UNIVERSIDADE ESTADUAL PAULISTA "JÚLIO DE MESQUITA FILHO"
FACULDADE DE ENGENHARIA
CAMPUS DE ILHA SOLTEIRA

ANA KARINA VIEIRA DA SILVA

ACESSO REMOTO SEGURO DE EXPERIMENTAÇÃO EM TEMPO REAL

Ilha Solteira

2014

ANA KARINA VIEIRA DA SILVA

ACESSO REMOTO SEGURO DE EXPERIMENTAÇÃO EM TEMPO REAL

Dissertação de Mestrado apresentado à Faculdade de Engenharia de Ilha Solteira – UNESP como parte dos requisitos para a obtenção do título de Mestre em Engenharia Elétrica. Área de concentração: Automação.

Orientador: Prof. Dr. Luís Carlos Origa de Oliveira
Co-orientadora: Profª. Dra. Christiane Marie Schweitzer

Ilha Solteira
2014

FICHA CATALOGRÁFICA
Desenvolvido pelo Serviço Técnico de Biblioteca e Documentação

S586a

Silva, Ana Karina Vieira da.
Acesso remoto seguro de experimentação em tempo real / Ana Karina Vieira da Silva. -- Ilha Solteira: [s.n.], 2014
88 f. : il.

Dissertação (mestrado) - Universidade Estadual Paulista. Faculdade de Engenharia de Ilha Solteira. Área de conhecimento: Automação, 2014

Orientador: Luís Carlos Origa de Oliveira
Co-orientador: Christiane Marie Schweitzer
Inclui bibliografia

1. Engenharia de software. 2. Acesso remoto. 3. Análise e desenvolvimento de sistemas web.

unesp

UNIVERSIDADE ESTADUAL PAULISTA
CAMPUS DE ILHA SOLTEIRA
FACULDADE DE ENGENHARIA DE ILHA SOLTEIRA

CERTIFICADO DE APROVAÇÃO

TÍTULO: Acesso Remoto Seguro a Experimentação em Tempo Real

AUTORA: ANA KARINA VIEIRA DA SILVA
ORIENTADOR: Prof. Dr. LUIS CARLOS ORIGA DE OLIVEIRA
CO-ORIENTADORA: Profa. Dra. CHRISTIANE MARIE SCHWEITZER

Aprovada como parte das exigências para obtenção do Título de Mestre em Engenharia Elétrica , Área: AUTOMAÇÃO, pela Comissão Examinadora:

Christiane Marie Schweitzer
Profa. Dra. CHRISTIANE MARIE SCHWEITZER
Departamento de Matemática / Faculdade de Engenharia de Ilha Solteira

Profa. Dra. ERICA REGINA M D MACHADO
Departamento de Matemática / Faculdade de Engenharia de Ilha Solteira

Prof. Dr. CRISTIANO MACIEL
Instituto de Computação / Universidade Federal de Mato Grosso

Data da realização: 29 de agosto de 2014.

DEDICO

Ao meu pai **Maurico José da Silva** e a minha mãe **Silvani Aparecida Vieira**, que me educaram sempre me mostraram que o importante é correr atrás do que quer ser honesta, justa, solidária e com essa educação me possibilitam mais essa conquista, exemplos de vida tanto profissional como pessoal.

AGRADECIMENTOS

Primeiramente a DEUS, pois ele que nas horas difíceis sempre me mostra para onde seguir e que decisão tomar.

Aos Meus Pais, Maurico José da Silva e Silvani Aparecida Vieira, por me orientar, educar, auxiliar e me acolher, quando preciso e ao apoio nessa etapa.

Aos Meus irmãos, Mennes Vieira da Silva, Denis Aparecido Vieira da Silva, Enis Vieira da Silva e Lara Vitória Teles Santos, por todo carinho e companheirismo.

Aos Professores Dr. Luís Carlos Origa de Oliveira e a Professora Dra. Christiane Marie Schweitzer, pela a orientação na elaboração deste trabalho. Pela compreensão, dedicação e paciência, no decorrer do desenvolvimento desse trabalho.

Aos amigos, cada um com sua importância, que sempre estão prontos para ajudar, em qualquer momento da minha vida. Principalmente os momentos difíceis que passei até chegar aqui, mesmo longe se tornaram presentes quando necessário, oferecendo seu ombro amigo. Obrigada!

Aos que tentaram interromper minha caminhada e meus objetivos, pois com isso fiquei mais forte, e segui em frente com garra e determinação.

E ressalto a imensa importância de ter encontrado apoio, a amizade, o companheirismo e o carinho de Gabriela Christal, Raiane Piacente, Juciene Toniol, Fabiana Bertolo e Maria Aparecida Pinto, pessoas que estiveram do meu lado por todo esse período. Muito Obrigada, suas Lindas!

Em especial ao Rodrigo Baron, Rodrigo Oliveira, Maurício Eduardo Chicupo o Jithin Joseph Swathish Thazhath Johnce e Fernando Silva Moraes, por que foi essencial a ajuda de cada um de vocês no processo de desenvolvimento deste projeto. Grata sempre por toda atenção.

Agradeço a UNESP, pela estrutura oferecida para o desenvolvimento do trabalho.

A CAPES, pelo apoio financeiro para a realização do projeto.

À banca examinadora que analisou este trabalho, e com suas sugestões contribui para o melhoramento do mesmo.

Agradeço a equipe UCDB, meu local de trabalho, onde encontrei colegas que em tão pouco tempo me acolheram, mostrando compreensão, solidariedade, apoio na etapa finalização do Mestrado.

“J m is considere seus estudos como uma obrigação, e sim como uma oportunidade invejável para aprender a conhecer a influência libertadora da beleza do reino do espírito, para seu próprio prazer pessoal e para proveito da comunidade à qual seu futuro tr lho pertencer.” Al e rt Einsten

RESUMO

O acesso remoto a aplicações e serviços proporciona comodidade e agilidade para experimentação remota em tempo real. Ao pesquisador, permite acesso e controle de instrumentos e aplicações experimentais por meio da Internet. Isso vem se tornando possível, pois existem várias tecnologias de redes que vêm sendo desenvolvidas e aplicadas, de forma a garantir o melhor desempenho e segurança dos serviços implementados.

O LQEE [Laboratório de Qualidade de Energia Elétrica] do DEE [Departamento de Engenharia Elétrica] de Ilha Solteira disponibiliza a realização de experimento em tempo real, aos usuários do LQEE, de forma simples de acessar, apenas pelo o navegador de internet. Embora a administração e o monitoramento desse serviço são feito de forma manual gerando desconforto e insegurança aos responsáveis pelo o LQEE e os experimentos.

Assim, o presente trabalho tem por objetivo aplicar uma solução por meio de técnicas e ferramentas para acesso remoto de aplicações e de instrumentos de experimentação. Esta solução define uma infraestrutura básica de acesso, de forma segura e confiável. Diversas técnicas foram investigadas, e algumas delas incorporadas na solução. Além do acesso remoto, o gerenciamento e controle dos usuários e experimentos se fazem necessário. Para isto, uma aplicação Web foi projetada, a qual permite o acesso remoto seguro e controlado a mobilidade, possibilitando o usuário acessar o serviço de experimentação de qualquer dispositivo e de qualquer lugar via acesso a Internet. O sistema de controle e gerenciamento de usuários e experimentação foi desenvolvido em Java, por ser uma linguagem multiplataforma e sua implantação testada com uma infraestrutura cliente–servidor, composta de dois servidores de acesso às aplicações e clientes remotos, com diferentes dispositivos.

Palavras-chave: Segurança. Acesso remoto. Internet. Engenharia de software.

ABSTRACT

The remote access to applications and services provides convenience and agility for real-time remote experimentation. To researcher is enabled the access and control of instruments and experimental applications through the Internet. And this has become possible because there are several technologies of networks that have been developed and applied in order to ensure optimal performance and security of the services implemented. The LQEE [Laboratory of Electric Energy Quality] of DEE [Department of Electrical Engineering] of Ilha Solteira provides the realization of real time experiments, in order to provide to LQEE users simple access through Internet browser. The administration and monitoring of this service has been done manually, generating discomfort and insecurity to the responsible for the LQEE and the experiments. Thus, the present work aims to apply a solution by means of techniques and tools for remote access applications and experimental instruments. This solution defines a basic infrastructure, access safely and reliable. Several techniques have been investigated, and some of them incorporated into the solution. In addition, remote access, management, and control of users and experiments are necessary. For this, a Web application is designed, which allows secure remote access and controlled mobility, enabling the user to access the service from any device, anywhere by Internet access. The system of control and management of users and experimentation was developed in Java, for being a cross-platform language, and their implantation tested with a client-server infrastructure, composed of two servers of access to applications and remote clients with different devices.

Keywords: Security. Remote access. Internet. Software engineering.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

AD	*Active Directory*
ADCS	*Active Directory* Services Certificates
ADDS	*Active Directory* Domain Services
ADFS	*Active Directory* Federation Services
ADLDS	*Active Directory* Lighweight Directory Services
ADRMS	*Active Directory* Rights Management Services
AES	Advanced Encryption Standard
AJAX	Asynchronous JavaScript and XML
BGP	Border Gateway Protocol
CASE	Computer Aided Software Engineering
CGI	Common Gateway Interface
C3	Chrysler Comprehensive Compensation
CoBIT	Control Objectives For Information end Relatet
DASYLab	Data Acquisition System Laboratory
DORII	Deployment of Remote Instrumentation Infrastructure
FTP	File Transfer Protocol
GRIDCC	Grid enabled Remote Instrumentation with Distributed Control and Computation
GRPS	General Packet Radio Service
HTTP	Hypertext Transfer Protocol
HTML	Hypertext Markup Language
IDE	Integrated Development Environment
IP	Protocolo Internet
IS	Servidor Intermediário
ISO	International Organization for Standardization
ITIL	Information Technology Infrastructure Library
ITU-T	International Telecommunications Union -
JSON	Telecommunication
LabView	JavaScript Object Notation
MySQL	Laboratory Virtual Instrument Engineering Workbench
OSPF	Structured Query Language
RDP	Open Shortest Path First

	Remote Desktop Protocol
RINGrid	Remote Instrumentation in Next Generation Grids
SNMP	Simple Network Management Protocol
SMTP	Simple Mail Transfer Protocol
TFTP	Trivial File Transfer Protocol
TI	Tecnologia da Informação
TCP	Transmission Control Protocol
ThinVNC	Controle Remoto de Computadores
UDP	User Datagram Protocol
URL	Universal Resource Locator
UML	Unified Modeling Language
UMTS	Universal Mobile Telecommunication System
VNC	Virtual Network Computing
VPN	Virtual Private Network
VLAB	Laboratórios Virtuais
XP	*Extreme Programming*
XML	eXtensible Markup Language

SUMÁRIO

1 INTRODUÇÃO

Nos dias de hoje, o acesso a Internet está presente nas mais diversas e inovadoras tecnologias, permitindo interconexão e compartilhamentos de recursos e informações.

O acesso remoto é um exemplo destas tecnologias, oferecendo ao usuário uma interação em tempo real com laboratórios e equipamentos, tanto no controle instrumental quanto no monitoramento dos resultados. E isso vem se tornando possível, pois há várias tecnologias de redes de computadores que vêm sendo desenvolvidas e ampliadas, de forma a garantir o melhor desempenho e segurança dos serviços implementados.

Toda esta infraestrutura tecnológica, não só viabiliza diversas aplicações e serviços, como torna uma realidade o acesso remoto a equipamentos de alto custo e de propósito específico de laboratórios experimentais, permitindo também, grandes progressos no processo de ensino-aprendizagem à distância.

O **LQEE** [Laboratório de Qualidade de Energia Elétrica] da Faculdade de Engenharia de Ilha Solteira é exemplo de um ambiente que tem a necessidade de instrumentação remota segura. Atualmente, o LQEE utiliza o software ***DASYLab*** [*Data Acquisition System Laboratory*: Laboratório de Aquisição de Dados do Sistema] para realização de seus experimentos e permite que usuários acessem remotamente este sistema, elaborando-os e executando-os. No entanto, esse acesso é realizado sem qualquer mecanismo de segurança implementado, as restrições de acesso e configurações são realizadas por intervenção humana.

Assim, o presente trabalho tem por objetivo aplicar as soluções pesquisadas para o acesso remoto seguro de aplicações em tempo real, bem como, implementar um sistema para gerenciar e controlar o uso do ambiente e das ferramentas de experimentação remota do Laboratório de Qualidade de Energia Elétrica da FEIS/UNESP, utilizando mecanismos de segurança de sistemas, redes e dados.

Para o desenvolvimento deste trabalho, diversas soluções foram investigadas, as quais serão detalhadas no decorrer desse trabalho, bem como, o desenvolvimento do "***SiSLQEE***" [Sistema de Administração e Monitoramento de Experimentos do Laboratório de Qualidade de Energia Elétrica] utilizado para controlar o acesso ao sistema de experimentos DASYLab que em conjunto com outras soluções, permite o acesso remoto seguro ao servidor de experimentação.

Por meio da solução proposta será possível encontrar respostas para questões, como:

- *É possível construir uma solução para acesso remoto seguro de aplicações?*

- *É possível garantir segurança de dados e serviços com esta solução?*
- *Pode-se obter o controle de acesso, integridade e disponibilidade, que são as garantias de segurança em tecnologia da informação, com esta solução?*

Esta Dissertação de mestrado está organizada da seguinte maneira:

No segundo capítulo são apresentados os trabalhos relacionados e o referencial teórico para entendimento desse trabalho, com os conceitos e técnicas estudados para o desenvolvimento do sistema **SiSLQEE**.

No terceiro capítulo tem-se um breve histórico do LQEE e o Sistema de Experimentação remota, uma descrição do funcionamento sem administração e monitoramento virtual e automático.

Já no quarto capítulo segue o detalhamento das aplicações, dos conceitos e técnicas estudadas nos capítulos anteriores, descrevendo e ilustrando os resultados obtidos, ressaltando os serviços e recursos do Windows Server 2008 R2, para proteger as informações dos servidores onde o mesmo foi instalado e configurado.

O quinto capítulo apresenta o resultado implantado, detalhando e ilustrando o funcionamento da aplicação web SiSLQEE e a tecnologia ThinVNC, aplicada para realizar um acesso remoto seguro em experimentação em tempo real no Laboratório de Qualidade de Energia Elétrica do Departamento de Engenharia Elétrica de Ilha Solteira, Campus III – UNESP. Em seguida o sexto capítulo com as conclusões deste trabalho e por fim as referências bibliográficas.

2 TRABALHOS RELACIONADOS E REFERENCIAIS TEÓRICOS

Nessa seção serão apresentados alguns trabalhos relacionados ao tema desta dissertação, bem como, um referencial teórico, metodologias, técnicas, conceitos e recursos necessários para desenvolver uma solução viável para o LQEE. Entre estes conceitos uma abordagem simplificada é apresentada sobre engenharia de software, métodos ágeis e os aspectos de segurança de sistemas e dados, como também, sobre testes e vulnerabilidades de sistemas computacionais.

2.1 TRABALHOS RELACIONADOS

No processo de desenvolvimento dessa dissertação foram investigados alguns trabalhos sobre laboratórios virtuais, laboratórios remotos, instrumentação remota, e acesso remoto a aplicações. Estes tópicos estão relacionados e diferenciam-se dependendo de seu contexto de aplicação. É interessante enfatizar que esta investigação se faz necessária, pois proverá subsídios pontuais para solução do problema a ser abordado neste trabalho.

Alguns trabalhos aqui apresentados evidenciam os rumos da pesquisa e o estado da arte sobre a experimentação remota, mecanismos de controle de acesso e a metodologias de desenvolvimento desse tipo de sistemas.

Muitos laboratórios possuem inúmeras vezes, uma infraestrutura única de equipamentos e softwares, de alto custo, e localizada em ambientes de acesso controlado e distante de grupos de pesquisas afins. Para o acesso a esta infraestrutura, de forma cômoda e ágil, estes laboratórios permitem aos usuários desenvolverem experimentos remotamente. Para isto, é necessária a autorização do uso destes equipamentos de forma efetiva e compartilhada, de forma a explorar o potencial destes equipamentos. Neste sentido, é possível desenvolver uma infraestrutura para instrumentação remota, por meio da implementação de interfaces de acesso e gerenciamento. Essas interfaces de acesso caracterizam um laboratório virtual (via software) que permite desenvolver experimentos em equipamentos reais via Internet (OLIVEIRA et al., 2010).

Além disso, por serem equipamentos de alto custo e de experimentação, há a necessidade de um controle e gerenciamento efetivo sobre os equipamentos, devendo usuários devem ser cadastrados e autorizados, bem como, seus experimentos controlados.

Nesta linha, há diversos grupos de pesquisa, que tem desenvolvido ferramentas computacionais para a implementação destes laboratórios virtuais remotos, para diversas aplicações via Internet, conhecidos com WebLabs.

Os WebLabs são laboratórios virtuais implementados para operação e controle de equipamentos e sistemas, bem como, desenvolvimento de experimentos reais, monitorados e gerenciados à distância pela da Internet (BORGES, 2002). Os Weblabs foram desenvolvidos para viabilizar o compartilhamento de equipamentos e ideias entre pesquisadores de diferentes laboratórios, abrindo possibilidades inovadoras nas pesquisas colaborativas. Os usuários acessam uma URL e, clicando em botões virtuais da tela, controlam os instrumentos do laboratório, visualizam resultados em imagens em tempo real. O principal objetivo do WebLab é oferecer ao usuário uma interação em tempo real com o equipamento, tanto no controle do instrumento quanto no monitoramento dos resultados (BORGES, 2002; BOTTENTUIT, COUTINHO; 2007; CIUBOTARIU; OLIVEIRA, 2010; RICHTER; WATSON; KASSAVETIS; KRAFT; GRUBE; BOEHRINGER; VRIES; HATZIKRANIOTIS; LOGOTHETIDIS, 2012).

Segundo Marín, Sanz, Nebot e Wirz (2005) apresentam uma arquitetura de laboratório remoto baseado na Internet, a qual permite a pesquisadores e estudantes, controlar e programar remotamente dois robôs em tempo real. Esta solução é utilizada para ensino à distância e demonstra a utilização de programação remota em conjunto com uma interface multimídia, proporcionando um controle remoto flexível e produtivo, chamado Tele Laboratório. O acesso remoto é realizado por meio de uma Applet Java com controle de usuários na interação com os robôs, realizando experimentos por meio de algoritmos em diversas linguagens de programação. Os resultados dos experimentos podem ser visualizados através da interface do Tele L or tório "*client-side*" em Java multimídia 3D.

De forma bastante inovadora o MIT (*Massachussets Institute*) disponibiliza acesso remoto a laboratórios experimentais para alunos e pesquisadores, permitindo a eles executar experiências em hardware através da Internet. O sistema de acesso é composto de um servidor ligado a uma máquina de destino em tempo real xPC que se comunica com o módulo de controle do motor DC via interfaces de aquisição de dados. O software desenvolvido tem como objetivo substituir o uso de um servidor de vídeo para um modelo virtual, com isso melhorar a flexibilidade e reduzir o custo de implementação. O software tem um banco de dados que permite múltiplos acessos, para realizar e assistir aos experimentos, sob supervisão remota de um servidor de administração (LEE; TRUONG; RHODES; MCLAREN; WANG, 2010).

Hrusha et al. (2007) desenvolveu um sistema de medição com acesso via Web, o qual explora canais de comunicação existentes. Esta solução permite acesso remoto a recursos experimentais para coleta de informações, bem com, atualização dinâmica das informações das medições (controle e automação). O usuário poderá se conectar a servidores remotos, que estão em redes locais protegidas por *proxies* e *firewalls*. Este sistema é divido em duas partes, um Servidor Intermediário (IS) e um Servidor de Medição (MS). O Servidor Intermediário inclui páginas Web, CGI e Applets Java, que são carregadas juntamente com as páginas HTML, proporcionando aos usuários a possibilidade de monitorar e controlar o sistema usando *browsers*, transferências de dados através de HTTP-consulta e PHP para a escrita em CGI e o Servidor de Medição (MS) é implementado pelo software LabView, usando a rotina CGI e processo HTTP/Applet Java (HRUSHA; KOCHAN; KURYLYAK; OSOLINSKIY, 2007).

Freire (2007) apresenta um trabalho que foi realizado no curso de Engenharia Eletrotécnica e de Computadores do Instituto Superior Técnico – Universidade Técnica de Lisboa e consiste de uma aplicação de tecnologias de transmissão de dados GPRS, UMTS (3G) e VNC, para uso e controle remoto de dispositivos. Do lado cliente, o terminal móvel implementado em Java J2ME estabelece comunicação (via 3G ou outra tecnologia) com outro computador pessoal. Esta solução utiliza para o acesso remoto a ferramenta VNC proporcionando melhor atendimento as necessidades exigidas, dando possibilidade de alteração nos código, tanto do lado do cliente como do lado do servidor. Em conjunto, as duas tecnologias, a Java e o VNC, desenvolvem um sistema que permite essa comunicação de forma confiável (FREIRE; NUNES, 2007).

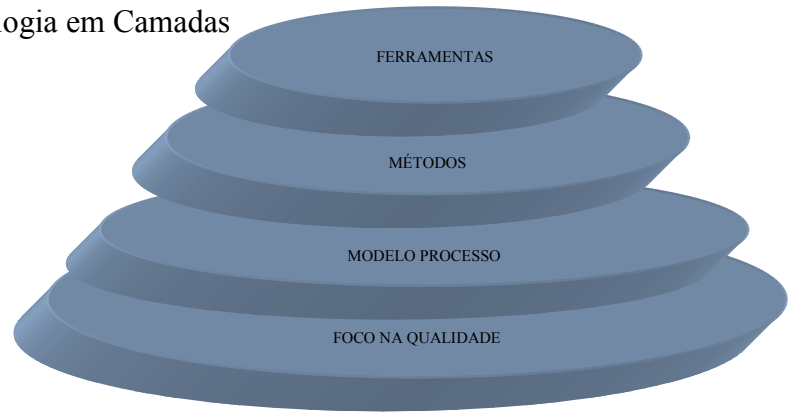
Soares (2004) apresenta um estudo sobre Metodologias Ágeis *Extreme Programming* e *Scrum* para o Desenvolvimento de Software, destacando que assa metodologias têm sido bem aceitas pela indústria de software e por pesquisadores da Engenharia de Software. A *Extreme Programming* (XP) e *Scrum* podem ser aplicadas em conjunto. A XP usada para a fase de desenvolvimento e a *Scrum* para o planejamento e gerenciamento do projeto. A Integração das duas metodologias seria relativamente simples, uma vez que elas compartilham algumas características, como a necessidade da presença do cliente, pequenas liberações e o incentivo em fazer mudanças necessárias para atender requisitos reais dos interessados no projeto. Nesse estudo foi identifica que existe uma proposta da metodologia híbrida XP*@Scrum*. Apesar de não existirem estudos empíricos suficientes, alguns autores recomendam o uso em conjunto da *Scrum* e da XP para grandes projetos de software.

