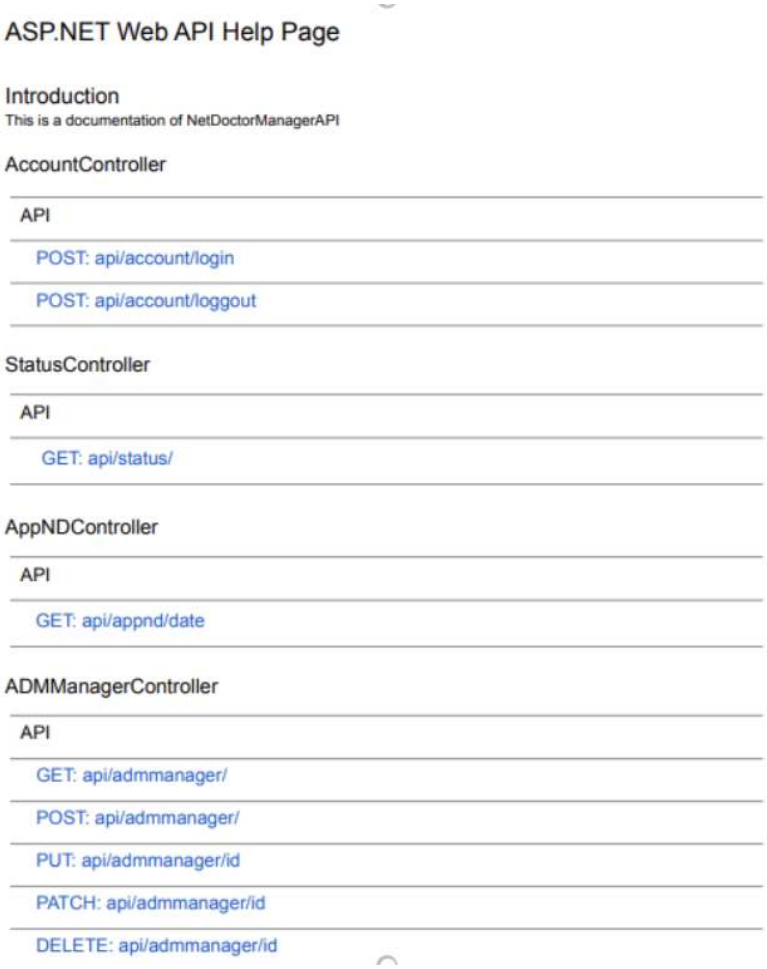
A solução foi desenvolvida com base em um fluxo de informações unidirecional, onde apenas o sistema *on-premise* é responsável por enviar os dados para o sistema on-cloud através de requisições aos endpoints privados da API. O aplicativo móvel realiza apenas consultas das informações no banco de dados on-cloud através de requisições aos endpoints públicos da API específicos para consulta.

A API foi desenvolvida de forma que suas rotas recebem requisições baseadas no protocolo HTTP e os comportamentos das rotas são definidos de acordo com o verbo HTTP[3] informado na requisição, bem como definido no padrão de arquitetura REST. A seguir é evidenciada na figura 9 a implementação da Web API Restful.

[3] O protocolo HTTP define um conjunto de métodos de requisição responsáveis por indicar a ação a ser executada para um dado recurso. São comumente referenciados como HTTP Verbs (Verbos HTTP). Exemplos de Verbos HTTP: GET, POST, DELETE.

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

Figura 9 – Implementação da Web APi Restful (configuração)



Na API há uma rota/controller chamada admmanager que é utilizada pelo sistema Netdoctor para inserir, atualizar e excluir os agendamentos de consultas do banco de dados on-cloud. Através do método POST nesta chamada, é possível fazer a inserção de uma tupla na tabela de agendamentos, utilizando o método DELETE é possível deletar um agendamento. Da mesma forma, utilizando o método PUT é possível fazer a atualização completa de uma tupla e por fim com o método PATCH é possível alterar um determinado campo, como apresentado na figura 9 a seguir.

Para o aplicativo móvel, com o objetivo de autenticar o usuário, temos a rota/controller account, que pode ser requisitada com o verbo POST e possui duas rotas nomeadas, login e loggout, que operam da seguinte forma: a rota login recebe via parâmetros o usuário e senha do profissional que está entrando no aplicativo móvel e retorna um token de autenticação; a rota logout recebe apenas o token do usuário para removê-lo da base de autenticação, fazendo assim o encerramento da validade deste token para requisições.