Conclui-se que os trabalhos relacionados nesse capítulo contribuíram para, a solução proposta nessa dissertação. Considerando-os estudos realizados em busca de solução viável para o acesso remoto seguro, encontrou-se nesses trabalhos o caminho para a solução viável e segura para os usuários do LQEE.

2.2 ENGENHARIA DE SOFTWARE E MÉTODOS ÁGEIS

Segundo Pressman (2011), a engenharia de software é uma tecnologia em camadas, qualquer abordagem de engenharia deve estar fundamentada em um comprometimento organizacional com a qualidade. A gestão da qualidade total seis Sigma e filosofias similares promovem uma cultura de aperfeiçoamento contínua de processos, e é esta cultura que, no final das contas, leva ao desenvolvimento de abordagens cada vez mais efetivas na engenharia de software. Na Figura 1 ilustra a tecnologia em camadas.

Figura 1 - Tecnologia em Camadas



Fonte: Pressman (2011).

A tecnologia em camadas é descrita nos itens abaixo. Inicia-se, na primeira camada:

O Foco na qualidade: a qualidade é almejada em qualquer parte da engenharia. É mais focada na fase do processo de desenvolvimento, permitindo ao gerente de projetos um controle ao desenvolvimento que é uma referência. A camada mais importante dessa tecnologia que sustenta a engenharia de software (PRESSMAN, 2011).

Modelo Processo: é a base da engenharia de software, constitui em elos entre os métodos e ferramentas, a sequência em que os métodos serão aplicados, o conjunto de tarefas que apoiam o modelo do processo garantindo a qualidade, o gerenciamento de configuração, e a produção de documentos (PRESSMAN, 2011).

Métodos: proporcion m os det lhe s de “como f er” p r construir o softw re e envolvem um amplo conjunto de tarefas (PRESSMAN, 2011).

Ferramentas: fornecem suporte automatizado aos métodos. E atualmente existem ferramentas para sustentar cada um dos métodos. Quando as ferramentas são integradas é estabelecido um sistema de suporte ao desenvolvimento de software chamado ***CASE*** [*Computer Aided Software Engineering*: Engenharia de Software Auxiliada por Computador] (PRESSMAN, 2011).

Um processo de software é um conjunto de atividades e resultados associados que auxiliam na produção de software. Entre as várias atividades associadas, existem, por exemplo, a análise de requisitos e a codificação. Os resultados do processo é um produto que reflete a forma como o processo foi conduzido (SOARES, 2004).

Segundo Sommerville (2012) existem vários processo para o desenvolvimento de software, com atividades comuns fundamentais a todos eles:

Especificação de Software: definição das funcionalidades (requisitos) e das restrições do software. Geralmente é uma fase em que o desenvolvimento conversa com cliente para definir as características do novo software (SOMMERVILLE, 2012).

Os requisitos são definidos por meio de reuniões entre os interessados, no software, para descobrir qual é a real necessidade do cliente, em que são identificados os primeiros requisitos.

Segundo Sommerville (2012), o requisito pode variar de uma declaração abstrata de alto nível de um serviço ou de uma restrição do sistema para uma especificação matemática funcional.

Os requisitos podem ser dos seguintes tipos:

Requisitos de usuário – são declarações em linguagem natural com diagramas de serviços que o sistema deverá fornecer e suas restrições operacionais. Escrito para os clientes (SOMMERVILLE, 2012).

Requisitos de sistema – é um documento estruturado que estabelece descrições detalhadas das funções do sistema, serviços e restrições operacionais. Define o que deve ser implementado assim, pode ser parte de um contrato entre o cliente e o desenvolvedor (SOMMERVILLE, 2012).

E os requisitos se classificam como funcionais e não funcionais:

Funcionais – quando o sistema deve fornecer declarações de serviços, como o sistema deve reagir a entradas específicas e como o sistema deve se comportar em determinadas situações. Pode explicitar o que o sistema não deve fazer.

E **não funcionais** – tem restrições aos serviços ou funções oferecidas pelo sistema, tais como restrições de tempo, no processo de desenvolvimento, padrões e muitas vezes se aplica

ao sistema com um todo ao invés de características individuais aos serviços (SOMMERVILLE, 2012).

De acordo com Sommerville (2012), os requisitos não funcionais podem afetar a arquitetura geral de um sistema, em vez de componentes individuais, e também pode gerar requisitos que restringem os requisitos existentes:

Requisitos do produto (velocidade de execução, confiabilidade);

Requisitos organizacionais (padrões de processo usados, requisitos de implementação);

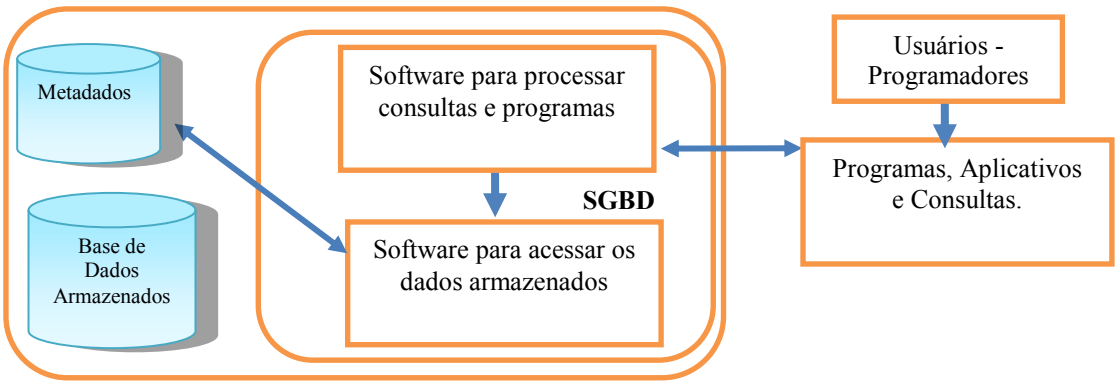
Requisitos externos (fatores externos ao sistema e os processo de desenvolvimento como requisitos regulares e legais).

Projeto e Implementação do Software: o software é produzido de acordo com as especificações. Nesta fase são proposto modelos por meio de diagramas, e estes modelo são implementados em alguma linguagem de programação (SOMMERVILLE, 2012).

Para implementar o software é necessário o a construção do projeto de banco de dados (BD), em que inicialmente é criado um BD com informações básicas, com auxílio de técnicas de programação em conjunto com um SGBD [Sistema Gerenciador de Banco de Dados], que ficará responsável para armazenar informações que alimentaram o sistema.

SGBD [O Sistema Gerenciador de Banco de Dados] é o software que permite criar, manter e manipular banco de dados para diversas aplicações. A criação e manutenção de bancos de dados é tarefa de uma pessoa ou grupo de pessoas, normalmente referenciada como o administrado de banco de dados. Esse sistema é constituído por um banco de dados e por um sistema gerenciador de bando de dados (PRESSMAN, 2011). O funcionamento do SGBD é ilustrado na Figura 2.

Figura 2 - Sistema simplificado de Banco de dados.



Fonte: Elaboração do próprio autor.

Testes são fases seguintes do processo de desenvolvimento de um software. Na fase de implementação é feita a tradução dos requisitos em forma legível de máquina, que é um conjunto de código, a programação específica que existe para cada tipo de software, e para fazer o que o usuário precise que funciona é necessário passar as informações em forma de códigos. Então será utilizada a linguagem de programação (JAVA, C++, C#, Visual Basic, Visual Studio, Ruby, Delphi, PHP entre outras). Assim por meio do levantamento de requisitos em um programa específico é construído o código como uma dessas linguagens citadas acima, e por fim é traduzido ao cliente de uma forma de um software, atendendo suas necessidades específicas (PRESSMAN, 2011).

Os testes, podem ser realizados paralelamente com a implementação, onde o processo de realização de testes concentra-se nos aspectos lógicos internos do software, garantindo que todas as instruções tenham sido testadas, e consiste também nos aspectos funcionais externos, ou seja, realizando testes para descobrir erros e garantir que a entrada definida produza resultados reais que concordem com os resultados exigidos (PRESSMAN, 2011).

Validação de Software: o software é valido para garantir que todas as funcionalidades especificadas foram implementadas (SOMMERVILLE, 2012).

Evolução de software: o software precisa e evoluir para continuar sendo útil ao cliente (SOMMERVILLE, 2012).

Existem vários processos de software definidos na literatura da Engenharia de Software. É comum mesmo algumas organizações criarem seu próprio processo ou adaptar algum processo à sua realidade (SOARES, 2004).

Para iniciar um processo de software é necessário definir o modelo de ciclo de vida do software, são definidas as etapas a seguir para o desenvolvimento de um software, na engenharia de software têm-se as metodologias tradicionais, em que são orientadas a documentação e metodologias ágeis, que procuram desenvolver software com o mínimo de documentação (SOARES, 2004).

Segue no subitem 2.2.1 um breve conceito das metodologias tradicionais, tais como cascata, incremental, prototipagem e espiral e no subitem 2.2.2 o surgimento das metodologias ágeis, e suas principais metodologias utilizados *Extreme Programming* e *Scrum*.

2.2.1 Metodologias Tradicionais para desenvolvimento de software

As metodologias são também chamadas de orientadas à documentação. Segundo Sommerville (2012), é uma metodologia baseado em estágios de desenvolvimento separados, como os produtos a serem produzidos em cada um desses estágios planejados

antecipadamente, propondo uma abordagem sistemática para o desenvolvimento de software, tais como cascata, incremental, espiral, prototipagem, entre outros (SOMMERVILE, 2012).

Modelo Cascata: sugere uma abordagem sequencial e sistemática para o desenvolvimento de software, começando como levantamento de necessidades por parte do cliente, avançando pelas fases de planejamento, modelagem, construção, emprego e culminando no suporte contínuo do software concluído (PRESSMAN, 2011).

Modelo Prototipagem: frequentemente, o cliente define uma série de objetivos gerais para software, mas não identifica, detalhadamente, os requisitos para funções e recursos. Em outros casos, o desenvolvedor encontra-se inseguro quanto à eficiência de um algoritmo, quanto à adaptabilidade de um sistema operacional ou quanto à forma em que deva ocorrer a interação homem/máquina. Em situações como essas, e em muitas outras, o paradigma de prototipação pode ser a melhor escolha de abordagem (PRESSMAN, 2011).

Modelo Espiral: usando-se esse modelo o software será desenvolvido em uma série de versões evolucionárias. Nas primeiras iterações, a versão pode consistir em um modelo ou em um protótipo. Já nas "iterações" posteriores, são produzidas versões cada vez mais completas do sistema que passa pelo processo de engenharia (PRESSMAN, 2011).

Modelo Incremental: combina elementos dos fluxos de processos lineares e paralelos, aplica sequências lineares, de forma escalonada, à medida que o tempo vai v nç do. C d sequênci line r ger "increment is" (entregáveis/ prov dos/li e r dos) do software (PRESSMAN, 2011).

2.2.2 Metodologias Ágeis para desenvolvimento de software

As novas metodologias e técnicas que buscam garantir agilidade e ao mesmo tempo um processo de desenvolvimento de software robusto e veloz, com a necessidade de seguir o processo de desenvolvimento rápido, focando no retorno ao usuário final e pouco documentação, cresceu a necessidade de novas metodologias para desenvolver software.

A palavra ágil foi relacionada pela primeira vez à Engenharia de Software em 2001. Um grupo de dezessete representantes de metodologias de desenvolvimento de software consider d s "leves", que estavam inovando no mundo da construção de sistemas, se reuniu por dois dias em uma estação de esqui em Snowbird, Utah para levantar pontos em comum entre suas metodologias. A partir dessa reunião, foi criado o Manifesto Ágil, em que são definidos os valores Ágeis (FOWLER, 2000; HISGHSMITH, 2002; TELES, 2004; BECK, 2000).

E assim surgiu a Programação ágil, Desenvolvimento ágil, Manifesto ágil ou Métodos ágeis. São diversos títulos, mas com o mesmo objetivo que é promover práticas ágeis de desenvolvimento de software e no gerenciamento de projetos.

A programação ágil vem sendo cada vez mais valorizada, pois há uma grande quantidade de engenheiros de software e gerentes de projetos que começaram a adaptar tais conceitos para o seu próprio dia a dia. Com isso foram surgindo novas metodologias e diferentes técnicas direcionadas a dinamizar o processo de desenvolvimento de software. Com a evolução no mercado mundial do software, novas ideias e estratégias de marketing vêm surgindo, e a comunidade ágil vem se revelando bastante eficaz, com novas maneiras de resolver problemas antigos na área de gerência de projeto e da programação (HIGHSMITH, 2001).

A metodologia de programação ágil sugere novidades, abordando um conjunto de novas ideias e modificações nas velhas, enfatizando uma maior colaboração entre programadores, especialistas do domínio e comunicações presenciais. Os métodos ágeis normalmente produzem menos documentação do que os outros métodos, pois ela prioriza os resultados do software funcionado (HIGHSMITH, 2001).

Segundo Beck (2000), o manifesto ágil é definido por quatro valores em que é importante entender que, ao mesmo tempo em que valorizar os conceitos do lado direito, deve valorizar mais ainda os valores do lado esquerdo.

Estamos descobrindo maneiras melhores de desenvolver software fazendo-o nós mesmos e ajudando outros a fazê-los. Por meio deste trabalho, passamos a valorizar (BECK, 2000):

- **Indivíduos e interações entre eles** mais que processo e ferramentas;
- **Software em funcionamento mais** que documentação abrangente;
- **Colaboração com o cliente mais** que negociação de contratos;
- **Responder a mudanças mais** que seguir um plano;

O "Manifesto Ágil" não rejeita os processos e ferramentas, documentação, negociação de contratos ou o planejamento, mas simplesmente mostra que eles têm importância secundária quando comparado com os indivíduos e interações, com o software estar executável, com a colaboração do cliente e as respostas rápidas a mudanças e alterações. Esses conceitos aproximam-se de forma que as Tecnologias da Informação trabalham e respondem a mudança (TEIXEIRA, 2012; BECK, 2000; SOARES, 2004). As metodologias ágeis mais conhecidas são XP [*Extreme Programming:* Programação Extrema] e *Scrum*. Estas

metodologias vêm sendo utilizadas em complementação aos processos de sistemas já desenvolvidos, bem como, uma complementando a outra.

2.2.2.1 Extreme Programming

Os métodos utilizados para o desenvolvimento de software vêm ganhando importância e gerando interesse tanto na comunidade de desenvolvedores quanto na área acadêmica. Nos métodos tradicionais de desenvolvimento de software o foco é voltado para a documentação, em que são necessários requerimentos completos e fixos, o que torna essa metodologia pesada e não flexível. Então diante disso surgiu a XP em que se visa um rápido desenvolvimento, atendendo as reais necessidades do cliente e permite modificações à medida que novas necessidades vão aparecendo.

Em XP, a codificação é a principal tarefa, baseia-se em uma revisão permanente do código, testes, entregas frequentes, participação do usuário final, reconstrução contínua, refinamento contínuo da arquitetura, integração contínua, planejamento, projeto e recomeçar o mesmo a qualquer hora. Esse fator de desenvolvimento ágil que a XP oferece, tem um maior interesse para a indústria e academia, pois proporciona vários benefícios no processo de produção (BECH, 1999; JEFFRIES, ANDERSON 2001).

Esta metodologia foi criada em 1996, por Kent Bech, no Departamento de Computação da montadora de carros *Daimler Crysler* e possui diferenças em relação a outros modelos, podendo ser aplicado a projetos de alto risco e com requisitos dinâmicos. Essa metodologia vem fazendo sucesso em diversos países, por ajudar a criar sistemas de melhor qualidade, que são produzidos em menos tempo e de forma mais econômica que o habitual. As principais vantagens do uso dos métodos ágeis são: a transparência durante todo o processo de desenvolvimento de uma aplicação; a simplicidade no acompanhamento da aplicação da metodologia; um maior compromisso da equipe envolvida, o uso de ferramentas simples e à vista e a maior produtividade ou aprendizado. Esse método estabelece conjunto de ações que impactam de forma direta na obtenção de melhores produtos/software, além de modificar, potencializar e simplificar o dia a dia da equipe de desenvolvimento e sua relação com o gerente e o com o cliente (TEXEIRA, 2012; BECK, 2004).

Segundo Highsmith (2000), os primeiros projetos a usar XP foi o sistema de Recursos Humanos C3 [*Chrysler Comprehensive Compensation*: Compensaçãoo Compreensiva Chrysler], *Daimler Crysler*. Após anos de fracasso utilizando metodologias tradicionais, com o uso da XP o projeto ficou pronto em pouco mais de um ano. Há, também,

outras empresas que usaram a metodologia XP, casos bem sucedidos como Brasil Telecom, SoftSite Tecnologia, Embrapa Informática, Motorola e Siemens AG.

A maioria das regras da metodologia XP causa polêmica a primeira vista e muitas não fazem sentido se aplicadas isoladamente. De acordo com Beck (2000), a XP enfatiza o desenvolvimento rápido do projeto e visa garantir a satisfação do cliente, além de favorecer o cumprimento das estimativas.

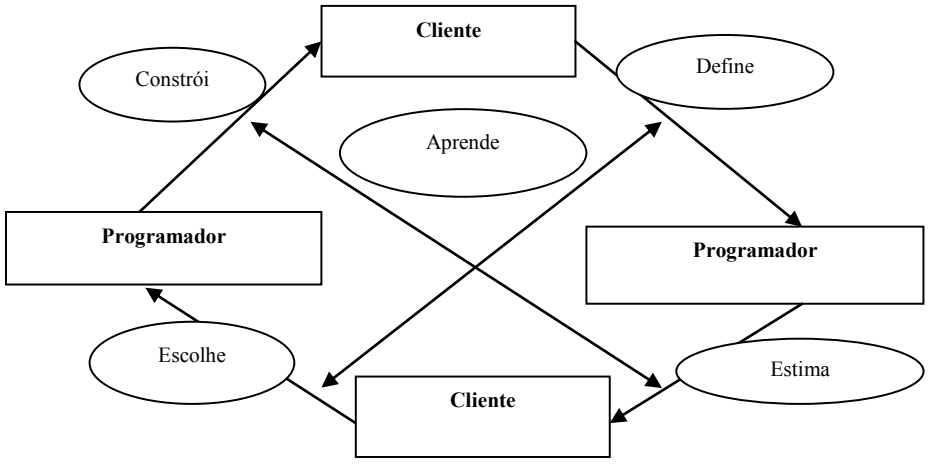
A proposta da XP sugere o que é melhor fazer algo simples hoje e pagar um pouco mais amanhã para fazer modificações necessárias do que implementar algo complicado hoje que talvez não venha a ser usado, sempre considerando que requisitos são mutáveis.

O cliente então constantemente sugere novas características e informações aos desenvolvedores. Eventuais erros e não conformidades são rapidamente identificados e corrigidos nas próximas versões. Desta forma, a tendência é que o produto final esteja de acordo com as expectativas reais do cliente (JEFFRIES; ANDERSON, 2001; TELES, 2005).

De acordo com, Jeffries e Anderson (2001), Teles (2005), os desenvolvedores de um projeto XP procuram programar as funcionalidades priorizadas para cada etapa com a maior qualidade possível, porém com foco apenas naquilo que é visivelmente essencial. Generalizações que não se provem prontamente indispensáveis são evitadas, pois mantendo o sistema suficiente e simples o tempo todo, qualquer coisa que inserida será aceita facilmente e na menor quantidade de lugares possível.

Uma preocupação permanente dos desenvolvedores é não ter tempo suficiente para realizar um trabalho de qualidade e a XP trata essas questões dividindo claramente a responsabilidade por decisões técnicas e de negócios (BECK; FOWLER, 2001; TELES, 2006). Na Figura 3 pode-se visualizar o ciclo de vida XP.

Figura 3 - Ciclo de Vida XP



Fonte: Elaboração do próprio autor.

A Figura 3 representa claramente que no funcionamento da XP tudo se inicia quando o cliente tem um problema a ser resolvido e precisa de um software, ele detalha isso para o analista/desenvolvedor, que por sua vez projeta e desenvolve o software, de acordo com as especificações e definições do cliente, tudo é estimado, avaliando e com isso os dois são beneficiados com os resultados.

A XP baseia-se em nas seguintes Práticas:

Planejamento: Consiste em decidir as prioridades do projeto. A XP baseia-se em requisitos atuais para desenvolvimento de software, não em requisitos futuros. Além disso, a XP procura evitar os problemas de relacionamento entre a área de negócios e a área de desenvolvimento. As duas áreas devem cooperar para o sucesso do projeto, e cada uma deve focar em partes específicas do projeto. Desta forma, enquanto a área de negócios deve decidir sobre o escopo, a composição das versões e as datas de entrega, os analista e programadores devem decidir sobre as estimativas de prazo, o processo de desenvolvimento e o cronograma detalhado para que o software seja entregue nas datas especificadas (BECK, 2000; SOARES, 2004; SOUZA, 2007).

Projetos XP procuram dividir o tempo disponível para o projeto utilizando dois conceitos: release e iterações (pequenas versões do sistema). O XP tem releases que assumem poucos meses, que se dividem em iterações de duas semanas, que por sua vez se decompõem em tarefas que adotam alguns dias (BECK; FOWLER, 2001; TELES, 2005).

Em cada ciclo de *release*, o cliente controla o escopo, decidindo o que fazer e o que adiar, de modo a prover o melhor release possível na data acertada. O trabalho se encaixa no cronograma baseado no valor para o negócio, dificuldade a velocidade de implementação da equipe (JEFFRIES; ANDERSON, 2001; TELES, 2005).

Entregas frequentes: Visa à construção de um software simples, e conforme os requisitos surgem, há a atualização do software. Cada versão entregue deve ter o menor tamanho possível, contendo os requisitos de maior valor para o negócio. Idealmente devem ser entregues versões a cada mês, ou no máximo a cada dois meses, aumentando a possibilidade de *feedback* rápido do cliente. Isto evita surpresas caso o software seja entregue após muito tempo, melhora as avaliações e o *feedback* do cliente, aumentando a probabilidade do software final estar de acordo com os requisitos do cliente (BECK, 2000; TELES 2005).

Metáfora: São as descrições de um software sem a utilização de termos técnicos, com a finalidade de guiar o desenvolvimento do software. (BECK, 2000; SOARES, 2004).

Projeto simples: O programa desenvolvido pelo método XP deve ser o mais simples possível e satisfazer os requisitos atuais, sem a preocupação de adicionar funcionalidades futuras. Eventuais condições futuros devem ser adicionadas assim que eles realmente existirem (BECK, 2000; SOARES, 2004).

Teste de Aceitação: A XP focaliza a validação do projeto durante todo o processo de desenvolvimento. Os programadores desenvolvem o software criando primeiramente os testes (BECK, 2000; SOARES, 2004).

Programação em pares: A programação é feita em dupla, ou seja, dois programadores trabalham em um único computador. O desenvolvedor que está com o controle do teclado e do *mouse* implementa o código, enquanto o outro observa continuamente o trabalho que está sendo feito, procurando identificar erros sintáticos e semânticos e pensando estrategicamente em como melhorar o código que está sendo implementando. Esses papéis podem e devem ser alterados continuamente. Uma grande vantagem da programação em dupla é a possibilidade dos desenvolvedores estarem sempre aprendendo um com o outro. (BECK, 2000; SOARES, 2004).

Embora a programação em par ofereça oportunidades de melhoria nos projetos, a reação intuitiva de muitas pessoas é rejeitar a ideia, porque assumem que haverá um aumento de cem por cento de programador-hora colocando dois programadores para fazer o trabalho que um pode fazer (WILLIAMS; KESSLER, 2003).

A programação **em par** também eleva a força da equipe, permitindo que ela supere melhor a perda ou a adição de um novo membro. Com a programação em par, o risco de projeto associado à perda de um programador chave é reduzido, pois existe outro membro que entende do que está sendo desenvolvido (WILLIAMS; KESSLER, 2003).

Fatoração: Enfoca o aperfeiçoamento do projeto do software e está presente em todo o desenvolvimento. A reconstrução deve ser feita apenas quando é necessário, ou seja, quando um dos desenvolvedores da dupla (quando for aplicando a técnica de programação em par), ou os dois, percebem que é possível simplificar o módulo atual sem perder nenhuma funcionalidade (BECK, 2000; SOARES; 2004).

Propriedade coletiva: O código do projeto pertence a todos os membros da equipe. Isto significa que qualquer pessoa pode adicionar valor a um código, mesmo que ele próprio não o tenha escrito o mesmo, pode fazê-lo, desde que faça a bateria de testes necessária (BECK, 2000; TELES, 2005; TEIXEIRA, 2012).

Isto é possível porque na XP todos são responsáveis pelo software inteiro. Uma grande vantagem desta prática é que, um membro da equipe continua o projeto com poucas

dificuldades, pois todos conhecem todas as partes do software, mesmo que não seja de forma detalhada (BECK, 2004; SOARES, 2004).

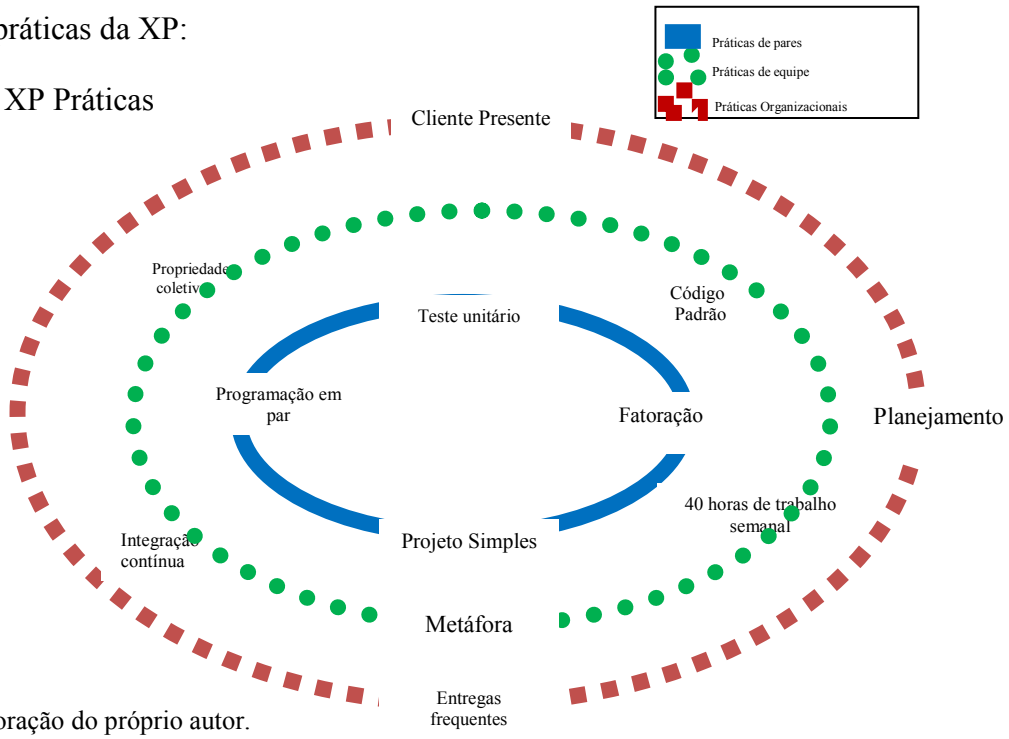
Integração contínua: Interagir e construir o software várias vezes por dia, mantendo os programadores em sintonia, além de possibilitar processos rápidos. Integrar apenas um conjunto de modificações de cada vez é uma prática que funciona bem porque fica óbvio quem deve fazer as correções quando os testes falham: a última equipe que integrou código novo ao software. Esta prática é facilitada com o uso de apenas uma máquina de integração (BECK, 2000; SOARES, 2004).

40 horas de trabalho semanal: A XP assume que fazer horas extras não deve ser feitas constantemente. Caso seja necessário trabalhar mais de 40 horas pela segunda semana consecutiva, existe um problema sério no projeto que deve ser resolvido não com aumento de horas trabalhadas, mas com melhor planejamento. Esta prática procura ratificar o foco nas pessoas e não em processos e planejamentos (BECK, 2000; SOARES, 2004).

Cliente presente: É fundamental a participação do cliente/usuário final durante todo o desenvolvimento do projeto. O cliente/usuário final deve estar sempre disponível para sanar todas as dúvidas de requisitos, evitando atrasos e até mesmo construções erradas (BECK, 2000; SOARES, 2004).

Código padrão: Padronização na arquitetura do código, para que este possa ser compartilhado entre todos os programadores (BECK, 2000; SOARES, 2004). A Figura 4 ilustra as práticas da XP:

Figura 4 - XP Práticas



Fonte: Elaboração do próprio autor.

A metodologia XP não deve ser usada quando os valores da organização não estiverem alinhados, com sistemas de premiação individual e em contratos de escopo fechando porque dificultará as mudanças e a utilização otimizada do mesmo. Tem uma documentação sucinta, objetiva e rápida. Foca-se na finalização do software e em solucionar a necessidade do cliente, em ter uma informatização comum de um sistema na sua empresa (BECK, 2000; TEIXEIRA 2012).

Assim, todas as práticas abordam e reúne características que fazem com que a XP seja um método bem definido, que permite o desenvolvimento ágil, sendo capaz de reagir rapidamente a mudanças e ao mesmo tempo seguro, com diversas técnicas intrínsecas de qualidade (COCKBURN, 1999; SANTOS, 2007).

2.2.2.2 Scrum

O termo *Scrum* foi usado pela primeira vez no contexto de processo de desenvolvimento por Ikujiro Non k e Hirot k T keuchi em um rtigo ch m do "O jogo do desenvolvimento de novos produtos" pu li c do em H rv rd Business Review em 1986 (PAULA, 2010).

A metodologia *Scrum* tem a finalidade de gerenciar e controlar a produção de software empregando processos leves, iterativos e incrementais. São formados por equipes pequenas e com requisitos menos estáticos. Essas equipes são formadas por projetistas, programadores, engenheiros e gerentes de projeto. Focando a qualidade, trabalhando em cima de funcionalidades definidas no início de cada etapa do projeto, que são as iterações chamadas de *Sprint* [Ciclo] (SCHWABER, 2001; PAULA, 2010).

Segundo Schwaber (2001), o *Scrum* baseia-se em 5 (cinco) principais características:

Flexibilidade dos resultados: A decisão sobre o que será entregue e quando é tomada pelo cliente, de modo que os interesses do negócio sejam respeitados. As estimativas dos desenvolvedores auxiliam os clientes/usuários nessa definição (DANTAS, 2003).

Equipes pequenas: Uma equipe de seis ou dez pessoas, mas é possível ter mais de uma equipe de desenvolvimento (DANTAS, 2003; SCHWABER, 2001; RISING, 2000).

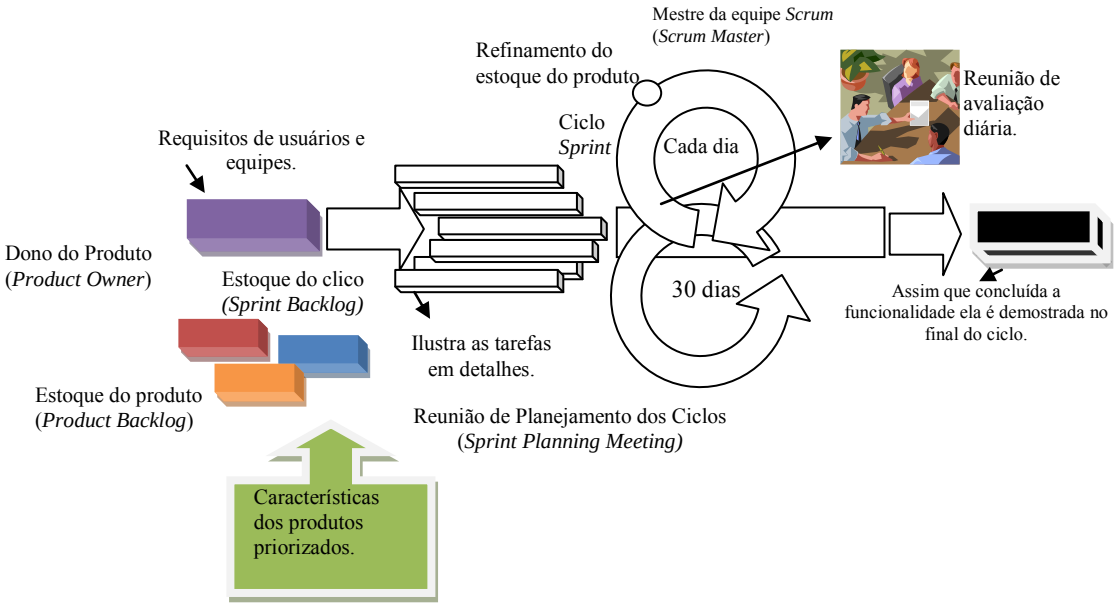
Revisões frequentes: O progresso, a complexidade e os riscos do projeto são constantemente acompanhados e medidos. Ao final de cada ciclo de desenvolvimento, uma versão executável do software é produzida para avaliação pelo cliente/usuário (DANTAS, 2003; SCHWABER, 2001).

Colaboração: A troca de informações e de resultados é constante entre equipes (DANTAS, 2003).

Orientação a objetos: Cada equipe assume uma responsabilidade por um conjunto dos objetos relacionados (DANTAS, 2003).

Scrum é dividido em iterações/etapas, cada uma delas é chamada de *Sprint* [Ciclo]. O *Sprint* representa um ciclo. Ele pode durar de duas a quatro semanas, mas é tipicamente mensal, pois 30 dias fornecem uma melhor visibilidade dos objetivos do projeto e comprometimento da equipe. No final de cada ciclo deve-se ter uma parte do produto concluída que possa ser apresentada ao cliente. Estas entregas parciais vão sendo desenvolvidas até o produto estar concluído, e à medida que forem surgindo novas funcionalidades desejadas, novos ciclos vão sendo realizados, visando à melhoria do ambiente de desenvolvimento (PAULA, 2010; MATTIOLI; LAMOUNIER; CARDOSO; ALVES, 2009; SUBRAMANIAM; HUNT, 2006). A figura 5 foi adaptada com adição das traduções das práticas e papéis, redesenhada, para uma melhor resolução substituída as formas e figuras originais para demostrar o processo completo do *Scrum*:

Figura 5 - Processo completo do *SCRUM*



Fonte: Hunt (2006).

A seguir serão apresentados detalhes da Figura 7, o processo completo *Scrum*.

Um projeto *Scrum* começa mesmo que ainda se tenha uma visão superficial no princípio, talvez expressa em termos de marketing ou uma visão técnica, mas que se esclarece melhor à medida que o projeto evolui. A partir dessa visão inicial se elabora uma lista enxuta dos principais itens, de tudo o que se deseja para o software, contendo funcionalidades e requisitos que precisarão ser desenvolvidos até a finalização do projeto. Esta lista de itens é

conhecida como Estoque do Produto [*Product Backlog*]. Cada item tem uma prioridade de entrega que indica o quanto de valor ele gera para o negócio. Esta lista não precisa estar completa logo no começo, ela pode ganhar outros itens no decorrer do projeto (PAULA, 2010; HUNT, 2006; BAPTISTA, 2013).

No início de cada ciclo a equipe seleciona os itens do Estoque do Produto [*Product Backlog*], de acordo com as suas prioridades e o tempo que será gasto para completar as suas funcionalidades. Esta nova lista é conhecida como Estoque do Ciclo [*Sprint Backlog*]. É importante que a equipe identifique os itens do Estoque do Ciclo [*Sprint Backlog*], porque ela estará comprometida a finalizar tais tarefas. Os seus membros são quem deverão escolher com o que irão se comprometer. Nesse momento as tarefas maiores são subdivididas em partes menores (PAULA, 2010; CARVALHO, MELLO, 2012; BAPTISTA, 2013). Após o Ciclo de Estoque [*Sprint Backlog*] estar concluído, o total de horas trabalhadas é comparado com o total previsto anteriormente no Estoque do Produto [*Product Backlog*] (PAULA, 2010; DEEMER; BENEFIELD; LARMAN; VODDE, 2011; PAULA, 2010; STRAUHS, 2013; BAPTISTA, 2013).

Os aspectos com maior visibilidade do projeto são priorizando no *Scrum*, e tem uma melhor inserção no mercado e uma maior e melhor colaboração entre os integrantes do grupo de trabalho. A Tabela 1 contém os papéis do *Scrum.*

Tabela 1 – Papéis e suas descrições *Scrum*

Descrição dos papéis delegados aos funcionários de cada projeto implementado com *Scrum*	
Papéis	**Descrição**
Dono do Produto [*Product Owner*]	Representante do cliente no projeto. Suas responsabilidades são definir funcionalidades de acordo com o valor de mercado, planejar e fazer a lista de prioridades.
Mestre da equipe *Scrum* [*Scrum Master*]	É o intermediador entre os interesses da equipe de desenvolvimento e dos clientes/usuários. Também responsável por manter o funcionamento com produtividade da equipe.
Equipe [*Team*]	É a equipe responsável pelo desenvolvimento do projeto. Ele é multidisciplinar e é composto por um grupo de cinco a nove integrantes.
Cliente [*Client*]	Participa das tarefas relacionadas à implementação da lista de funcionalidades.

Fonte: Deemer et al. (2010).

A Tabela 2 contém as práticas e as descrições do *Scrum.*

Tabela 2 – Práticas e as descrições *Scrum.*

Descrição das práticas executadas durante a implementação da metodologia *Scrum*	
Práticas	**Descrição**
Reunião de Planejamento do ciclo *Scrum* [*Scrum Sprint Planning Meeting*]	Reunião em que o Dono do Produto planeja e faz a lista de prioridades que deverão ser cumpridas no projeto por completo.
Reunião Diária do *Scrum* [*Daily Scrum Meeting*]	Reunião diária em que cada membro da equipe responde o que já fez o que pretende fazer e se há algum impedimento para a conclusão de sua tarefa.
Criação do Incremento do Produto [*Product Increment*]	A finalização das funcionalidades definidas para um determinado ciclo.
Retrospectiva do ciclo [*Sprint Retrospective*]	Reunião de retrospectiva do ciclo em que são avaliados aspectos como o trabalho em equipe, os pontos positivos, negativos e como desenvolver estratégias de crescimento.
Atualização do Estoque do Produto [*Product Backlog*]	É feita a atualização da lista de prioridade feita logo no início do projeto, com o objetivo de listar o que deve ser entregue ao cliente.

Fonte: Deemer et al. (2010).

Assim, pode-se se aplicar as práticas em conjunto com os papéis do processo *Scrum.* Segue as definições de cada prática da metodologia:

Reunião de Planejamento do Clico *Scrum* [*Scrum Sprint Planning Meeting*]: Esta reunião é realizada com a presença do Dono do Produto, Mestre da equipe *Scrum* e todos os interessados, diretores ou representantes do cliente. Durante a reunião o Dono do Produto, explica as funcionalidades que tem maior prioridade para a equipe *Scrum* e é responsável pelas as perguntas para que eles possam, depois da reunião, definir quais atividades eles irão mover do estoque do produto para o clico do estoque (SCHWABER; SUTHERLAND, 2011; CARVALHO; MELLO, 2012). Depois da reunião do Planejamento do ciclo, a equipe *Scrum* reúne-se separadamente para discutir o que foi dito e decidir o quanto eles se comprometem a

fazer durante o próximo ciclo. Em alguns casos, haverá negociações com o Dono do Produto, mas será sempre prerrogativa da equipe *Scrum* determinar o quanto eles podem se comprometer (STRAUHS, 2013; DEEMER, 2010);

Reunião Diária do *Scrum* [*Daily Scrum Meeting*]: nesse encontro diário realizado pela equipe e o Mestre da equipe *Scrum* os membros discutem o trabalho realizado, os trabalhos futuros e os possíveis impedimentos que possam interferir no progresso do trabalho. Esta reunião é uma maneira eficiente de manter os membros cientes dos objetivos e impedir que o projeto, sai do cronograma definido inicialmente. São tipicamente rápidas e objetivas, realizadas em pé, preferencialmente pela manhã, duram de 15 a 30 minutos para evitar a perda do foco do que está sendo desenvolvido e as possíveis divergências de assuntos (RISING, JANOFF, 2000; CARVALHO; MELLO, 2012).

As reuniões não representam uma forma de cobrança dos gerentes de projetos, mas é uma maneira de sincronizar a equipe às tarefas e relatar os impedimentos que podem estar interferindo no bom andamento do Ciclo. Nas reuniões diárias, os únicos autorizados a falar são o Mestre da equipe *Scrum* e a equipe envolvida. Os demais participantes devem apenas ouvir, se falarem algo devem ser convidados a se retirar da reunião, pois podem querer mudar o rumo do projeto, tentando priorizar itens que são importantes para eles e não para o projeto ou seu objetivo imediato. O Mestre da equipe *Scrum* é o responsável por manter a ordem nessa reunião, autorizar os membros a falar, cada um em sua vez, evitar desvio de foco e assuntos desnecessários (RISING, JANOFF, 2000; CARVALHO; MELLO, 2012; PAULA, 2010).

Criação do Incremento do Produto: sinaliza a realização de um incremento do produto. Os membros da equipe trabalham de maneira colaborativa de forma a realizar todas as metas definidas para ciclo a ser concluído (PAULA, 2010; CARVALHO; MELLO, 2012);

Retrospectiva do clico [*Sprint Retrospective*]: Ocorre no final de cada ciclo. Neste momento a equipe exibe o Incremento do Produto, potencialmente utilizável e que foi construído. O Dono do Produto é responsável por validar e solicitar ajustes para que o projeto se torne apropriado às necessidades do usuário/cliente. Os participantes desta reunião incluem tipicamente: o Dono do Produto, a Equipe *Scrum*, o Mestre da equipe *Scrum*, a diretoria, clientes e engenheiros de outros projetos. O ideal é que a equipe tenha concluído todos os itens do Estoque do Produto, reservados para o ciclo.

É uma reunião informal de quatro horas de duração, que ajuda a equipe de forma colaborativa a obter um *feedback* e determinar quais os objetivos da próxima iteração, assim como a qualidade e as capacidades das funcionalidades produzidas. Ao final da apresentação

das funcionalidades desenvolvidas, as partes interessadas (*stakeholders*) são questionadas, um a um, quanto às suas impressões, mudanças necessárias e alterações de prioridades. Possíveis rearranjos nos itens do Estoque do Produto ou funcionalidades não desenvolvidas como planejada, podem ser implementadas no próximo ciclo (DEEMER; BENEFIELD; LARMAN; VODDE, 2011; PAULA, 2010; STRAUHS, 2013);

Atualização do Estoque do Produto [*Product Backlog*]: O Dono do Produto é responsável por reorganizar as prioridades de toda lista de itens do Estoque do Produto, para que um próximo ciclo possa ser iniciado de acordo as prioridades dos itens.

Vale salientar que em alguns projetos de desenvolvimento nem todas as regras metodológicas são aplicáveis, devido às diversidades de funcionalidade, prazos e equipe. No entanto, é possível adaptar as metodologias adequando as funcionalidades e as melhores práticas.

As Metodologias ágeis têm sido apontadas como uma alternativa às abordagens tradicionais para o desenvolvimento de software. As metodologias ágeis tradicionais, conhecidas como orientadas a planejamentos, devem ser aplicadas apenas em situações em que os requisitos do sistema são estáveis e requisitos futuros são previsíveis (SOARES, 2004).

2.3 SEGURANÇA DE SISTEMAS E DADOS

Com o crescimento do uso da Internet, a realização de diversos serviços como: transações bancárias, comércio eletrônico, acessos à base de dados, educação à distância, tornaram-se viáveis. No entanto, a efetivação destes serviços impõe uma preocupação constante, que é a segurança dos sistemas, dos serviços e principalmente da informação. Isso ocorre devido ao uso destes serviços por pessoas que tenham intenção de prejudicar o sistema ou ainda despreparadas diante da tecnologia, as quais abrem vulnerabilidades a esses serviços citados acima, transações bancárias, acesso remoto a experimentação em tempo real e comércio eletrônico.

Dentro deste escopo, este subitem tem por objetivo investigar a segurança da informação, identificando principais ameaças, vulnerabilidade e contramedidas de segurança a sistemas e serviços disponibilizados em rede.

A segurança da informação está relacionada à proteção de dados, de redes e aplicações por meio de mecanismos tradicionais e avançados de segurança. Atualmente, há diversos textos normatizando procedimentos e melhores práticas sobre segurança. E entre eles destacam-se: BS17799 que se tornou um documento ISO [*International Organization for*

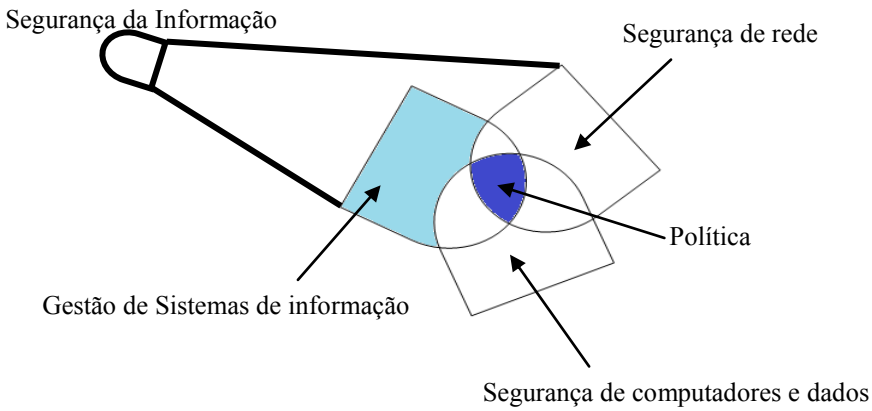
Standardization: Organização Internacional para Padronização] e incorporando uma família de outras normas denominadas série 27000. Nesta série destacam-se: a ISO 27001 que trata da gestão da segurança e a ISO 27002 que trata de requisitos e das boas práticas de segurança.