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

Para a consulta de informações, temos a rota/controller appnd que recebe as requisições através do método GET, recebe como parâmetro uma querystring[4] contendo a data para a qual se deseja receber as informações, e retorna com o arquivo JSON contendo os dados referentes às consultas agendadas nesta data, como apresentado na figura 9. Para possíveis consultas relacionadas à disponibilidade da API, foi implementado uma rota/controller chamada status, que recebe uma requisição através do método GET e retorna o status de funcionamento da API.

A API retorna dados em formato de texto no modelo de armazenamento chamado JSON, que serão consumidos pelo aplicativo móvel. O JSON é um modelo de apresentação em que os dados são dispostos em conjuntos de objetos nomeados prosseguido com pares nome/valor, sendo que a resposta da requisição é um objeto que possui outros objetos também dispostos em nome/valor. A figura 10 exemplifica o retorno da requisição ao endpoint responsável por enviar ao aplicativo móvel os dados referentes à agenda de um profissional em um determinado dia.

Figura 10 – Exemplo de retorno de uma agenda médica notada em JSON.

```
{
  "totalOfSchedulesToday" : 1,
  "schedules" : [
      {
        "patient" : "Example Name",
        "healthInsurance": "Medexample",
        "status": "In progress",
        "scheduledTime": "09:00",
        "isReturn": true
      },
   ]
}
```

2.6 A Implementação da App Mobile

Para implementação do aplicativo mobile, foi utilizado o framework Flutter disponibilizado pela Google em 2018, que possibilita a criação de apps multiplataformas mantendo o desempenho de apps nativos. Flutter é um kit de ferramentas de IU do Google para construir aplicativos bonitos e nativamente compilados para dispositivos móveis, web, desktop e incorporados a partir de uma única base de código (FLUTTER, 2021).

O aplicativo foi desenvolvido com base no levamento de requisitos realizado e, portanto, permite que usuários com determinado perfil profissional, que utilizam o sistema NETDOCTOR, acessem informações em tempo real através da Web API que originalmente estariam armazenados no banco de dados local do sistema legado.

Para utilizar o aplicativo móvel, o profissional insere seu usuário e sua senha nos respectivos campos da tela de autenticação no aplicativo que então cria uma chave Hash MD5 de ambas as informações e as envia em uma requisição via método POST/HTTP para a API, que por sua vez faz

[4] Uma querystring é uma parte de uma URL (uniform resource locator) que atribui valores a parâmetros especificados. Uma querystring geralmente inclui campos adicionados a um URL por um navegador da Web ou outro aplicativo cliente, por exemplo, como parte de um formulário HTML.

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

as devidas validações e caso as informações enviadas estejam de acordo com os presentes no banco de dados on-cloud, a API retorna uma resposta com o código HTTP 200[5], além de um token de acesso que é persistido localmente e é utilizado nas futuras requisições do aplicativo para a API de forma a validar o acesso do usuário naquela sessão.

Uma vez autenticado, o profissional tem acesso à sua agenda de consultas e informações relacionadas que são relevantes para o seu trabalho, informações estas que foram diretamente enviadas pelo sistema Netdoctor e estão disponíveis via à Web API desenvolvida.

Na figura 11 é demonstrada a tela inicial do aplicativo mobile onde o profissional deve digitar seu usuário e senha previamente cadastrados no banco de dados e também é demonstrada a tela onde são apresentadas ao profissional autenticado no aplicativo, as informações retornadas da API referentes a sua agenda diária.

Figura 11: Telas do aplicativo móvel: Autenticação e Aplicação

No próximo capítulo, serão apresentados os resultados obtidos com o desenvolvimento da solução proposta neste trabalho.

3 RESULTADOS

A NBR ISO 27002 define políticas que devem ser seguidas para preservar aspectos de segurança da informação, que conforme já discutido neste trabalho, podem ser utilizadas de guia para o desenvolvimento de novas funcionalidades para sistemas legados.

[5] O código HTTP 200 OK é a resposta de status de sucesso que indica que a requisição foi bem-sucedida.