Quando uma organização adota um sistema informatizado para que tenha segurança em seus processos precisa garantir seis pilares (FERREIRA, 2006):

- **Confidencialidade**: Garantia de que a informação é acessível somente por pessoas autorizadas a terem acesso;
- **Integridade**: salvaguarda da exatidão da informação e dos métodos de processamento;
- **Disponibilidade**: garantia de que os usuários autorizados obtenham acesso à informação e aos ativos correspondentes sempre que necessários;
- **Legalidade**: o uso da informação deve estar de acordo com as leis aplicáveis, regulamentos, licenças e contratos;
- **Auditabilidade**: o acesso e o uso da informação devem ser registrados, possibilitando a identificação de quem fez o acesso e o que foi feito com a informação;
- **Irretratabilidade**: o usuário que gerou ou alterou a informação não pode negar o fato, pois existem mecanismos que garantem sua autoria.

Quando qualquer um desses itens é "quebrado" os sistemas e os dados passam a estar vulneráveis e receber ameaças que podem se transformar em incidentes de segurança. Nenhum sistema pode ser considerado 100% seguro, havendo a necessidade de se criar e/ou fazer uso de ferramentas de segurança: como antivírus, firewall e demais sistemas de proteção que servem para manter e garantir a segurança. A Figura 6 ilustra os Componentes da Segurança da Informação:

Figura 6 - Componentes da Segurança da Informação



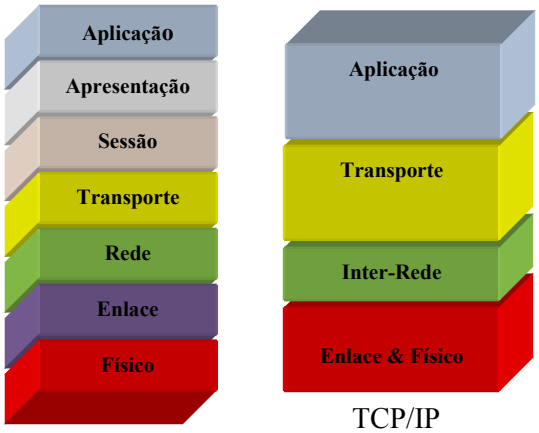
Fonte: Williams et al. (2013).

As técnicas de segurança utilizadas para proteger a informação e os sistemas consistem em técnicas criptográficas, de autenticação, por meio do uso de certificação e assinatura digital. Além disso, protocolos de segurança, filtragem de dados e comandos também são utilizados.

2.3.1 Ameaças, Vulnerabilidades em camadas.

As redes de computadores, e serviços providos por elas, são definidas, na literatura, em camadas, segundo padrões de interconexão OSI e TCP/IP, conforme apresentado na Figura 7.

Figura 7 - Arquitetura OSI e TCP/IP.



Fonte: Tanenbaum (2003).

Esta estrutura em camadas é conhecida como arquitetura de rede e define como é funcionamento do tráfego de informação na rede em diversos níveis e elementos componentes da rede. Atualmente, a arquitetura TCP/IP é a utilizada por serviços e elementos de rede, a qual será abordada sob o foco de segurança.

As camadas Enlace & Físico definem as propriedades físicas da rede, como níveis de voltagem, tipos e tamanhos de cabos, conectores, frequência, transferência confiável no meio físico. Algumas ameaças que podem afetar estas camadas interferindo na comunicação origem e destino. Como exemplos dessas ameaças há: o vandalismo (acesso cabos lógicos e de força, disjuntores, acesso a equipamentos, racks distribuídos nos prédios); a manutenção na rede elétrica interface da rede; os picos de energia afetam serviços; as redes sem fio (frequência, captura de sinal e pacotes).

A camada de Rede é responsável pelo roteamento de dados através de várias redes até o destino final, e pode ser ameaçada por vulnerabilidades em roteadores; firewalls mal

configurados; IP (IPV4 não oferece confidencialidade, pacotes podem ser lidos na rede pública ou nos provedores de serviços, ataques de *spoofing (*IPs de destino falsos*)*; ataques de *DoS* [*Denial of Service*: Negação de Serviço]; e vulnerabilidades nos algoritmos de roteamento como BGP e OSPF.

A camada de Transporte prove o controle de fluxo de dados, sequencialmente de pacotes, controle de congestionamento e retransmissão de pacotes perdidos pela camada de rede (TCP). Pode sofrer ameaças: nas aplicações TCP, UDP que são alvos de "*port scan*" que permite o mapeamento da rede, e portas abertas com o serviço rodando.

A camada de Aplicação é caracterizada pelos protocolos de alto nível e aplicações, sistemas e serviços providos na rede para clientes. As ameaças a estes serviços são intensificadas de acordo com a criticidade e custo envolvido ao prover os serviços, tanto da importância da informação quanto no custo dispendido em protegê-la. Ameaças típicas desta camada são: *Plugins* para *Browser* (*ActiveX, Applets Java*); senhas enviadas sem criptografia; vírus, *worms*, *Trojans*; falhas de software; e falha na configuração de serviços (FTP, HTTP).

Ainda ao que se refere a serviços e aplicações, as vulnerabilidades mais comuns são os controles de acesso inadequados em roteadores, Servidores de Acesso Remoto (RAS) sem proteção do firewall, serviços não utilizados rodando como serviços de gerenciamento da rede: SNMP, *finger*, SMTP, *telnet*, FTP e TFTP, cada qual com riscos específicos, e portas abertas para invasores. Senhas fracas e reutilizadas nas estações de trabalho podem permitir acessos não autorizados.

Nota-se que as vulnerabilidades estão em todos os níveis de rede. Isto se dá devido a diversos fatores, destacando-se: a complexidade dos sistemas, controles de qualidade ineficientes diante das pressões de mercado, deficiências no perfil dos programadores, aumento de dispositivos de rede, gestão ineficiente de rede, e prioridades com eficiência e não segurança.

Inicialmente, quando se aborda segurança de sistemas e serviços há algumas premissas a serem atacadas (TANEMBAUM, 2003):

Sigilo: está relacionado ao fato de manter as informações longe de usuários não autorizados. É isso que geralmente, transparece quando é discutindo sobre segurança de redes.

Autenticação: cuida do processo de determinar quem comunica com as informações sigilosas, consiste principalmente no controle de acesso de dados e serviços.

Controle de integridade: pode-se certificar de que uma mensagem recebida, por exemplo, é realmente legítima e não algo mal-intencionado que foi enviado.

Estes aspectos de segurança são essencialmente críticos e serão elementos-chave deste projeto. Assim, existem algumas contramedidas que podem ser utilizadas para se garantir a segurança, as quais são detalhadas a seguir.

2.3.2 Contramedidas de Segurança

As redes de computadores são divididas em camadas. Estas podem ser responsáveis pela segurança dos dados e serviços disponíveis nas redes. Por exemplo, a camada de enlace de dados codifica os dados para saírem da rede e descodificam ao chegar ao seu destino, à camada de rede é responsável pelos *firewalls*, e a segurança do IP está nessa camada, à camada de transporte, criptografam dados e por fim a camada de aplicação é responsável pelo controle de autenticação do usuário.

Assim, é importante detalhar algumas soluções de segurança, já tradicionalmente, utilizadas nos sistemas e serviços em rede. Dentre elas, destacam-se:

Firewall pode ser definido como uma barreira de proteção, que controla o tráfego de dados entre os equipamentos de uma rede e a Internet. Com o objetivo de permitir somente a transmissão e a recepção de dados autorizados, o *firewall* controla o acesso ao sistema por meio de regras e a filtragem de dados. O *firewall* pode ser usado para ajudar a impedir que sua rede ou seu computador seja acessado sem autorização. Em redes corporativas, é possível evitar que os usuários acessem serviços ou sistemas indevidos, além de ter o controle sobre as ações realizadas na rede, sendo possível até mesmo descobrir quais usuários as efetuou (MARIMOTO, 2008).

Autenticação é o processo de identificação das partes comunicantes de uma transação, estabelece uma conexão segura entre cliente e servidor. Umas das etapas mais importantes no processo de autenticação baseiam-se na identidade de cada usuário, para que sejam liberadas as permissões de acessos, bem como, a identificação de destino, em que o usuário tem a garantia em se conectar ao destino (servidor) válido. Existem três elementos de autenticação que são: algo que se sabe (senha); algo que possui (token); e algo que é (impressão digital). (STALLINGS, 2008; INTERNATIONAL BUSINESS MACHINES- IBM, 2013).

Assinatura Digital define-se em um método de validar e garantir integridade de dados, como em documentos digitais, vendas e contratos online. É uma segurança que utiliza operações matemáticas e criptografia permitindo a aferição da origem do documento online. (KAZIENKO 2003; STALLINGS, 2008).

Criptografia tem a responsabilidade de transformar informações em códigos, com a finalidade de assegurar as informações em uma comunicação segura (STALLINGS, 2008).

Existem dois tipos de criptografia a pública assimétrica que utiliza duas chaves distintas: uma pública, que pode ser livremente divulgada e uma privada, que deve ser mantida somente pelo seu dono, a informação e codificada com umas das chaves e somente a outra chave do par pode decodifica-la. A escolha da chave depende do tipo de proteção. Essa chave pode ser armazenada de diversas formas: arquivo no computador, um *smartcard* ou um *token*. A privada simétrica utiliza a mesma chave para codificar e descodificar as informações. Há diversos algoritmos aceitos como padrões mundiais, que garantem a segurança de dados e serviços (CERT, 2014). Função Hashing é um exemplo de criptografia que pode ser usando para criptografar senhas no banco de dados, as principais funções são: AES, RSA, SHA-1, SHA-2, MD2, MD4 e MD5 (OLIVEIRA, 2012).

Certificação X509 é formato de certificado que obedecem ao padrão internacional ITU-T X.509. Usado no processo de autenticação (uso de chaves públicas) e sigilo de informações (uso de chaves privadas). A validade de um certificado é realizada por meio de Autoridades Certificadoras, o que é bastante utilizado para identificar as partes em uma transação ou acesso a sistemas e serviços (MOREIRA, 2003).

As recomendações NIST SP 800-82 e NIST SP 800-53 definem controles de segurança para sistemas de controle industriais. Estas recomendações trazem contramedidas de segurança segundo controles de gerenciamento, operacional e técnico, enfatizando as melhores práticas (NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY-NIST, 2009 – 2011).

Controles de Gerenciamento: são contramedidas de segurança que tem por objetivo o gerenciamento de riscos e gerenciamento de segurança da informação. Segundo NIST SP 800-53, cinco famílias de controles são definidas: avaliação e autorização de segurança; planejamento; avaliação de riscos; e aquisição de serviços e sistemas;

Controles Operacionais: são contramedidas de segurança que são primariamente implementadas por pessoas e sistemas opostos. Segundo NIST SP 800-53, nove famílias de controles são definidas: segurança pessoal; proteção ambiental e física; planejamento de contingências; gerenciamento de configurações; manutenção; integridade de sistemas e de informação; proteção de mídias; respostas a incidentes; treinamento e sensibilidade.

Controles Técnicos: são contramedidas de segurança que são implementadas e executadas por sistemas, através de mecanismos vinculados a *hardware*, *software*, *firmware* do sistema. Segundo NIST SP 800-53, quatro famílias de controles são definidas: autenticação e identificação; controle de acesso; auditoria e contabilidade; e proteção de comunicações e sistemas.

Estas **soluções de segurança** não são exaustivas, havendo outras e combinações delas. Estas soluções são utilizadas de acordo com a criticidade do sistema e das informações que se quer assegurar, e pelo tempo em que se fazem relevantes, segundo **controles de segurança** que se quer implementar.

Com base nestas soluções e contramedidas de segurança investigadas, o presente projeto deverá especificar, implementar, implantar e testar a segurança do SiSLQEE que o responsável pelo acesso remoto seguro ao ambiente de experimentação do LQEE, em tempo real.

2.3.3 Testes em Sistemas Computacionais

Conforme apresentado no item anterior (2.2), as metodologias de desenvolvimento de software abordam o criação de testes para análise de eficiência do sistema implementado. Estes testes são imprescindíveis para o desenvolvimento confiável de sistemas. Além destes testes, quando abordamos sistemas em redes, há um novo conjunto de ameaças e vulnerabilidades, exigindo a criação de testes de análise de segurança. Para isto são necessários os seguintes elementos para os testes de segurança, os quais foram investigados:

- Definição de Ferramentas de Análise;
- Análise de Vulnerabilidades;
- Definição de Ferramentas de Correções;
- Implantação das Melhores Práticas.
-

2.4 WINDOWS SERVER 2008 E SEUS RECURSOS

As empresas a cada dia estão, cada vez mais, capacitando seus funcionários e prestadores de serviços a trabalhar fora da empresa, em casa, em uma viagem ou de uma instalação terceirizada. São novos ambientes de trabalho que já vem surgindo há algum tempo e inovando o mercado de trabalho e as universidades e com isso aumentando a flexibilidade, reduzindo custos e a necessidade de grandes ambientes corporativos. As tecnologias e facilidades são necessárias para aumentar a segurança do acesso remoto, pois são dados corporativos valiosos e precisam ser mantidos de maneira segura, com o crescimento do uso do acesso remoto nas organizações. É necessário aplicar as técnicas de seguranças, que protegem as informações das empresas e dos usuários.

O Sistema Operacional Windows Server 2008 *R2* foi desenvolvido sobre os recursos e as capacidades do Windows Server 2008, o R2 permite a criação de soluções para

organizações mais fáceis de planejar, implantar e gerenciar do que as versões anteriores do Windows Server. Utilizando mais recursos de segurança, confiabilidade e desempenho fornecidos pelo Server 2008, o R2 estende a conectividade e o controle para recursos locais e remotos. Isso significa que as organizações podem se beneficiar da redução de custos e do aumento da eficiência obtidos por meio do melhor gerenciamento e controle dos recursos em toda a corporação (MICROSOFT, 2008; RUSSEL; ZACKER, 2010). Seguem nas seções abaixo, alguns recursos do Windows Server 2008 R2, que possível utilizar para proteger informações por meio das diretivas de segurança do sistema operacional para servidores. Seção 2.4.1 descreve os serviços disponibilizados pelo *Active Directory*, a seção 2.4.2 será apresentada os Serviços de Área de Trabalho Remota e na seção 2.4.3 é a descrição dos requisitos do dispositivo do cliente, para utilizar os serviços de RDP.

2.4.1 Active Directory

Entre os diversos recursos providos pelo Windows Server 2008, o AD [*Active Directory*: Diretório Ativa] consiste no serviço de diretório bastante aceito pelo seu desempenho e segurança, sendo uma ferramenta de fácil gerenciamento de informações (MICROSOFT, 2008; RUSSEL; ZACKER, 2010).

O *AD* fornece os meios para se gerenciar identidades e relacionamentos de usuários. Integrado ao *Windows Server 2008 R2*, permite configurar e administrar centralmente os parâmetros do sistema, de usuários e aplicativos (MICROSOFT, 2008; RUSSEL; ZACKER, 2010). Por meio dos seguintes serviços:

- **ADCS [*Active Directory Services Certificates:* Serviço de Certificado do Diretório Ativo]** fornece serviços para emissão e gerenciamento de certificados de chaves públicas usados em sistemas de segurança de software que empregam tecnologia de chave pública (MICROSOFT, 2008; RUSSEL; ZACKER, 2010).
- **ADDS [*Active Directory Domain Services:* Serviço de Domínio do Diretório Ativo]** permite criar uma infraestrutura escalável, segura e gerenciável para o gerenciamento de usuários e de recursos; fornece suporte para aplicativos habilitados para o diretório; gerenciar usuários, computadores, grupos, impressoras, aplicativos e outros objetos habilitados para diretório, de forma eficiente em um local centralizado e seguro (MICROSOFT, 2008; RUSSEL; ZACKER, 2010).
- **ADFS [*Active Directory Federatin Services:* Serviço de Federação do Diretório Ativo]** esse serviço é usado pra criar uma solução de acesso a identidades que seja

segura, para diferentes plataformas, inclusive em outros sistemas operacionais (MICROSOFT, 2008; RUSSEL; ZACKER, 2010).

- **ADLDS [*Active Directory Lighweight Directory Services:* Serviços de Diretório Leves do Diretório Ativo]** esse serviço é usado para habilitar aplicativos para o diretório sem causar sobrecarga de domínios na rede e derivações (MICROSOFT, 2008; RUSSEL; ZACKER, 2010).
- **ADRMS [*Active Directory Rights Management Services:* Serviços de Gerenciamento de Diretórios do Diretório Ativo]** esse serviço aumenta a segurança de uma organização protegendo as informações por meio de diretivas de uso persistentes, que permanecem com as informações, independentemente do local para onde forem movidas. O ADRMS é usado para evitar que informações essenciais, como relatórios financeiros, especificações de produto, dados de cliente e e-mails confidenciais caiam em mãos erradas acidentalmente ou intencionalmente (MICROSOFT, 2008; RUSSEL; ZACKER, 2010).

2.4.2 Serviços de Área de Trabalho Remota

Os Serviços de Área de Trabalho Remota oferecem soluções que permitem aos usuários acessar programas instalados em um servidor de Sessão da Área de Trabalho Remota, ou acessar a área de trabalho completa do Windows, podendo acessar um servidor de uma rede corporativa ou na Internet (MICROSOFT, 2008; RUSSEL; ZACKER, 2010).

O RDP [*Remote Desktop Protocol*: Área de Trabalho Remota] permite que os usuários executem aplicativos em um servidor remoto, por meio de conexões seguras de rede, em que são utilizados é utilizada autenticação e criptografia para proteger o tráfego de informações (MICROSOFT, 2008; THOMPSON, 2010). Serviço que disponibiliza administrar um servidor remotamente, executando uma aplicação ou uma área de trabalho completa do servidor a partir de um computador cliente, por meio de rede. Uma vez conectado, o RDP permite controlar com o mouse e teclado o computador remoto, enquanto é mostrada no computador local a tela do ambiente operacional remoto (MICROSOFT, 2008; THOMPSON, 2010).

Depois de estabelecida conexão entre um cliente e um servidor com o Serviço de Área de Trabalho Remota, apenas comandos de mouse, teclado e a tela do ambiente são enviadas pela rede. É utilizado para a comunicação entre um cliente e um servidor o RDP a porta padrão TCP 3389, serviço que pode ser executado em dois modos. (BORGES, 2002; MICROSOFT, 2008).

A Infraestrutura de *VDI* [*Virtual Desktop Infrastructure: Área de Trabalho Virtual]* e serviços de gerenciamento e desempenho emprega virtualização baseado nos Serviços de Terminal, também conhecidos como Serviços da Área de Trabalho Remota (BORGES, 2002). Como ilustrado na Figura 8.

Figura 8 – Virtualização de dados

Fonte: elaboração do próprio autor.

A VDI é uma arquitetura de área de trabalho que permite ao cliente centralizar o armazenamento, a execução e o gerenciamento de uma área de trabalho do Windows no *datacenter*. Com isso pode-se ter acesso a outros Sistemas Operacionais além do instalado na máquina em outros ambientes de área de trabalho, podem ser abertos e gerenciados em máquinas virtuais ou em um servidor centralizado (MICROSOFT, 2008; THOMPSON, 2010).

2.4.3 Requisitos do dispositivo do cliente

Para usar o RDP, os computadores (Tablet, Iphone, Android, etc) do cliente devem ter acesso à Internet, e uma versão da Conexão de Área de Trabalho Remoto que forneça suporte ao RDP 6.1, no sistema operacional Windows. Já é um serviço do sistema operacional, é necessário somente ativar no dispositivo, e em outros sistemas operacionais é necessário usar um RDP de terceiros com versões para cada dispositivo.

2.5 Considerações Finais do capítulo

Nesse capítulo foram apresentados conceitos, técnicas e práticas de segurança, para melhor proteger informações em uma terminada máquina, com isso a necessidade de se utilizar alguns dos recursos do Sistema Operacional Windows Server 2008 R2, para as configurações de segurança dos servidores de aplicação e dados. Os detalhes dos recursos adotados seguem descritos no quarto capítulo.

Tomando como base os trabalhos apresentados nesse capítulo, determinaram-se as técnicas a serem adotadas para proporcionar aos usuários e integrantes do LQEE, um acesso remoto seguro aos sistemas de experimentos, utilizando ferramentas e conceitos já utilizados em outras pesquisas.

Além disso, foram apresentadas as técnicas da Engenharia de Software, identificando as práticas e papéis, das metodologias ágeis para desenvolvimento de software, das quais algumas serão adotadas, para o desenvolvimento do SiSLQEE. Princípios e mecanismos de segurança, também, foram investigados, bem como a necessidade da análise e testes no software e servidores de dados e web. Segue no próximo capítulo a descrição do funcionamento do LQEE e os requisitos para o desenvolvimento do sistema e acesso remoto seguro.

3 ACESSO REMOTO A EXPERIMENTOS DO LABORATÓRIO DE QUALIDADE DE ENERGIA ELÉTRICA (LQEE).

Nessa seção serão apresentadas: uma descrição sobre o Laboratório de Qualidade de Energia Elétrica, uma descrição sobre o programa de experimentos DASYLab 13® e o problema da disponibilização do sistema de experimentos na rede de internet.

3.1 O LQEE

O Laboratório de Qualidade de Energia Elétrica é uma unidade auxiliar de ensino, pesquisa e prestação de serviços do Departamento de Engenharia Elétrica da Faculdade de Engenharia de Ilha Solteira. Trabalha atualmente com as linhas de Pesquisa:

1. Desenvolvimento de transformadores de potência assimetricamente magnetizados;
2. Degradação de vida útil de capacitores em sistemas elétricos poluídos;
3. Tecnologias energeticamente eficientes na qualidade de energia elétrica;
4. Sensibilidade de equipamentos de eletroeletrônicos a mergulhos de tensão;
5. Filtros de harmônicas;
6. Fontes harmônicas.

No LQEE destacam-se duas áreas: Área de Recursos Computacionais e Desenvolvimento de Equipamento que se concentra em recursos de software e hardware necessários para o desenvolvimento dos diferentes trabalhos de pesquisa e a Área de Ensaios e Estudos Experimentais-que se destina a realização de testes de equipamentos e dispositivos de potência, com destaque para itens relacionados com a qualidade de energia, bem como os estudos experimentais associados às pesquisas em desenvolvimento.

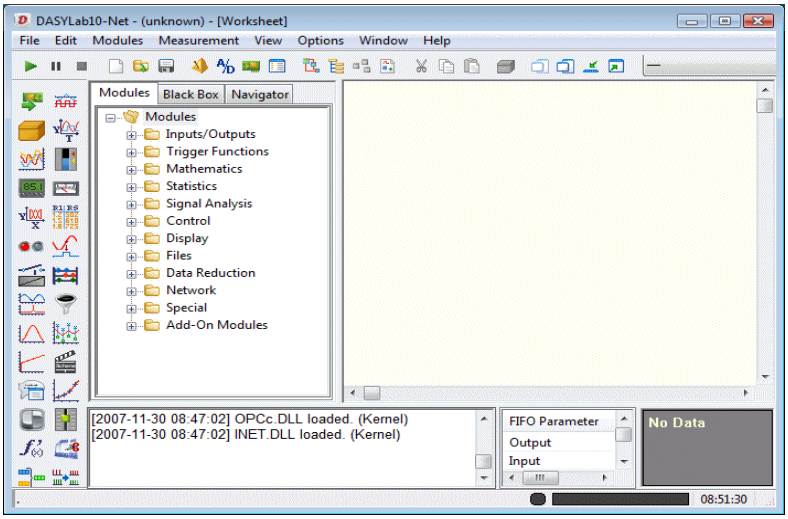
É composto por equipamentos específicos tais como: fontes de alimentação e geradores de distúrbios elétricos; Sintetizador de formas de Onda e Analisador de Harmônicas Trifásico; Sistemas de aquisição de dados; Analisadores de energia e grandezas elétricas (FACULDADE DE ENGENHARIA- FEIS-LQEE, 2014).

3.2 SOFTWARE DE EXPERIMENTAÇÃO: DASYLab

O DASYLab 13® é um programa de computador que permite aquisição e análise de dados. Esses dados advêm de experimentos de qualidade de energia elétrica, notadamente sinais de tensão e corrente. DASYLab 13® é uma aplicação de controle representado por um diagrama de fluxo de dados diretamente na tela, na que os ícones selecionados a partir de uma

barra de módulo são posicionados e ligados com o mouse para formar um fluxograma esquemático das diferentes fases no fluxo de dados - entrada, processamento e saída. Este fluxograma esquemático é chamado de planilha (DASYLab, 2013). A edição da planilha é fácil, e os módulos podem ser adicionados, movidos e apagados com um clique do mouse (DASYLab, 2013). A Figura 11 mostra a tela principal do programa DASYLab.

Figura 9 - Tela DASYLab



Fonte: Elaboração do próprio autor.

3.2.1 Funcionalidades

As principais funcionalidades do *DASYLab* são:

- Criar aplicativos complexos em tempo mínimo sem programação com linguagem específica;
- Construir planilhas utilizando funções gráficas;
- Implementar operações em tempo real;
- Fornecer monitores padrão em tempo real (gráficos e medidores);
- Fornecer biblioteca completa de funções de computação;
- Criar funções geradoras para simular entradas;
- Fornecem exemplos funcionais e ajuda on-line;
- Suportam *hardware* de aquisição de dados. (DASYLAB, 2013).

3.2.2 Requisitos do Software

O DASYLab exige um mínimo de hardware e configuração de *software* para trabalhar sem restrição:

Quanto ao Hardware necessário:

- processador x86 compatível, 1 GHz ou mais;
- pelo menos 1 GB de RAM exigida; 2 Gbyte ou mais recomendados ;
- espaço livre em disco de 250 MB, dos quais pelo menos 200 MB é na partição do sistema operacional;
- profundidade de cor de 24 ou 32 bits (True Color), com uma resolução de pelo menos 1024x768 pixels;

Quanto ao Sistema Operacional, atende ao:

- Windows XP Pro, com o Service Pack 3;
- Windows Vista, 32-bit e 64-bit (como uma aplicação 32-bit);
- Windows 7, 32-bit e 64-bit (como aplicação de 32-bit);

3.3 EXPERIMENTAÇÃO REMOTA

Muitos dos experimentos remotos do LQEE são realizados por meio da ferramenta de processamento de sinais DASYLab 13®. Atualmente, o agendamento dos experimentos é feito manualmente pelo professor, aluno ou auxiliar do professor, via e-mail, disponibilizando uma planilha do Excel para cada aluno, na qual todos podem ver os horários agendados. Há a possibilidade de troca de horários entre os alunos.

O responsável pelos equipamentos (professor ou auxiliar) deve ficar todo tempo do experimento em frente à máquina monitorando os experimentos, caso o aluno clique com o botão errado ou o uso não permitido do mouse, o responsável pelo experimento (professor ou auxiliar) deve estar alerta para desfazer qualquer operação não autorizada.

Na UNESP o sistema funciona em um computador do LQEE, os alunos acessam via Internet por meio de um programa de acesso remoto o VNC [*Virtual Network Computing*: Computação de Rede Virtual] para realizar os experimentos, tendo liberdade de entrar e sair do ambiente sem qualquer critério.

Na atual situação é possível várias pessoas acessarem o sistema e tentar realizar o experimento ao mesmo tempo. Por isso é necessário que o responsável pelos equipamentos acompanhe todos os experimentos. E quem permite isso é o acesso remoto e não o sistema de experimentos. À medida que os alunos vão realizando os experimentos, o responsável registra manualmente todas as informações, hora acessada, se realizou totalmente o experimento, se trocou de horário. E envia manualmente via e-mail os dados para o professor responsável.

3.4 Considerações finais do capítulo

Em virtude da situação vulnerável em que se encontra o acesso aos experimentos que são realizados no LQEE, aplicaram-se os conceitos, tecnologias e técnicas para melhorar a disponibilidade dos experimentos do LQEE em tempo real aos usuários. É fundamental que a solução de acesso remoto incorpore: segurança de rede (firewall, certificação digital); conexão de um usuário por vez; bloqueio de algumas funcionalidades do mouse; e controle de usuários. Para isto, uma arquitetura inicial de acesso e uso do ambiente de experimentação foi projetada, conforme será apresentado no capítulo a seguir, a arquitetura apresentada no quinto capítulo tem o objetivo de solucionar o problema do LQEE, por meio de uma aplicação Web Java em conjunto com a tecnologia VNC, que disponibilizando o acesso aos sistemas de experimento DASYLab, de um forma segura, administrado e monitorando o servidor de experimentos virtualmente e para chegar a uma arquitetura final, o projeto foi dividido em etapas que estão detalhadas no quarto capítulo.

4 ARQUITETURA INICIAL

Para encontrar a melhor solução, para realizar o acesso remoto seguro, foram realizados alguns testes, os quais resultaram no projeto de uma arquitetura inicial. Esta arquitetura foi sendo trabalhada em três etapas. Inicialmente permitindo o acesso ao DASYLab, com os recursos de segurança do Windows Server, e na sequência incorporando funcionalidades com o Sistema SiSLQEE e os mecanismos de segurança.

De acordo com as técnicas estudas sobre Engenharia de Software o referencial teórico, para o desenvolvimento deste projeto, foram adotadas e aplicadas algumas práticas XP e *Scrum,* formando um sistema híbrido de métodos ágeis para o processo de desenvolvimento do projeto, juntamente com as técnicas de segurança.

Para iniciar o projeto os requisitos foram separados e discutidos com a equipe envolvida no projeto: solicitantes de uma solução, pra o problema que o LQEE possui, “Professor Orig ” “Aluno de Doutor do Rodrigo” em reuniões condu id s pelo o gerente de projeto represent do pel “Professor Christi ne M r ie”. No primeiro momento for m discutidas as primeiras prioridades do projeto detalhadas na arquitetura inicial – primeira etapa.

No segundo momento foram discutidas as prioridades da arquitetura inicial – segunda etapa, além disso, a definição das ferramentas que foram utilizadas, a linguagem de programação e o banco de dados mais adequados para este projeto. O redator técnico representando por mim “Alun de Mestr do responsável pelo projeto An K rin ” notei todas as decisões tomadas, e detalhei cada etapa neste trabalho em tabelas, em cada etapa é possível visualizar a prática aplicada no processo de desenvolvimento do projeto.

No terceiro momento foram discutidos o avanço desse acesso remoto, e desenvolver uma aplicação Web, para monitorar e administrar o sistema de experimento, então foi aplicado às práticas XP e *Scrum*, para identificar as prioridades do sistema. Detalhados Arquitetura Inicial – Terceira etapa: Solução Implantada.

4.1 Arquitetura Inicial – Primeira etapa.

Segue na tabela 3 a primeira etapa do projeto, descrevendo metodologias utilizadas, práticas e descrição de cada uma delas.

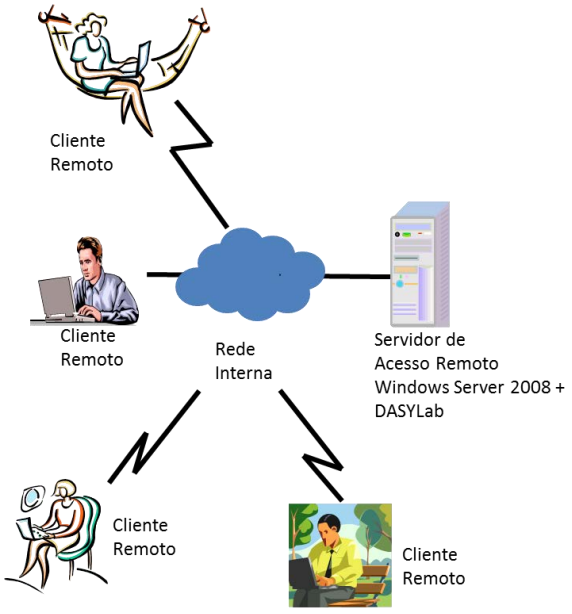
Tabela 3 – Práticas XP e *Scrum* utilizadas na primeira etapa.

Práticas	**Metodologia**	**Descrição**
Planejamento + Reunião	XP + *Scrum*	Reuniões realizadas com os membros do LQEE e o desenvolvedor, para levantamento de requisitos, identificação da prioridades de desenvolvimento do projeto, as quais destacam-se: proteger os dados e acessar o servidor do sistema de experimentos.
Teste	XP	Realização de testes após desenvolver as prioridades identificadas.
Retrospectiva do ciclo	*Scrum*	Reunião para avaliar os aspectos positivos e negativos, e determinar novas etapas, melhorando a que foi desenvolvida nessa etapa.

Fonte: Elaboração do próprio autor.

Esta **primeira** etapa tem por objetivo permitir o acesso remoto dos usuários à aplicação DASYLab, com restrições de uso, bloqueio/liberação de acesso de usuários cadastrados. Para isso, os recursos de segurança e acesso remoto do sistema Windows Server 2008 foram explorados. A figura 10 ilustra a arquitetura inicial primeira etapa.

Figura 10 – Arquitetura Inicial – primeira etapa



Fonte: Elaboração do próprio autor.

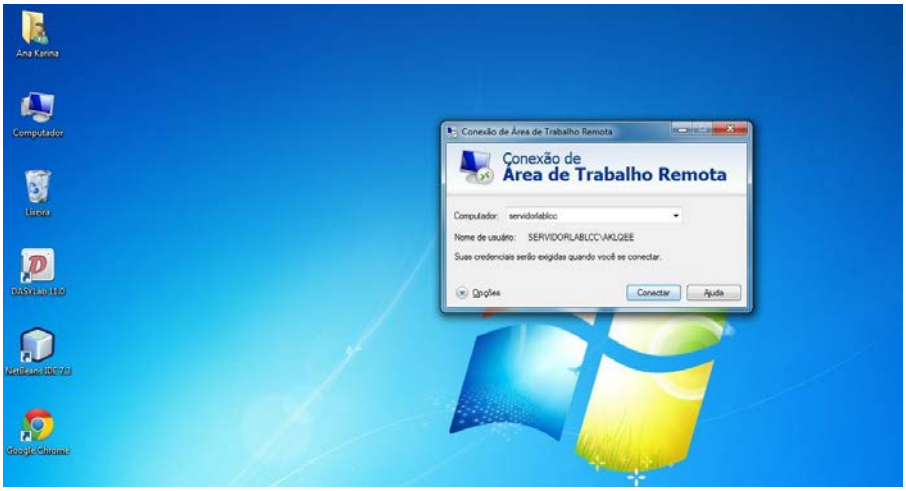
Para testes iniciais um computador **cliente** com o Windows Seven foi instalado, o qual é responsável pelo acesso remoto ao **servidor** de plic ção "teste" do L or tório de Qualidade de Energia Elétrica, com Sistema Operacional Windows Server 2008. As configurações do *Active Directory* foram realizadas para liberar o acesso do servidor na rede, interna e externamente.

Os principais serviços utilizados foram:

1. O ADCS foi ativado para a criação de um certificado digital para identificação do servidor no dispositivo do cliente;
2. Com o serviço ADDS ativado, foram criados os grupos e contas de usuários autorizados a acessar o servidor;
3. E por fim o serviço de ADRMS foi ativado para fazer a proteção das informações dos usuários e do servidor. Após a configuração do servidor, foi liberado o acesso via Conexão de Área de Trabalho Remoto em rede local e em rede externa, dentro das limitações especificadas.

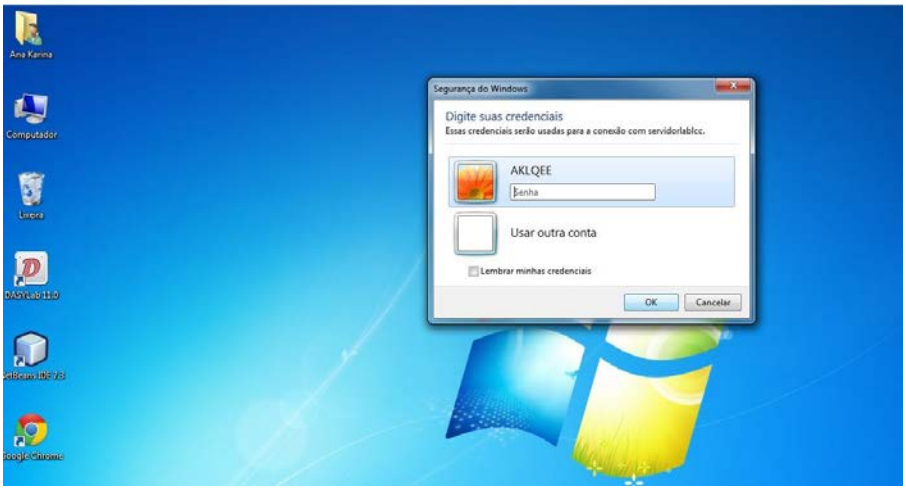
As Figuras 11 e 12 ilustram o acesso RDP.

Figura 11 - Conexão de Área Remota



Fonte: Windows Seven (2013).

Figura 12 - Informando credenciais



Fonte: Windows Seven (2013).

Após clicar em conectar é necessário informar o *login* de usuário e senha cadastrado no servidor e clicar em OK.

A tela do servidor ficará disponível e se o usuário que efetuou o login for **Administrador**. Então se houver outro usuário acessando o sistema, perde-se a conexão do usuário **não administrador**. Há configurações que permitem acesso com mais de um usuário ao mesmo tempo, no servidor em contas diferentes, mas neste teste foram cadastrados usuário administrador e usuário padrão, em que este tem somente acessos limitados e pode acessar somente o que foi determinado pelo administrador do sistema.

Para sair da tela do servidor e desconectar da área remota, basta clicar no botão fechar que aparece na parte superior da tela, aparecerá uma mensagem dizendo o que vai acontecer e se estiver de acordo clica no botão ok, para sair da conexão de área de trabalho remota.

Esse acesso pode ser feito de qualquer outro dispositivo, que tenha um programa que faça o serviço do RDP do Windows e que tenha acesso à Internet.

Nessa primeira etapa, o teste inicial foi resolver o problema do acesso remoto priorizando a segurança ao sistema do LQEE, então utilizou-se os recursos do *Windows Server 2008* e *Active Directory*.

4.2 Arquitetura inicial – segunda etapa (Applet Java)

Seguindo o uso das metodologias ágeis, foram descritas as práticas detalhadamente realizadas em cada um delas na segunda etapa do projeto. Ilustrada na Tabela 3 as práticas e suas descrições:

Tabela 4 – Práticas XP e *Scrum* e suas descrições

Práticas	**Metodologia**	**Descrição**
Planejamento + Reunião	XP + *Scrum*	Reuniões realizadas com os membros do LQEE (Aluno de Doutorado responsável pelos experimentos, Professor Origa o solicitador do projeto, Professora Christiane (Gerente de Projetos) e Ana Karina (Analista e Desenvolvedora)) e o desenvolvedor, para análise de requisitos. Definição da lista de prioridades: - definição das ferramentas a serem utilizadas para o desenvolvimento; - a criação de um banco de dados inicial; - desenvolver uma *Applet* Java, que identifica os usuários e permite o acesso ao sistema de experimentos; - bloquear ações não permitidas por usuários dentro do DASYLab.
Teste	XP	Realização de testes após desenvolver as prioridades identificadas.
Retrospectiva do ciclo	*Scrum*	Reunião para avaliar os aspectos positivos e negativos, e determinar novas etapas, melhorando a que foi desenvolvida nessa etapa.

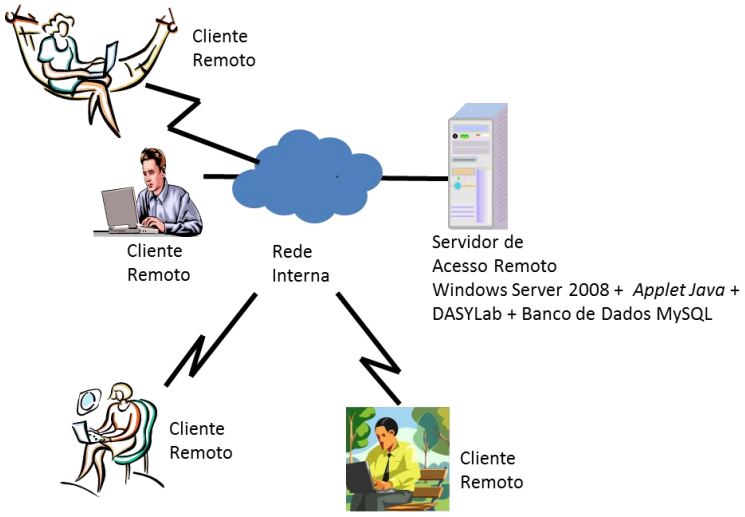
Fonte: Elaboração do próprio autor.

A **segunda** etapa preocupa-se com o bloqueio/liberação de funcionalidades do sistema, para isto há a necessidade de restrições do uso do mouse e teclado, cuja solução foi explorada através do uso de "*Applet Java*" no desenvolvimento de um plic ção.

Em conjunto com as técnicas e soluções apresentadas no segundo capítulo, foi desenvolvido uma *Applet Java* responsável pela autenticação do usuário e execução do software DASYLab. Este recurso permite executar ações na máquina remota com permissão do usuário a fim de disponibilizar o serviço de experimentação remota de forma segura e confiável, autorizando assim, a realização de experimentos e simulações

via web. O modelo definido herda características da arquitetura tradicional cliente/servidor, como pode ser visualizado na Figura 13.

Figura 13 – Arquitetura Inicial – Segunda etapa



Fonte: elaboração do próprio autor.

Este modelo possui elementos e funcionalidade, conforme descrito abaixo:

- **Usuário Pesquisador**: cliente que acessa o serviço via browser, com permissões de usuário remoto. Estas permissões são dadas pelo administrador do sistema. As permissões variam desde o controle total do seu experimento, até bloqueio total de suas ações, passando a ser um usuário espectador.
- **Usuário Administrador:** cliente que acessa o serviço via browser, com permissões de usuário administrador. Possui permissões totais de controle dos experimentos, bem como, de autorizar e configurar permissões de usuários remotos.
- **Servidor:** provê serviço de experimentação remota controlada. O controle é realizado por meio de: certificação digital, permissões de usuários em um banco de dados e bloqueios realizados via software, configurações do sistema operacional e serviços de diretório.

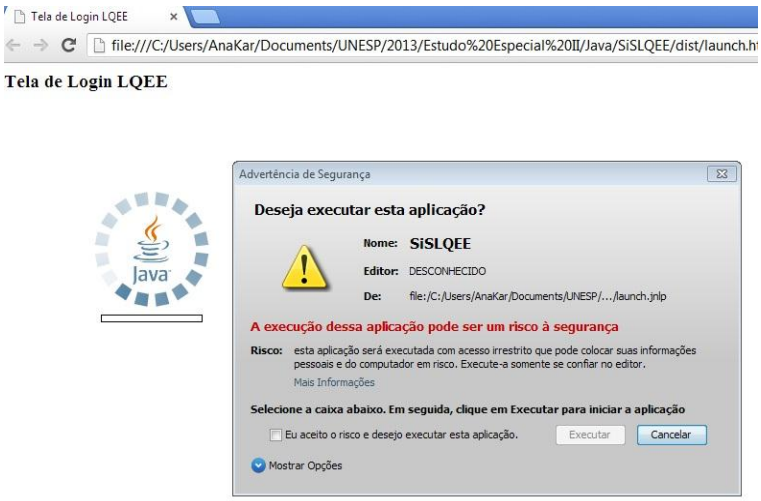
Nesse servidor instalou-se a aplicação *Applet Java*, o software DASYLab, o Administrador de Banco de Dados MySQL. Esses três softwares em conjunto permitem o acesso do usuário ao programa DASYLab por meio da execução *Applet Java*.

Inicialmente, foi definido um usuário e senha padrão previamente cadastrado manualmente no banco de dados. A verificação do usuário e senha é feita e então é permitido o acesso ao software de experimento DASYLab.

No desenvolvimento dessa aplicação foi escolhida a linguagem de programação Java, por ser uma linguagem de programação orientada a objetos, simples e portável a todas as plataformas e sistemas operacionais, tanto o código fonte como os binários. Como plataforma de desenvolvimento foi utilizada o software *Netbeans* por ser um ambiente integrado, modular e baseado em padrões IDE [*Integrated Development Environment:Ambiente de Desenvolvimento Integrado*].

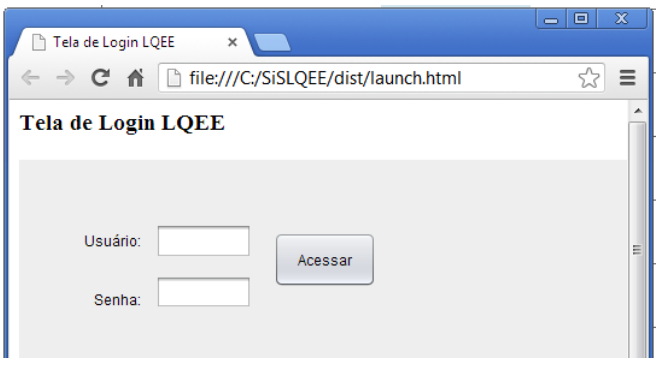
O SGBD escolhido foi o MySQL, por ser de código aberto, gratuito e de amplo uso na Web. Abaixo, segue a Figura 14 e 15, com a ilustração do funcionamento da *Applet Java*.

Figura 14 - Executando a Applet Assinada – Tela de Login



Fonte: Elaboração do próprio autor.