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

No capítulo 14 é descrito considerações de segurança da informação que devem estar presentes como parte integrante de todo o ciclo de vida de um projeto. Segundo a NBR ISO 27002, as necessidades de proteção requeridas dos ativos envolvidos, devem considerar em particular com relação à disponibilidade, confidencialidade e integridade.

Essas políticas serão a base da análise do protótipo. As características implementadas serão analisadas individualmente tendo como possíveis resultados: atende, atende parcial, não atende. Esses comparativos serão expostos e ao fim deste capítulo será possível responder à pergunta problema norteadora deste trabalho.

3.1 Disponibilidade

O fator disponibilidade está relacionado à capacidade de acessar as informações dos sistemas da empresa ao longo do tempo, em outras palavras, podemos considerar que há disponibilidade se os dados podem ser consultados a qualquer momento pelos colaboradores.

Em estruturas *on-premises*, uma das características principais é que os requisitantes precisam estar dentro da estrutura local da empresa, e em consequência em sua rede, para ter acesso às informações.

Com a implementação da arquitetura de *software* em uma estrutura on-cloud, foi possível apresentar uma maior disponibilidade das informações, uma vez que com a solução proposta, as informações passam a estar disponíveis a qualquer requisitante com acesso à Internet. Na figura 12 pode ser observado a implementação de um endpoint que retorna o status atual da API.

Figura 12 – Endpoint público que retorna o status de atividade da API.

```
1 public class StatusController : ApiController
2 {
3     public HttpResponseMessage GetStatusOfAPi()
4     {
5         .. .
6     }
7 }
```

É importante ressaltar que a arquitetura implementada foi projetada para que este aumento na disponibilidade não resultasse em aumento na exposição ao risco da segurança da informação do Banco de Dados.

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

3.2 Confidencialidade

De modo a garantir a confidencialidade dos dados trafegados na comunicação entre o sistema legado e o aplicativo mobile, a API busca atender a NBR ISO 27002:2013 conforme descrito no capítulo 14 Aquisição, desenvolvimento e manutenção de sistemas:

No capítulo 14.1 Requisitos de segurança de sistemas de informação, subtópico Serviços de aplicação seguros em redes públicas, temos em Diretrizes para implementação:

Convém que as considerações de segurança da informação para serviços de aplicação que transitam sobre redes públicas considerem os seguintes itens:
a) o nível de confiança que cada parte requer na identidade alegada, como, por exemplo, através de autenticação;

Visando atender a essa diretriz, foram implementadas algumas medidas como por exemplo a criação de um token de acesso gerado em tempo real para cada conexão em toda e qualquer requisição à API, que para ser gerado exige a autenticação do usuário através do uso de usuário e senha que estão persistidos no banco de dados com o uso de criptografia *Hash* MD5.

Figura 13 – Exemplo de endpoints na linguagem C#, no qual o primeiro é público e o segundo o usuário precisa estar autenticado.

```
public class AppNDController : ApiController
{
    public HttpResponseMessage DoLogin()
    {
        .. .
    }

    [Authorize]
    public HttpResponseMessage DoLogout()
    {
        .. .
    }

    [Authorize]
    public HttpResponseMessage GetListSchedule()
    {
        .. .
    }
}
```

Na figura 13 é exemplificado o controlador da API, onde pode-se verificar que a existência de um método público, ou seja, que não exige autenticação e dois métodos privados, que exigem a autenticação do usuário.

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

Figura 14 – Código para criptografar uma senha em MD5 em C#

```
using System.Text;
using System.Security.Cryptography;

namespace CryptoLib
{
    public static class Encryptor
    {
        public static string MD5Hash(string text)
        {
            MD5 md5 = new MD5CryptoServiceProvider();
            md5.ComputeHash(ASCIIEncoding.ASCII.GetBytes(text));
            byte[] result = md5.Hash;
            StringBuilder strBuilder = new StringBuilder();
            for (int i = 0; i < result.Length; i++)
            {
                strBuilder.Append(result[i].ToString("x2"));
            }

            return strBuilder.ToString();
        }
    }
}
```

Na figura 14 é possível verificar a classe implementada responsável por realizar a criptografia MD5 de um texto passado por parâmetro, o qual é retornado já criptografado. Portanto, a solução proposta atende a este aspecto.