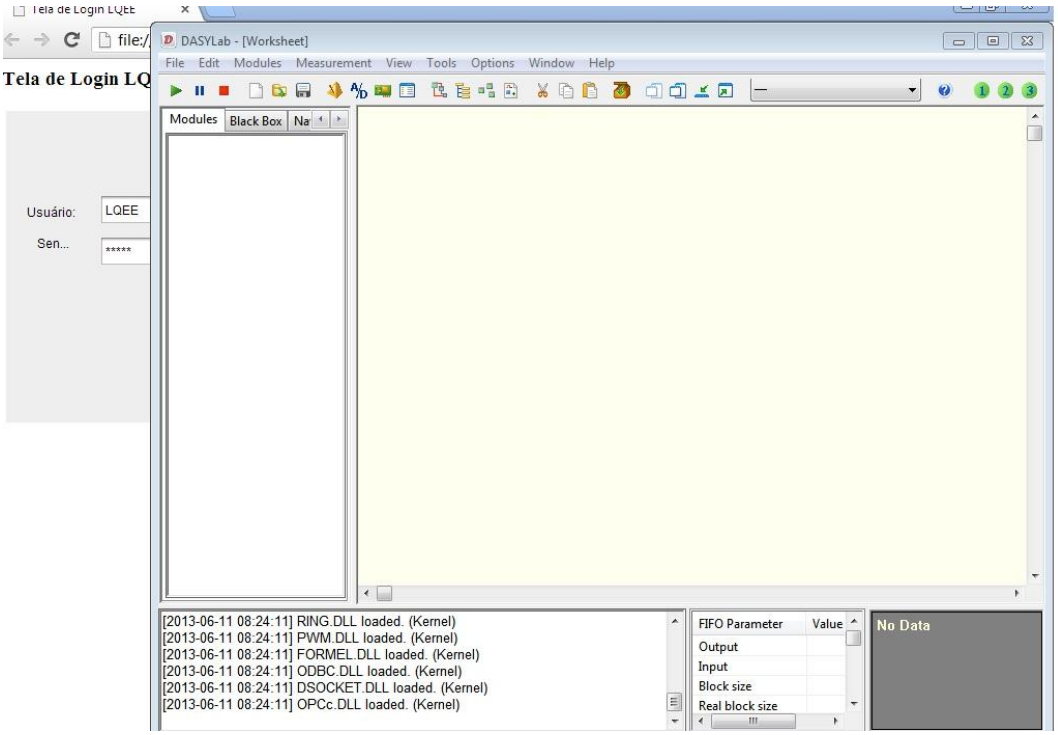
Figura 15 - Tela de Login/Usuário e Senha



Fonte: Elaboração do próprio autor

Após m r c r opção que “Eu ceito o risco e desejo execut r est plic ção” e clicar em executar a *Applet*, a aplicação DASYLab é exibida via navegador (*browser*). A Figura 16 apresenta a Tela de abertura do DASYLAB via Applet Java.

Figura 16 - Tela de abertura do DASYLab via Applet Java



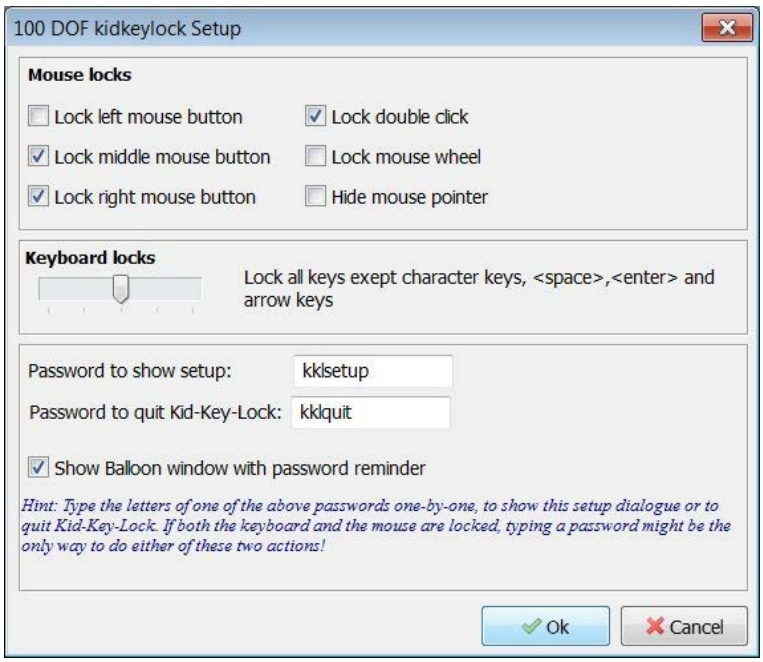
Fonte: Elaboração do próprio autor.

Esta foi uma primeira solução encontrada para resolver o problema do LQEE. Esta solução, em um primeiro momento, resolveria o problema do acesso indesejado de usuários não autorizados, ainda tem o problema do usuário tem opção de fazer alterações não aceitas no DASYLab. Então encontrou-se se a necessidade de bloquear o botão direito do mouse e o clique duplo na tela do software de experimentos. Para isso, foram pesquisadas ferramentas de terceiros que podem fazer essas restrições. Segue no item 6.1.2.1, a definição da ferramenta utilizada pra esse bloqueio.

4.2.1 Controle de Funcionalidades da Aplicação

Para proteger os dados e impedir o usuário de acessar configurações ou pressionar algumas teclas não permitidas, foi utilizado um programa de terceiros para fazer esse bloqueio de mouse e teclado, o *Kid-Key-Lock* (DOF, 2014), conforme configurações apresentadas na Figura 17 do programa de bloqueio.

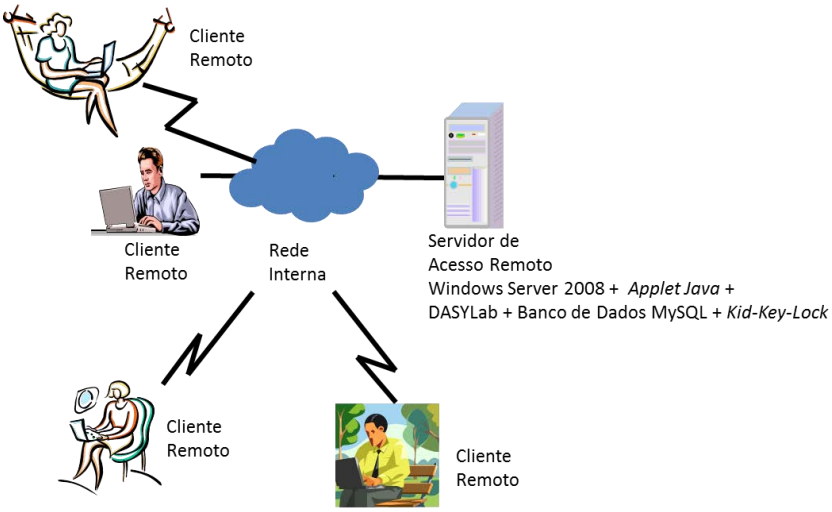
Figura 17 - Tela de configurações do Kid-Key-Lock



Fonte: Elaborada pelo autor.

Este programa (Kid-Key-Lock) foi desenvolvido por 100dof, e consiste em um utilitário simples e gratuito para bloquear funções de teclado e mouse específicos. Simples de instalar e configurar têm opções de bloquear todos os cliques do mouse e as teclas do teclado (DOF, 2013). Na Figura 18 é possível visualizar a arquitetura inicial – Segunda Etapa, demostrada em imagem.

Figura 18 – Arquitetura Inicial – segunda etapa com *Kid-Key-Lock*



Fonte: elaboração do próprio autor.

Essa foi à primeira solução (primeira e segunda etapa) encontrada para restringir a realização do experimento, somente usuários autenticados, em que um por vez acessa e realiza o experimento, no entanto com limitações de uso por determinados dispositivos.

Assim, para uma comodidade maior aos usuários e administradores do LQEE, foram estudadas e aplicadas técnicas de programação para desenvolver uma aplicação web para realizar o acesso remoto em conjunto com a tecnologia VNC, conforme apresentado a seguir.

4.3 Arquitetura Inicial – terceira etapa: Solução Implantada

A **terceira** etapa consiste na elaboração de um software capaz de gerenciar usuários e experimentos, cadastrando usuários, definindo perfis, controlando agendamento dos experimentos e uso do sistema. Este software foi intitulado **SiSLQEE** e consiste em uma aplicação Web desenvolvida em JAVA em conjunto com a tecnologia VNC, possibilitando acesso remoto ao servidor de aplicações/experimentos, com segurança, comodidade e qualidade.

Esta terceira etapa será detalhada no quinto capítulo.

5 SiSLQEE e MECANISMOS DE SEGURANÇA

Este capítulo apresenta o processo de desenvolvimento da **terceira** etapa da arquitetura de controle de acesso seguro a experimentos do LQEE.

Esta arquitetura incorpora funcionalidades da arquitetura cliente-servidor, em que clientes requisitam serviços de um servidor remoto. Os clientes são usuários que realizarão experimentação remota ou administração do sistema, e a máquina servidora será o servidor de aplicação do sistema de experimentação, com o software DASYLab instalado.

5.1 Planejamento do SiSLQEE

Para determinar, as prioridades do processo de desenvolvimento da aplicação Web, e aplicar as regras de segurança, continua-se usando os métodos ágeis. Seguem na Tabela 6, as práticas, a identificação e as descrições como foi aplicado à metodologia.

Tabela 5 – Práticas XP e *Scrum* utilizadas na terceira etapa

Práticas	**Metodologia**	**Descrição**
Planejamento + Reunião	XP + *Scrum*	Reuniões com os membros do LQEE e o desenvolvedor para analisar e definir os dados coletados no levantamento de requisitos certificar que estão corretos: Isso é feito por meio de perguntas pelo desenvolvedor e de acordo com as respostas dos membros do LQEE, e dos dados são identificados os requisitos para o desenvolvimento. É feita a identificação e definição da Lista de prioridades, que são: - uso de ferramentas CASE, para desenvolver os diagramas da UML necessários; - atualização do banco de dados; - codificações necessárias para o funcionamento do sistema (cadastros, relatórios, agendamentos, monitoramentos e administração);

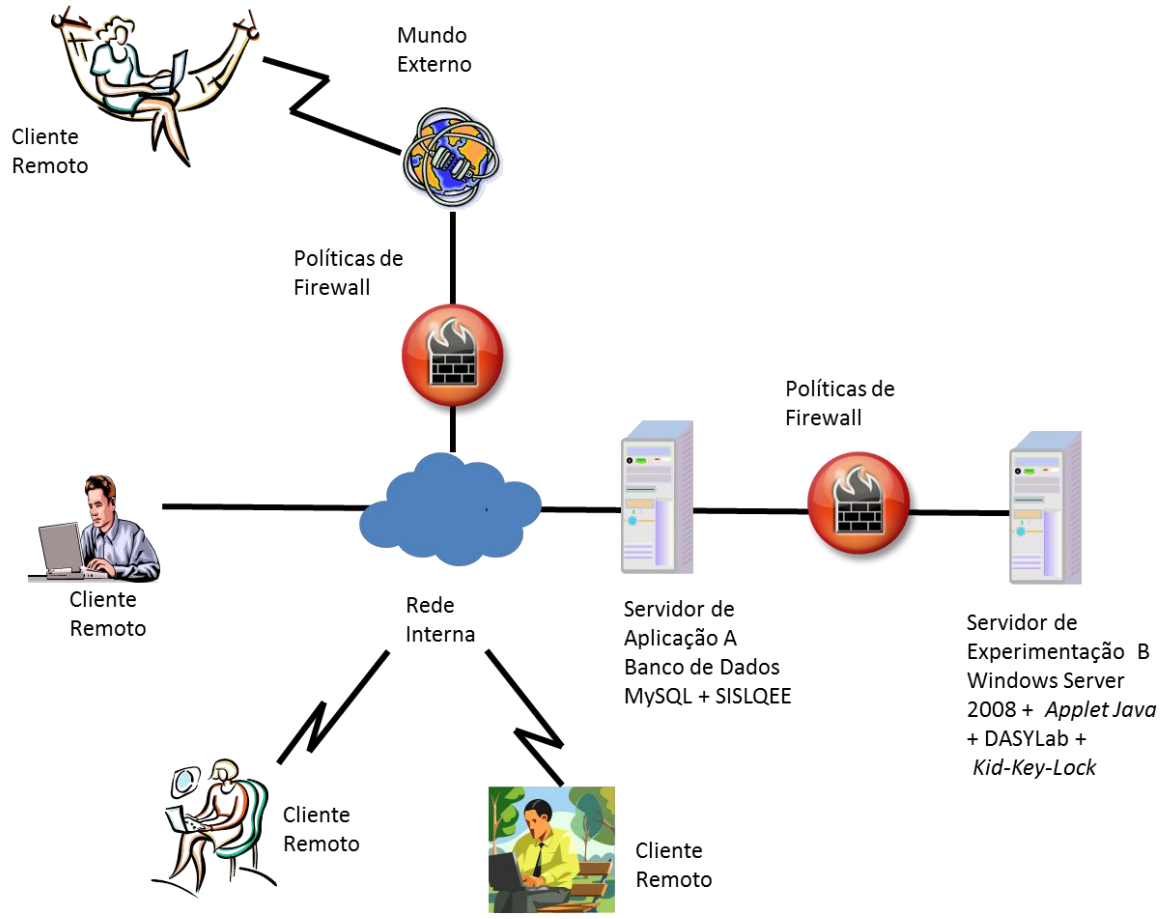
		- configurações de segurança no Sistema Operacional dos servidores A e B; - junção JAVA + VNC, permitindo o acesso remoto seguro; - Implantação do SiSLQEE; - realização de testes. - uso do SiSLQEE definitivo.
Projeto Simples	XP	O sistema foi desenvolvido, para atender as necessidades dos clientes da forma mais simples possível, com qualidade e segurança.
Teste	XP	Validação do SiSLQEE durante todo processo de desenvolvimento, utilizando testes disponibilizados pela a ferramenta de desenvolvimento.
Cliente Presente	XP	Um contato constante com o usuário final.
Código Padrão	XP	Padronização da codificação do sistema.
Orientação a objetos	*Scrum*	Foram criadas classes, separadas em camadas dentro do projeto, com a possibilidade de outro desenvolver entender o código e fazer alterações caso necessário.
Retrospectiva do ciclo	*Scrum*	Reunião para avaliar aspectos positivos e negativos, realizadas com os membros da equipe (LQEE + desenvolvedor).
Atualização do Estoque Produto	*Scrum*	De acordo com os resultados das reuniões, são reavaliadas as prioridades da Lista e atualizadas, e isso é feito de acordo com necessidade do cliente.

Fonte: Elaboração do próprio autor.

Esta terceira etapa consiste na integração das etapas anteriores e no desenvolvimento de um software que seja capaz de gerenciar usuários e experimentos, cadastrando usuários, definindo perfis, controlando agendamento dos experimentos e uso do sistema. Este software foi intitulado SiSLQEE e consiste em uma aplicação Web, que possui recursos da tecnologia VNC para o acesso remoto ao servidor de aplicações/experimentos.

Considerando as prioridades de garantir a segurança do servidor de experimentação (B) com a aplicação DASYLab, políticas de *firewall, VPN[Virtual Private Network: Rede Privada Virtual]* foram configuradas em que o acesso a este servidor somente será autorizado via servidor de aplicação (A). O servidor de aplicação fará a interface com o usuário (cliente), por meio do sistema SISLQEE, o qual tem diversas funcionalidades, como cadastro de usuários, cadastro e agendamento de experimentos e administração do ambiente como um todo. A Figura 19 abaixo ilustra este cenário.

Figura 19 – Arquitetura Inicial – terceira etapa



Fonte: elaboração do próprio autor.

Para garantir que arquitetura inicial esteja de acordo com a análise de requisitos e a extração das informações após aplicar as práticas XP e *Scrum*, foram aplicadas as técnicas de Engenharia de Software, conforme detalhamento de cada etapa do processo de desenvolvimento.

5.2 Análise de Requisitos e Projeto do “SiSLQEE".

De acordo com os conceitos e técnicas estudadas, neste trabalho foi estruturada uma Tabela com os requisitos principais do sistema, aplicando a técnica de levantamento de requisitos utilizado no *Extreme Programming*, fichas que foram preenchidas nas reuniões realizadas com os interessados do projeto, identificando as prioridades do projeto, determinando as tarefas e iterações entregas, trabalhando em paralelo as práticas XP, análise de requisitos + desenvolvimentos + testes + entregas frequentes e respectivos diagramas UML [*Unified Modeling Language*: *Linguagem Unificada de Modelagem*], proporcionando assim um melhor entendimento ao cliente, podendo ele ter uma visão geral do sistema final e uma orientação ao desenvolvedor para seguir no processo de desenvolvimento do sistema. A Tabela 7 ilustra as prioridades e as descrições de cada uma para o desenvolvimento do sistema:

Tabela 6 – Descrições dos requisitos e prioridades a serem desenvolvidas.

Descrições dos requisitos e suas prioridades a serem desenvolvidas para o sistema	
Prioridade	**Descrição**
Cadastrar Usuário	O usuário acessa o sistema via internet e se cadastra, informando dados solicitados, o administrador acessa o sistema, depois de aviso recebido por e-mail, verifica os dados e confirma o cadastro.
Acessar Sistema	Depois de autorizado o cadastro, pelo administrador, o usuário pode fazer o acesso ao sistema informando usuário e senha já cadastrados e autorizados.
Agendar Horários	Após acessar o sistema, o usuário poderá visualizar a tela de agendamento de horários, e pode verificar as datas e horários disponíveis e fazer o agendamento para acessar o software de experimentos.
Realizar Experimentos	Para realizar o experimento o usuário acessa o sistema, e poderá acessar o software experimentos, de acordo com o horário agendado, tendo também acesso a opção de somente visualizar o experimento.
Emitir Relatórios	Para emitir relatórios, o administrador acessa o sistema e tem opção de imprimir e visualizar relatórios do sistema.

Fonte: Elaboração do próprio autor.

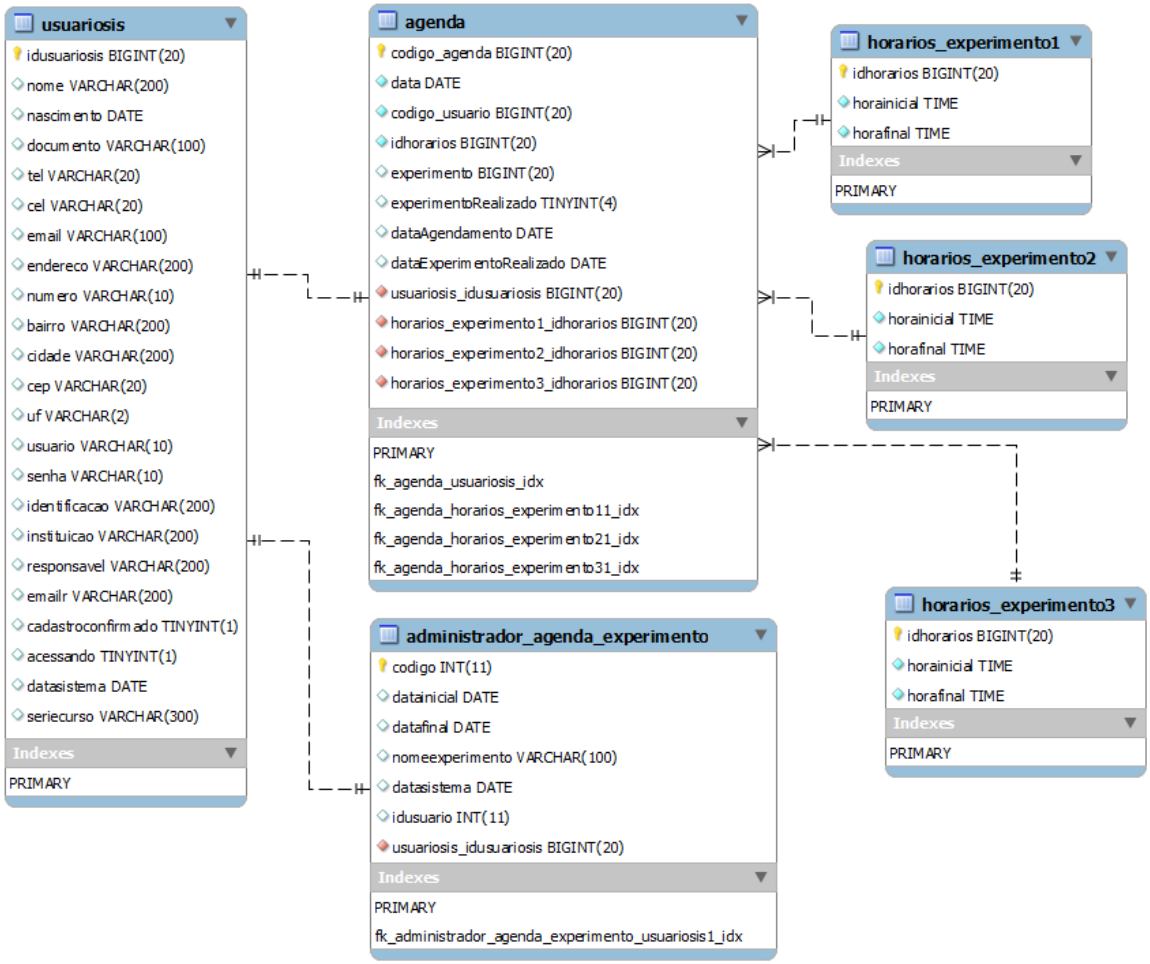
5.2.1 Diagramas da UML

Os diagramas UML demostram o funcionamento do sistema em forma de figuras e escritas para melhor entendimento.

5.2.1.1 Diagrama de Classe

Esse diagrama tem o objetivo de organizar e facilitar o entendimento das Tabelas que serão utilizadas no banco de dados. As classes são as Tabelas que serão construídas que são compostas por um conjunto de atributos que são os dados que são necessários para o funcionamento do sistema. Veja na Figura 20 o diagrama de classe do SiSLQEE.

Figura 20 - Diagrama de Classe SiSLQEE



Fonte: Elaboração do autor.

Após identificar os requisitos principais e definir as etapas do processo de desenvolvimento, iniciou-se o processo de desenvolvimento da aplicação web, e também das alterações no código-fonte para testes iniciais para a junção das tecnologias

JAVA+VNC. Estas alterações são apresentadas, a seguir, na definição da tecnologia ThinVNC bem como os critérios de sua escolha, e o exemplo da alteração do código-fonte da mesma.

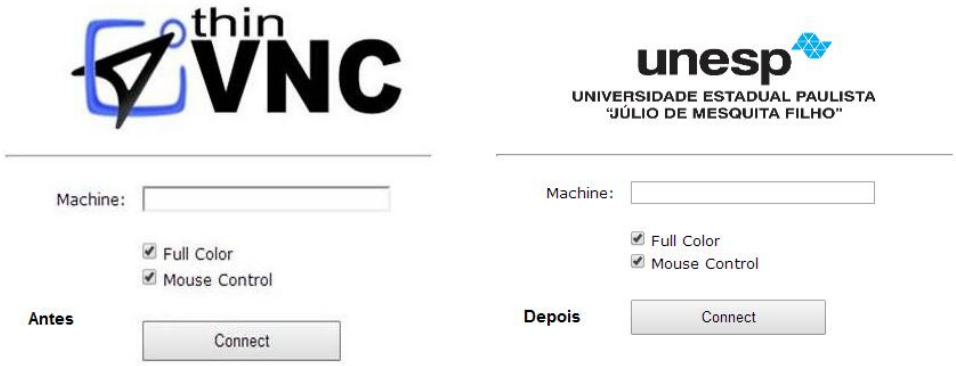
5.2.2 Tecnologia ThinVNC – Controle Remoto de Computadores

O ThinVNC é um software que permite que usuários acessem um computador remotamente, por meio do navegador. Para que isto aconteça, é definido o endereço IP da máquina remota, em que o mesmo será ativado, permitindo que o usuário remoto acesse a máquina. Este acesso poderá ser feito através de qualquer dispositivo (computadores *desktop*, *notebooks*, *smartphones*, etc) com acesso a Internet via navegador. O ThinVNC não é um VNC tradicional, pois usa padrões da Web atual: AJAX, JSON e HTML5, ligando em HTTP.

Para acessar o experimento foi desenvolvido um código em JAVA, que faz a comunicação direta ao ThinVNC. Juntamente com as configurações de segurança aplicadas nos servidores de aplicação e dados é permitido ao usuário cadastrado acessar o computador que realiza o experimento. Assim, o SiSLQEE controla usuário de forma integrada com a tecnologia ThinVNC. Para isto, algumas alterações no código fonte do ThinVNC foram realizadas.

Na Figura 21 podemos visualizar algumas modificações de tela por meio do código do ThinVNC.

Figura 21 – Troca de nomes da tela inicial



Fonte: ThinVNC (2013)

No período de testes encontrou-se, também, a necessidade de desabilitar a barra de ferramentas do ThinVNC, por questões de segurança pois com a barra o

usuário teria a acesso a tela do ThinVNC padrão dando possibilidade de fazer configurações não permitidas.

As alterações realizadas no código do ThinVNC dá garantias de segurança a aplicação DASYLab, como seria com a *Applet* Java.

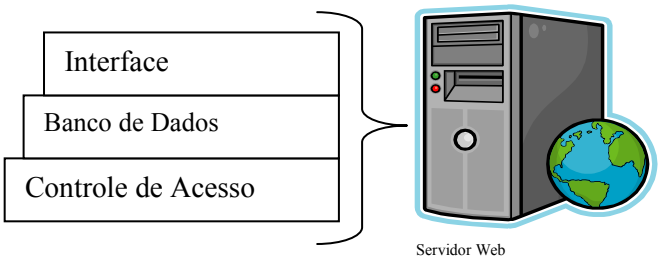
5.3 Desenvolvimento do SiSLQEE

O processo de desenvolvimento e configuração do software foi divido em três camadas, instaladas e configuradas no servidor de dados e web:

1. Interface: SiSLQEE – página principal;
2. Banco de Dados – o MySQL, em que estão os dados previamente cadastrados e onde ficaram as novas informações inseridas no sistema;
3. Controle de Acesso – JAVA + ThinVNC, que faz autenticação do usuário e permite o acesso ao servidor de aplicação em que fica o DASYLab, o software de realização as experimentações.

Esta arquitetura em camadas pode ser visualizada na Figura 22.

Figura 22 - Camadas do Sistema



Fonte: Elaboração do próprio autor.

A aplicação foi hospedada pelo servidor de aplicação Web (A) e é responsável pelo controle e gerenciamento de usuários e experimentos.

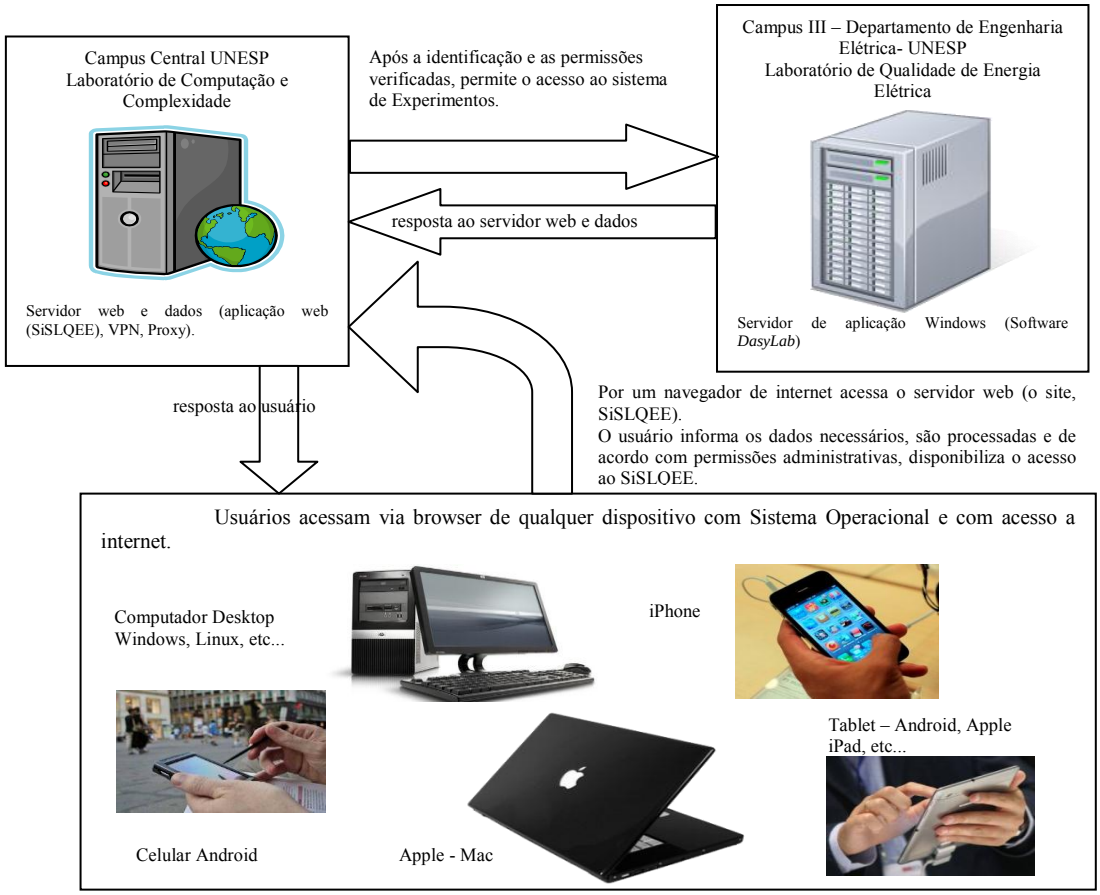
Nesta solução utilizou-se um conjunto quatro softwares para obter o acesso desejado, a fim de atender os requisitos desejados pelo LQEE. Assim, foram instalados nos servidores, de aplicação e de experimentos, o Sistema Operacional Windows Server 2008 R2, e feitas as configurações necessárias para a liberação da aplicação Web, como a instalação do Sistema Gerenciador de Banco de Dados o MySQL, a implantação da aplicação o SiSLQEE e configurações de acesso a rede.

No servidor de experimentação, foi instalado o mesmo Sistema Operacional para uso do programa DASYLab que é exclusivo plataforma Windows. Além disso, foi

feita a instalação do software ThinVNC com as alterações feitas no código fonte, para atender os requisitos do sistema. Considerando que quando o usuário acessar a tela do experimento ele poderá ter acesso à área de trabalho do Windows, então para evitar que esses usuários utilizem além do que é permitido, foram aplicadas as configurações estudadas, no item 2.4.1 *Active Directory*. Criou-se um usuário "LQEEAluno" que somente pode visualizar a área de trabalho, caso ele tente algum outro acesso não será permitido, com isso aplica-se os conceitos de segurança de dados e sistemas.

O usuário aluno/pesquisador acessa o SiSLQEE via URL, fazendo a **autenticação**, por meio usuário e senha, entra na página de usuários, sendo autorizado a editar seu cadastro, agendar horários, realizar experimentos e assistir experimentos. Já para o usuário administrador, ele tem além dessas funcionalidades citadas acima, as de controlar o acesso ao ambiente, definir o tipo do experimento, edição de listas de usuários cadastrados, listas de usuários agendados para experimento, bem como, opção de excluir e confirmar cadastro do usuário. Essa descrição está ilustrada na Figura 23.

Figura 23 - Acesso ao Sistema SiSLQEE e ao Software DASYLab



Fonte: Elaboração do próprio autor.

5.3.1 Testes de Usabilidade

O SiSLQEE foi testando, por etapas: a primeira foi feita no Cadastro de Usuários para verificar se os dados estavam sendo inseridos corretamente, e analisar dificuldades dos usuários, para isso foi feito um grupo na rede social Facebook solicitando que os integrantes do grupo realizassem o teste no sistema, em que 15 usuários participaram dessa primeira etapa. Também foi realizado um teste e avaliado via questionário por 12 alunos dos cursos de engenharias da UNESP de Ilha Solteira. Eles foram orientados a acessar o link da página, realizar o cadastro e descrever as dificuldades encontradas, e opiniões sobre o que poderia melhorar, de acordo com as opiniões. Foram corrigidas as dificuldades encontradas, tais como, o recebimento do e-mail de confirmação, estava confuso no primeiro momento. Por isso encontrou-se a necessidade de melhorar essa informação ao usuário, então na versão atual, o usuário recebe um e-mail assim que termina o cadastro, com a informação para o mesmo aguardar a confirmação de dados pelo administrador, o administrador recebe um e-mail, solicitado sua entrada no sistema e verificar e confirmar dados para que o usuário seja autorizado a acessar o sistema, assim que isso é realizado o usuário recebe um e-mail dizendo que pode acessar o sistema com login e senhas cadastradas.

Na segunda etapa foi realiza testes por 10 usuários. Eles se cadastram no sistema, após a etapa de correção, da primeira etapa depois do procedimento de cadastro e confirmação de dados os 10 usuários acessaram o sistema e se agendaram para realizar os experimentos. Foi constatado por eles que o sistema somente permite a realização de experimentos por usuários cadastrados e agendados para o determinado horário. Tendo os demais que tentarem acessar o sistema fora do seu horário aguardar para que um horário fora do seu esteja disponível.

Encontrou-se se a necessidade de inserir no sistema relatórios para o administrador verificar experimentos ativos que são divididos em quatro tipos, e relatório para visualizar os usuários cadastrados no sistema. Também foi inserida uma página para recuperação de senha, caso o usuário esquece sua senha.

No decorrer do processo de testes, foram encontradas vulnerabilidades, proporcionados chances de ataques ou que usuário acessassem informações não permitidas no servidor de experimento. Então, além das configurações das diretivas de segurança do Windows Server 2008, encontrou-se a necessidade de usar além da ferramenta de bloqueio de mouse e teclado, outra ferramenta que bloqueia a barra de

tarefas o iniciar do Windows, bloqueias a possibilidade de utilizar outros programas instalados no servidor, deixando somente visível a tela do Sistema de Experimentos DASYLab, com isso proporcionando mais segurança no acesso remoto. A figura 24 ilustra a solicitação dos testes de usabilidade do SiSLQEE.

Figura 24 – Tela da Rede Social Facebook, solicitando os testes no SiSLQEE

Fonte: Elaboração do próprio autor.

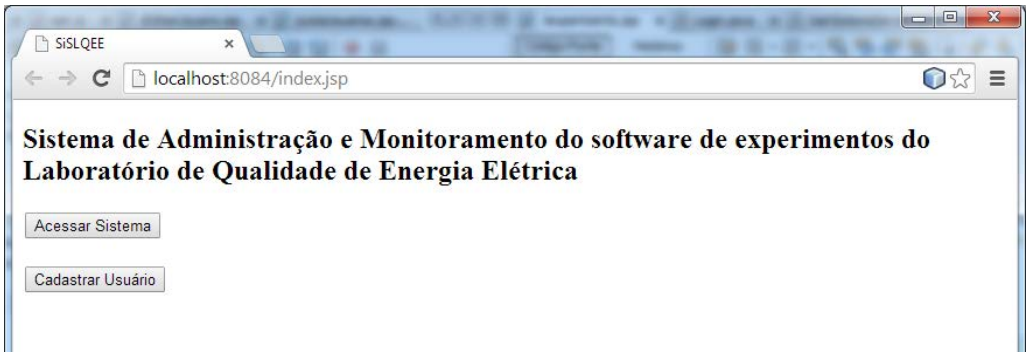
O SiSLQEE, ficou funcionado após estes testes e correções em um Servidor Web, no Campus Central do Laboratório de Computação e Complexidade de Ilha Solteira, com o MySQL Sistema Gerenciador de Bando de Dados, e Aplicação Web Java que faz o contato via código Java ao Sistema de Experimento DASYLab por meio da tecnologia VNC, permitido o acesso via internet em tempo Real, com as configurações nas Diretivas de Segurança do Windows, os Recursos do *Active Directory* e duas ferramentas da comunidade externa, tais como os programas que bloqueiam mouse, teclado e inativa a barra de tarefas do Windows. Isso ficou configurando no Servidor de Experimentos no Laboratório de Qualidade de Energia Elétrica segundo piso do Campus III do Departamento de Engenharia Elétrica de Ilha Solteira, que recebe tal acesso. Segue no próximo subitem as telas do SiSLQEE e as descrições das mesma.

5.3.2 Telas do SiSLQEE

Nessa sessão serão exibidas as imagens de cada página do SiSLQEE, e a respectiva descrição de uso de cada página, pelos usuários do sistema.

A página principal é a tela de apresentação do início do sistema/site, que mostra o acesso ao sistema/login e o cadastro de usuários. Se o usuário já está cadastrado, poderá fazer o login e entrar no sistema e acessar as demais funcionalidades. Caso contrário o usuário não cadastrado pode acessar o cadastro e se cadastrar. Este poderá acessar o sistema depois que o seu cadastro for confirmado pelo administrador do SiSLQEE. Conforme ilustrado na Figura 25.

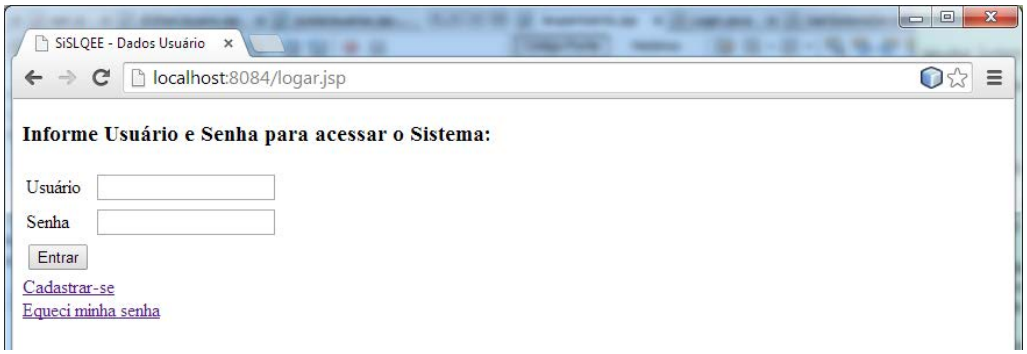
Figura 25 - Tela Principal do SiSQLEE.



Fonte: Elaborado pelo próprio autor

A tela de acesso ao sistema é lugar em que o usuário cadastrado e com o cadastro confirmado, informa seus dados que foram cadastrados no cadastro do usuário, Usuário/Senha, e se estiverem corretos terá acesso às demais funcionalidades do sistema. Ilustrado na Figura 26:

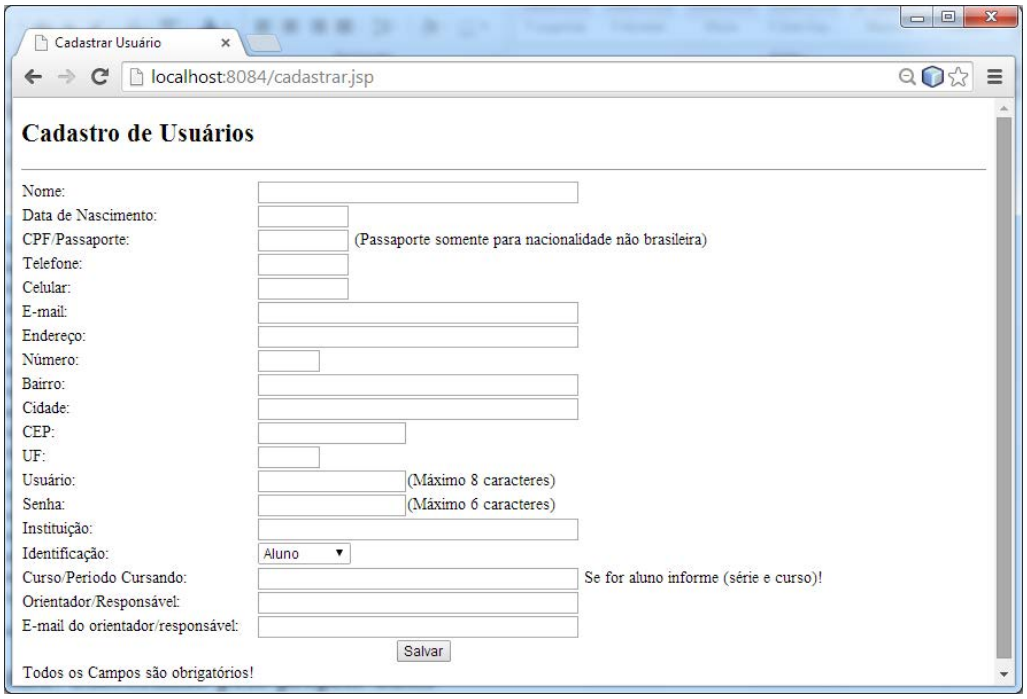
Figura 26 - Tela de Acessar o Sistema – Fazer Login.



Fonte: Elaborando pelo próprio autor

A Figura 27 ilustra a tela com o formulário de cadastro de usuários, pode ser acesso por usuário administrador/aluno, em que o usuário deverá informar os dados solicitados, e clicar no botão salvar, após salvar é enviado um e-mail ao administrador avisando que é necessário conferir e confirmar o cadastrado do usuário. Após esse procedimento, o usuário pode acessar o sistema.

Figura 27 - Tela de Cadastrar usuários do Sistema.

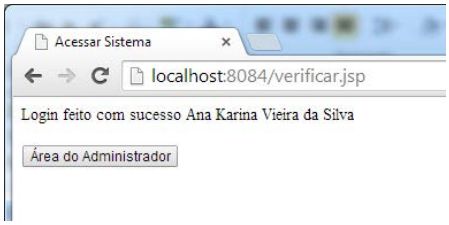


Fonte: Elaborando pelo próprio autor

Quando o cadastro é realizado com sucesso, uma janela com a mensagem é apresentada ao usuário, e este receberá um e-mail, com uma mensagem para aguardar a confirmação do cadastro.

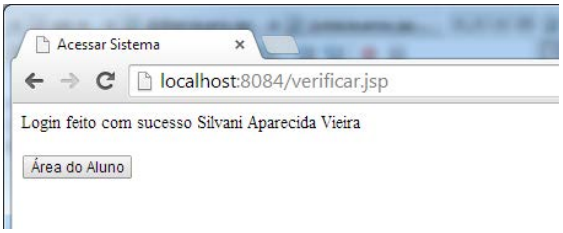
As Figuras 28 e 29, mostram as páginas que confirmam o acesso do usuário e do administrador no SiSLQEE.

Figura 28 - Mensagem de Login/Acesso a área do Administrador



Fonte: Elaborando pelo próprio autor

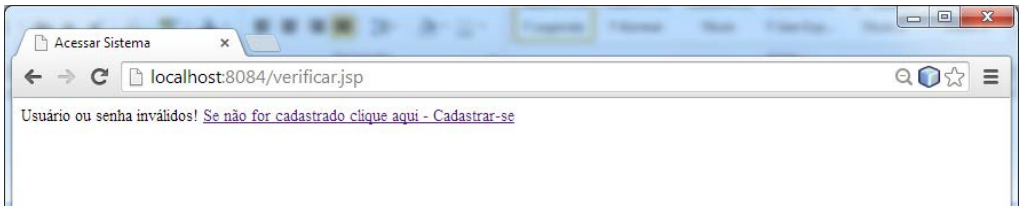
Figura 29 - Mensagem de Login/Acesso a área do Aluno



Fonte: Elaborando pelo próprio autor

A Figura 30 mostra a página que mostrar a mensagem aos usuários, quando estes informam dados errados ou não está cadastrado no sistema, dando opção para se cadastrar.

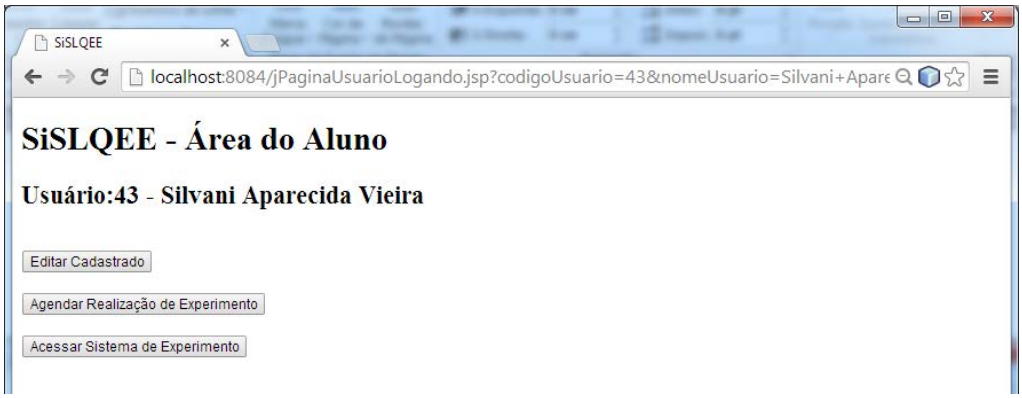
Figura 30 – Mensagem se o usuário não for cadastro ou dados errados



Fonte: Elaborando pelo próprio autor

A Figura 31, mostra a página em que o usuário aluno tem acesso ao SiSLQEE.

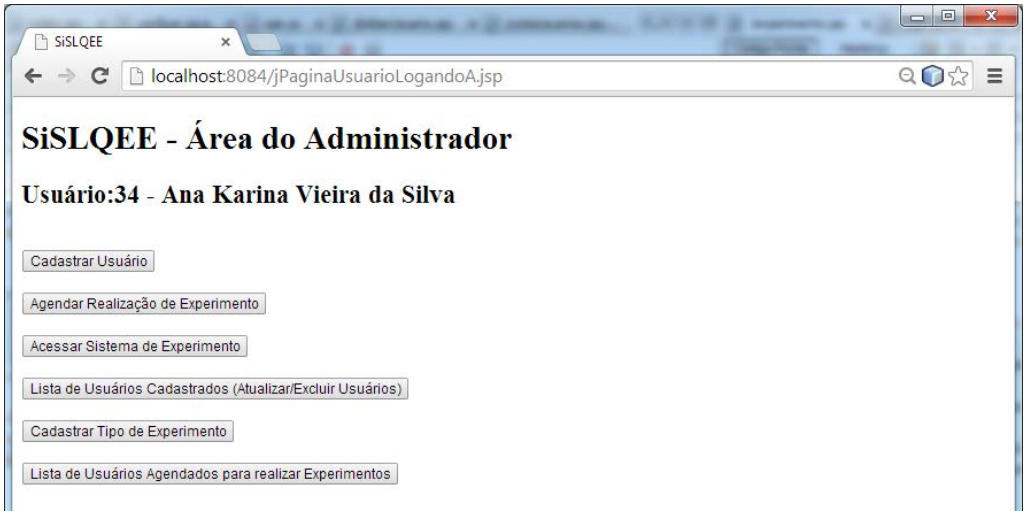
Figura 31 – Área do Aluno



Fonte: Elaborando pelo próprio autor

A Figura 32 mostra a página na qual o usuário administrador tem acesso ao SiSLQEE.