3.3 Integridade

Como apresentado anteriormente, todo acesso aos dados é realizado através da API, que isola cada tipo de requisição de acesso através dos endpoints específicos para cada operação, como as de inclusão, exclusão, alteração e consulta.

Visando manter a integridade das informações persistidas no banco on-cloud que são recebidas do sistema *on-premise*, são utilizadas validações na implementação de forma que as informações trafegam do sistema legado para API, afirmando o sincronismo entre os bancos.

Para as requisições de inclusão, alteração e exclusão, que realizam a persistência de dados, o único perfil de requisitante que possui permissão de acesso aos respectivos endpoints é o perfil do sistema legado, portanto temos a garantia de que os dados estarão sempre íntegros.

A figura 15 abaixo demonstra um controlador da API que só pode ser acessado por um usuário com perfil "NetdoctorHoldes" que é o perfil delegado ao sistema legado.

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

Figura 15 – Controlador com acesso exclusivo para o perfil do sistema legado em C#.

```
[Authorize(Roles = "NetdoctorHoldes")]
public class ADMManagerController : ApiController
{
    public HttpResponseMessage SincCalendar()
    {
        .. .
    }

    public HttpResponseMessage ImportCalendar()
    {
        .. .
    }
}
```

Os endpoints públicos que podem ser acessados por hosts não conhecidos, desde que devidamente autenticados, permitem somente a consulta a alguns dados, como o número total de pacientes agendados no dia, o convênio, e etc, de modo que é garantida a integridade das informações, pois não há a possibilidade de realizar alterações no banco de dados on-cloud.

A figura 16 exibe a implementação do método que valida se o IP de origem da requisição recebida é o autorizado pela API.

Compreende-se que o fluxo unidirecional da informação implementada pode ocasionar perda de sincronia entre o banco de dados na nuvem e o banco de dados legado, motivado por causas externas ou desconhecidas. Como solução, propõe-se uma interface de sincronização na API desenvolvida, que fará a comparação, com uma frequência, das tuplas existentes de ambos os bancos obedecendo o fluxo da informação.

Assim, esta solução garante a integridade do banco hospedado em estruturas externas à empresa proprietária do software ou utilizadora do serviço. Intervenções no SGBD tipificadas escritas indevidas serão resolvidas com o atraso estipulado na frequência de execução do procedimento proposto.

Parametrizar a frequência de execução, não é uma tarefa simples. O aumento da frequência pode ocasionar um “overload” dos recursos computacionais fazendo com que o sistema fique indisponível interferindo na classificação do sistema perante a NBR ISO 27002. A baixa frequência implica em ter um procedimento desenvolvido que não proporciona a solução foco do problema.

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

Figura 16 – Função para validação se o IP requisitante é reconhecido pela API em C#.

```
public static bool IsValidIP(this HttpRequestMessage request)
{
    var whiteListedIPs = ConfigurationManager
        .AppSettings["WhiteListedIPAddresses"];
    if (!string.IsNullOrEmpty(whiteListedIPs))
    {
        var whiteListIPList = whiteListedIPs.Split(',').ToList();
        var ipAddressString = request.GetIP();
        var ipAddress = IPAddress.Parse(ipAddressString);
        var isInwhiteListIPList = whiteListIPList
            .Where(a => a.Trim()
            .Equals(ipAddressString, StringComparison.InvariantCultureIgnoreCase))
            .Any();
        return isInwhiteListIPList;
    }

    return true;
}
```

Pelo motivo exposto não será parametrizado a frequência e o processo de sincronização não será tratado neste trabalho. Assim, a não integridade do banco on-cloud ocasionado por escritas indesejadas nas tuplas, não serão discutidas e retiradas dos requisitos sistemas. Desta forma, a solução proposta atende a NBR ISO 27002 no quanto a integridade.

CONSIDERAÇÕES FINAIS

Com a problemática suscitada frente à busca por compreender como o fluxo de informação entre a estrutura *on-premises* e cloud é construído preservando os aspectos legados, observou-se que o emprego de computação em nuvem tende a transformar de forma viável o desenvolvimento de softwares baseados em arquitetura orientada a serviços, para o intercâmbio de informações entre sistemas.

No desenvolvimento do estudo de caso, a fim de se destacar os aspectos estudados referentes à segurança da informação e à utilização de serviços web, verificou-se a viabilidade de oferecer a solução de forma segura, ágil, escalável e com baixo custo de implementação, operação e manutenção a partir de uma solução baseada em uma arquitetura multicamadas utilizando chamadas de serviços web.

Foram atendidos os objetivos do trabalho, destacando a teorização da segurança de informação em sistemas *on-premises* legados, bem como a segurança de informação em sistemas on-cloud, atendendo os aspectos mencionados na NBR ISO 27002.

Ao compreender a forte influência da virtualização de data centers nos dias de hoje, sugere-se, para trabalhos futuros, um aprofundamento sobre gerenciamento e proteção de dados com enfoque no ecossistema tecnológico, pincipalmente na sincronização entre os sistemas gerenciados de banco de dados.

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

REFERÊNCIAS

ABNT ISO 27002. ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. **NBR/ISO/IEC 27002**; 2013 Tecnologia da informação – técnicas de segurança – código de prática para controles de segurança da informação.

BÉRGAMO, Luciano; OLIVEIRA, Vitor Coimbra de. Big Data em Nuvem. **SITEFA-Simpósio de Tecnologia da Fatec Sertãozinho**, v. 2, n. 1, p. 404-411, 2019.

BHARGAVA, Sandeep; GOYAL, Drdinesh; KESWANI, Bright. Performance Comparison of Big Data Analytics Platforms. **International Journal of Engineering, Applied and Management Sciences Paradigms (IJEAM)**, 2019.

CHEROBINI, Tatiana Minuzzi. **Terceirização de serviços de TI**: aspectos de segurança. 2017.

CIPRIANO, Wellington Ferreira. **A segurança da informação com o advento da internet das coisas em ambientes hospitalares:** uma abordagem bibliográfica. 2021. Trabalho de Conclusão de Curso (Especialização) – Curso Gestão. Assessoramento e Estado-Maior, Escola de Formação Complementar do Exército, 2021.

COELHO, Flávia Estélia Silva et al. **Avaliação de disponibilidade de redes de computadores baseadas na arquitetura cliente/servidor em n-camadas**. 2000. 140f. Dissertação (Mestrado em Informática) - Pós-Graduação em Informática, Centro de Ciências e Tecnologia, Universidade Federal da Paraíba, Campus II, Campina Grande - Paraíba, 2000.

FERREIRA, Willian Ottoni; OLIVEIRA; Knop. Igor. Estruturação de Aplicações Distribuídas com a Arquitetura REST. **Caderno de Estudos em Sistemas de Informação**, v. 3, n. 1, 2017.

FISHER, Cameron et al. Cloud versus on-premise computing. **American Journal of Industrial and Business Management**, v. 8, n. 09, p. 1991, 2018.

FLUTTER. **Flutter**: Beautiful native apps in real time. Página inicial. Disponível em: https://flutter.dev/. Acesso em: 20 maio 2021.

GOMEZ, Carles; ARCIA-MORET, Andrés; CROWCROFT, Jon. TCP in the Internet of Things: from ostracism to prominence. **IEEE Internet Computing**, v. 22, n. 1, p. 29-41, 2018.

GRAHL, M. et al. Archive WEB API: A web service for the experiment data archive of Wendelstein 7-X. **Fusion Engineering and Design**, v. 123, p. 1015-1019, 2017.

JORGENSEN, Adam et al. **Microsoft Big Data Solutions**. Nova Jersey: John Wiley & Sons, 2014.

MAESTRI, Gabriela et al. **Indústria 4.0 no Setor Têxtil**: Diagnóstico Atual, Desafios e Oportunidades para o Futuro Digital. 2018. TCC (Graduação) - Universidade Federal de Santa Catarina, Blumenau, 2018.

NAKKEERAN, Ms Amizhthini; NIRANGA, Ms Mahikala; WICKRAMARACHCHI, Ruwan. **A Model for On-Premises ERP System and Cloud ERP Integration**. Dubai: Proceedings of the International Conference on Industrial Engineering and Operations Management, 2020. Avaliable in: http://www.ieomsociety.org/ieom2020/papers/575.pdf. Acessado em 23 jun, 2021.

OLIVEIRA, Thiago Rodrigues. Implantação de políticas de segurança da informação em uma pequena empresa. **Revista Eletrônica de Sistemas de Informação e Gestão Tecnológica**, v. 8, n. 1, 2017.

RECIMA21 - REVISTA CIENTÍFICA MULTIDISCIPLINAR
ISSN 2675-6218

PADILHA, Thais Cássia Cabral; MARINS, Fernando Augusto Silva. Sistemas ERP: características, custos e tendências. **Production**, v. 15, n. 1, p. 102-113, 2005.

PALANIMALAI, Shanmugasundaram. A Hybrid Integration Strategy for On-premises and Cloud based Solutions. **International Journal of Pure and Applied Mathematics**, v. 118, n. 18, p. 2079-2087, 2018.

PEREIRA, Adan Lucio et al. Computação em nuvem: a segurança da informação em ambientes na nuvem e em redes físicas. **Brazilian Journal of Production Engineering-BJPE**, p. 12-27, 2016.

PEREIRA, André Luiz Kuczner et al. Abordagem Restful Em Serviços Web. **Revista Gestão em Foco,** n. 10, Ano. 2018. Disponível em: https://portal.unisepe.com.br/unifia/wp-content/uploads/sites/10001/2018/12/032-ABORDAGEM-RESTFUL-EM-SERVI%C3%87OS-WEB.pdf. Acesso em: 24 Jun. 2021.

PEREIRA, Thiago Martins; SACILOTTI, Adani Cusin; JÚNIOR, José Roberto Madureira. Computação Em Nuvem: Plataforma Como Serviço. **Fundamentos da Ciência da Computação**, v. 2, p. 116-125, 2019.

RAMACHANDRA, Gururaj; IFTIKHAR, Mohsin; KHAN, Farrukh Aslam. A comprehensive survey on security in cloud computing. **Procedia Computer Science**, v. 110, p. 465-472, 2017.

SAHIN, Omur; AKAY, Bahriye. A Discrete Dynamic Artificial Bee Colony with Hyper-Scout for RESTful web service API test suite generation. **Applied Soft Computing**, v. 104, p. 107-46, 2021.

SAKAMOTO, Mário et al. **Melhoria contínua da qualidade no processo de produção da camada Back-end de sistemas web em arquitetura REST**. 2018. 70f. Dissertação (Mestrado em Inovação Tecnológica) - Programa de Mestrado Profissional em Inovação Tecnológica, Universidade Federal do Triângulo Mineiro, Uberaba, 2018.

SANTOS, Mateus da Silva; MARCON, Daniel Stefani. Pesquisa Experimental sobre Ataques Cibernéticos em Infraestruturas de Nuvens Públicas Baseadas em Microsoft Azure. **Revista Eletrônica Argentina-Brasil de Tecnologias da Informação e da Comunicação**, v. 3, n. 1, 2020.

SANTOS, Tiago. **Fundamentos da computação em nuvem**. São Paulo: Senac, 2018.

SCOBA, A. N. et al. The optimal placement of information resources on the nodes of a distributed information processing system based on a two-tier and three-tier client-server architecture. *In.:* **IOP Conference Series: Materials Science and Engineering**. IOP Publishing, 2019. p. 012-058.

SILVA, Maria Patrícia Holanda da. **Uma metodologia para melhorar a segurança em ambientes de computação em nuvem**: estudo de caso. 2019. 41 f. TCC (Graduação em Redes de Computadores) - Universidade Federal do Ceará, Campus de Quixadá, Quixadá, 2019.

TAURION, Cezar. **Cloud computing-computação em nuvem**. São Paulo: Brasport, 2009.

TEIXEIRA, Márcio Andrey. **Arquitetura Cliente/Servidor**. Catanduva, SP: [S.n.], 2019.

TSAREGORODTSEV, Anatoly V. et al. Information Security Risk Estimation For Cloud Infrastructure. International **Journal on Information Technologies & Security**, v. 10, n. 4, 2018.

WAN, Zhitao; DUAN, Lihua; WANG, Ping. Cloud migration: layer partition and integration. *In.:* **2017 IEEE International Conference on Edge Computing (EDGE).** IEEE, 2017. p. 150-157.