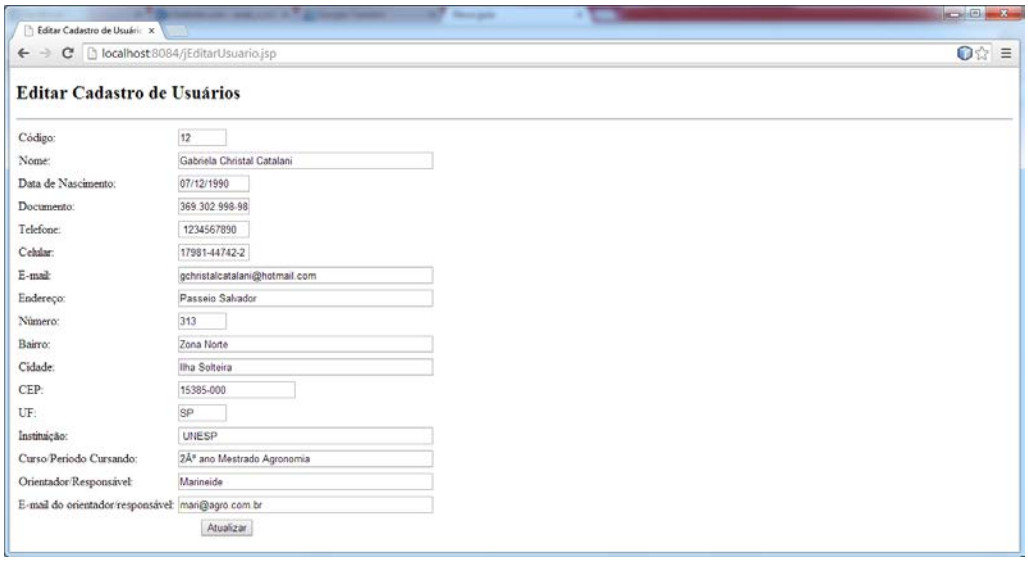
Figura 32 – Área do Administrador



Fonte: Elaborando pelo próprio autor

A Figura 33 mostra a tela que o usuário administrador/aluno pode fazer as atualizações dos dados cadastrados.

Figura 33 – Editar Cadastro de Usuários

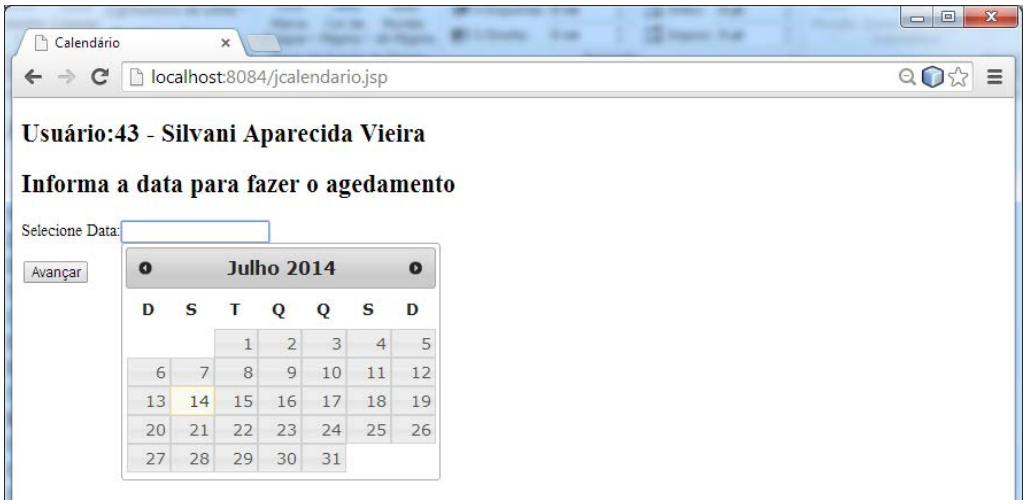


Fonte: Elaborando pelo próprio autor

Após realizar a edição nos dados cadastrados, uma mensagem de edição com sucesso é apresentada ao usuário.

A Figura 34 ilustra a página que mostra o calendário para realizar o agendamento para fazer o experimento.

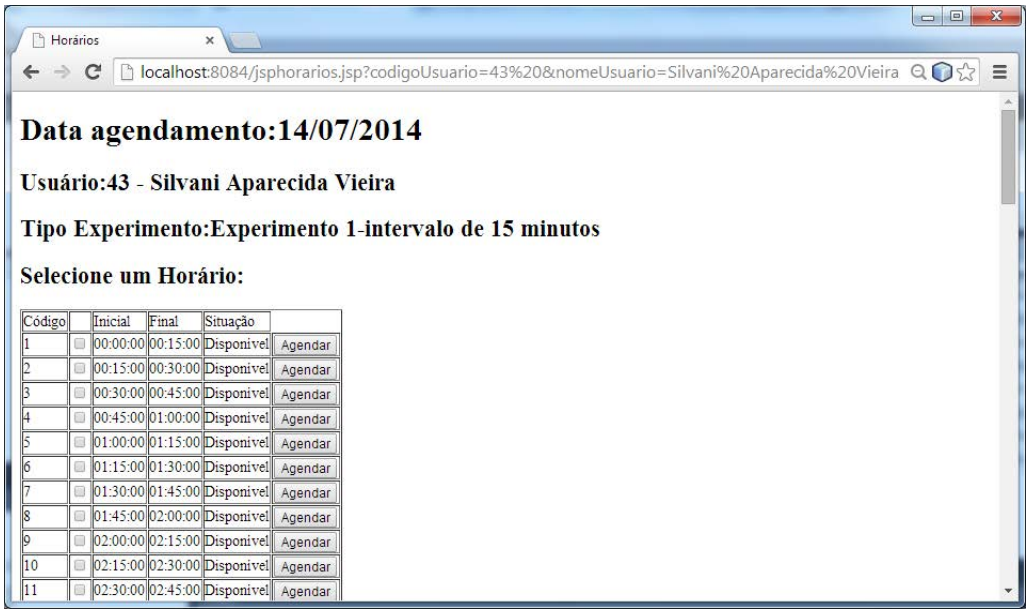
Figura 34 – Escolher data para agendar a Realização de Experimento



Fonte: Elaborando pelo próprio autor

A Figura 35 mostra a página que exibe os horários para agendar a realização do experimento.

Figura 35 – Escolher horário para agendar a realização do experimento

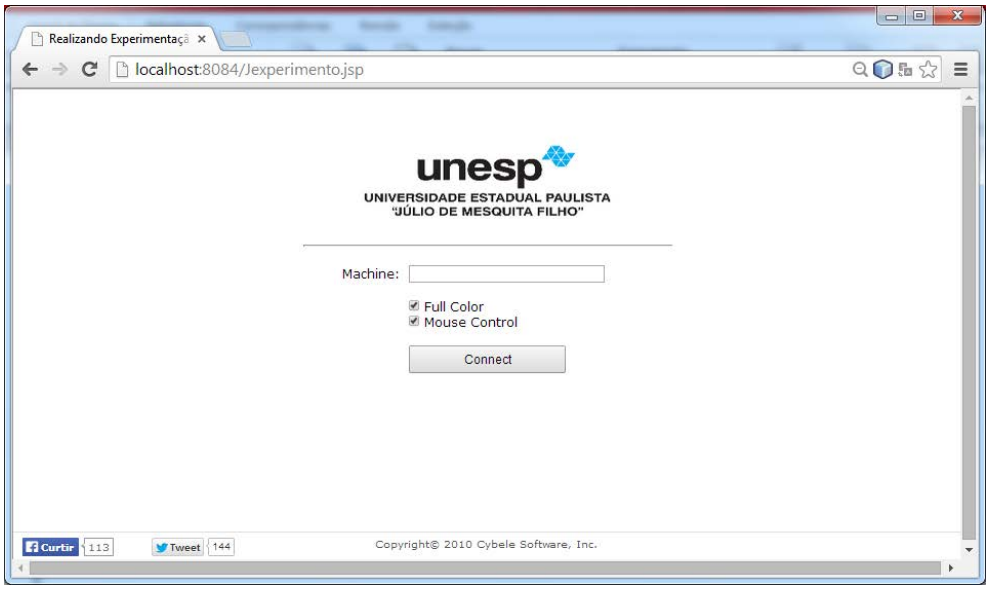


Fonte: Elaborando pelo próprio autor

Da mesma é emitida uma mensagem para o usuário que o agendamento foi realizado com sucesso.

A Figura 36 mostra a página em que é feita a conexão com o sistema de experimento o usuário tem que clic r no otão "Connect" p r d r continuid de o acesso.

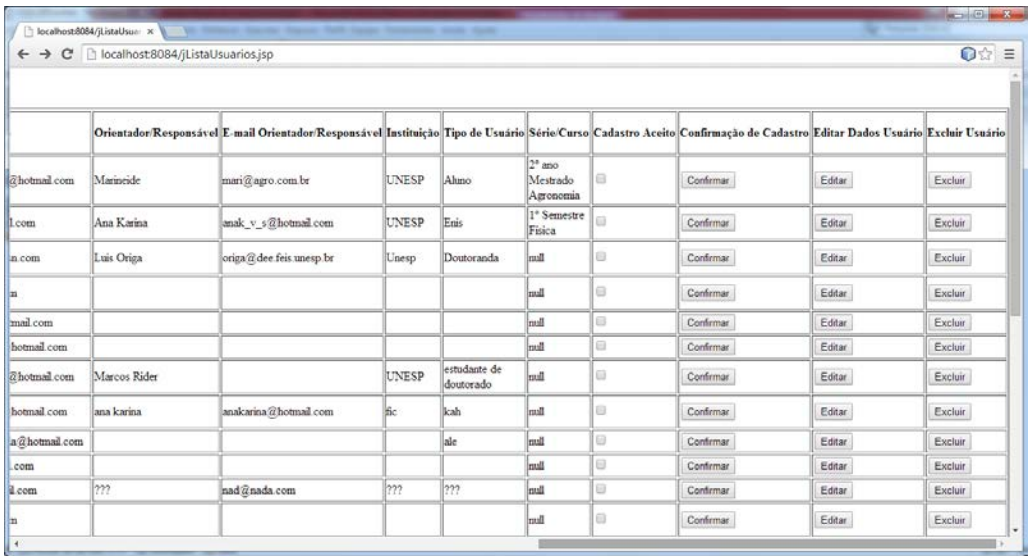
Figura 36 – Página que direciona o usuário para realização do experimento.



Fonte: Elaborando pelo próprio autor

A Figura 37 mostra a página na qual o usuário administrador tem acesso à lista que exi i d um os usuários c d str do no sistem e tem s opções e “confirm r c d stro” “edit r c d stro” e “excluir c d stro”.

Figura 37 - Lista de Usuários Cadastrados.



Fonte: Elaborando pelo próprio autor

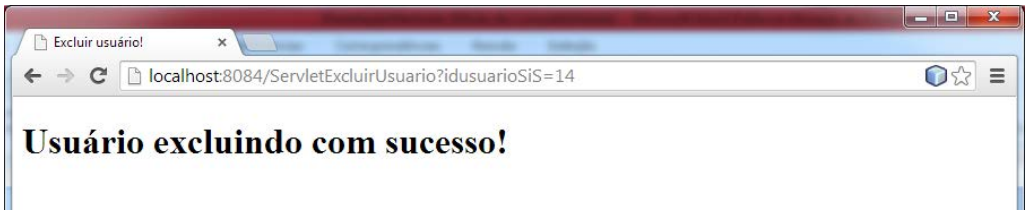
A Figura 38 mostra a mensagem que confirma o cadastro do usuário, após o administrador checar os dados e fazer essa confirmação, o usuário aluno recebe um e-mail com a confirmação de seu cadastro.

Figura 38 – Mensagem de confirmação e cadastro

Fonte: Elaborada pelo próprio autor.

A Figura 39 ilustra a página em que, após o usuário administrador clicar no botão excluir, aparece a mensagem que o usuário que ele selecionou na lista e clicou no botão excluir foi removido do sistema.

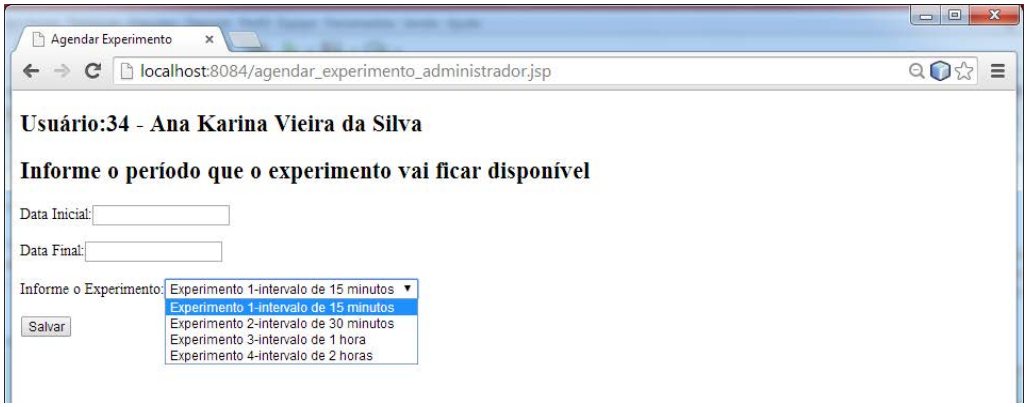
Figura 39 – Exclusão de usuário com sucesso.



Fonte: Elaborando pelo próprio autor

A Figura 40 ilustra a página em que o usuário administrador, seleciona o tipo do experimento e data de inicio e fim que o experimento vai ficar ativo, após ativar experimento, que é liberando os horários aos usuários para a realização do experimento.

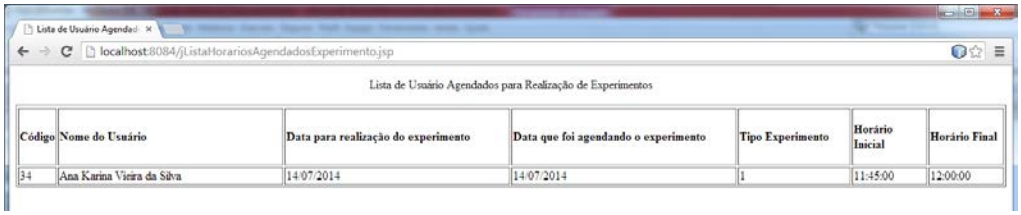
Figura 40 - Página de Agendar tipo de Experimentos e data.



Fonte: Elaborando pelo próprio autor

A Figura 41 ilustra a página que exibe a lista de usuários agendados para a realização de experimentos.

Figura 41 - Tela de Acessar o programa de realizar experimento.



Fonte: Elaborando pelo próprio autor.

A Figura 42 ilustra a página que exibe a tela para recuperação de senha, correspondente ao e-mail, cadastrado e verificando.

Figura 42 – Tela de recuperação de e-mail

Fonte: Elaborando pelo próprio autor.

A Figura 43 ilustra a mensagem ao usuário após informar o e-mail, para recuperação de senha.

Figura 43 – Tela de mensagem ao usuário de recuperação de senha

Fonte: Elaborando pelo próprio autor.

5.4 Resultados e Discussão

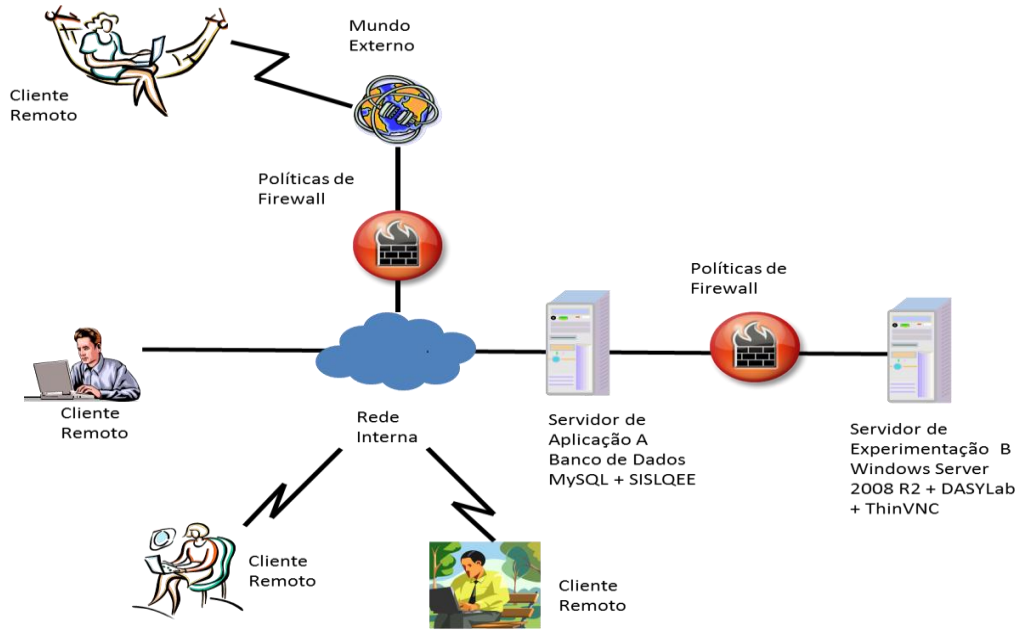
O SiSLQEE foi desenvolvido para atender a uma demanda do LQEE, permitindo acesso seguro a experimentação remota em tempo real. Esta solução habilita acesso a usuários, de qualquer lugar e de qualquer dispositivo com acesso a Internet. Permite aos usuários se cadastrarem no sistema e agendarem seus experimentos e executá-los com segurança, sem impor riscos aos equipamentos e integrantes do laboratório.

Esta solução permite que o administrador do sistema tenha um controle mais efetivo sobre os usuários que o acessam, autorizando-os ou não sob políticas administrativas, não havendo mais necessidade da interveniência operacional humana no tempo de execução dos experimentos.

O administrador recebe periodicamente mensagens de e-mail com informações relevantes, do uso e necessidades de uso do ambiente pelos usuários.

A Figura 44 consiste na Arquitetura Final do projeto.

Figura 44 – Arquitetura Final



Fonte: Elaboração do próprio autor

Esta solução de acesso seguro aos experimentos em tempo real atende as necessidades iniciais do LQEE e melhorias devem ser incorporadas, no que tange interface gráfica, usabilidade e acessibilidade, bem como, poderá ser integrada ao sistema de ensino a distância ou acadêmico da instituição, tendo assim, incorporados funcionalidades de gestão acadêmica destes sistemas.

A solução foi disponibilizada aos alunos de graduação e pós-graduação do Curso de Engenharia Elétrica da Faculdade de Engenharia de Ilha Solteira – UNESP. Estes alunos participaram de cada etapa de teste e na sequência responderam questionários, apresentado as dificuldades encontradas e os benefícios de se ter disponível uma solução como esta. Com isto, correções adicionais foram realizadas, bem como, melhorarias no sistema como um todo.

6 CONSIDERAÇÕES FINAIS

O presente trabalho tem por objetivo investigar e aplicar soluções para o acesso remoto seguro de aplicações, bem como, implementar um sistema para gerenciar e controlar o uso do ambiente e das ferramentas de experimentação remota do Laboratório de Qualidade de Energia Elétrica da FEIS/UNESP, utilizando mecanismos de segurança de sistemas, redes e dados.

Para o desenvolvimento deste, diversas soluções foram investigadas, as quais foram detalhadas no decorrer desse trabalho, destacando o estudo da Engenharia de Software e a escolha para o uso das Metodologias Ágeis, para o desenvolvimento do "SiSLQEE" [istem de Administr ção e Monitor mento de Experimentos do Laboratório de Qualidade de Energia Elétrica] utilizado para controlar o acesso ao sistema de experimentos DASYLab, que em conjunto com outras soluções, permitiu o acesso remoto seguro ao servidor de experimentação.

A Escolha da junção das ferramentas ágeis *Scrum* e *Extreme Programming*, tornou-se viável por ser apenas um a analista desenvolver, podendo focar apenas nas práticas que poderiam agilizar o processo de desenvolvimento do SiSLQEE, as práticas escolhidas foram bem destacadas e trabalhando no decorrer do processo de desenvolvimento do SiSLQEE. Embora foram encontradas dificuldades nesse processo de desenvolvimento, tais como a:

- Definição de ferramentas de desenvolvimento;
- Aprendizagem da linguagem Java;
- Equipamentos com limitações para implantação;
- Bloqueio do Windows Server, necessidade de uso de outras ferramentas disponíveis na comunidade;
- Alteração do código fonte do ThinVNC;
- Execução dos testes:
    - Colocar o sistema em funcionamento aos alunos (greve).

Mesmo com essas dificuldades foi possível aplicar a solução desenvolvida e deixar funcionado nos servidores de aplicação e experimento.

Como primeira contribuição deste trabalho, embora incipiente, foi possível construir uma solução para acesso remoto seguro de aplicações. Esta solução integra

técnicas de firewalls e configurações de proxies a fim de garantir segurança de dados e serviços providos no servidor do software DASYLab.

A segunda contribuição deste trabalho esteve relacionada diretamente ao desenvolvimento do SiSLQEE, que provê controle de acesso, integridade e disponibilidade, que são as garantias de segurança em tecnologia da informação conquistadas com esta solução, permitindo cadastro de usuários, agendamento de experimentos e controle operacional, com resultados bastante satisfatórios.

Além destas contribuições, outras, também relevantes, foram alcançadas, no que tange a complementação do profissional que completa este programa de Pós-Graduação. No desenvolver das atividades de pesquisa muitos conhecimentos foram adquiridos e consolidados, bem como, a experiência prática no desenvolvimento do processo de instalação e implantação de um sistema em rede.

Este trabalho aborda um tema que há muito em que se avançar, principalmente, em aspectos de desempenho e segurança. A solução proposta tem potencial de ser melhorada e ampliada, tornando-a automática e dinâmica, sem a necessidade da intervenção do administrador, além de integrada aos sistemas de ensino a distância.

REFERENCIAS

BECK, K. **Programação extrema explicada:** acolha as mudanças. Porto Alegre: Bookman, 2004.

BECK, K.; FOWLER, M. **Planning extreme programming**. New York: Addison-Wesley, 2000.

BECK, K.; HIGHSMITH J.; BEEDLE M.; BENNEKUM A.; COCKBURN A.; CUNHNINGHAM W.; GRENNING J.; HUNT A.; JEFFRIES R; KERN J.; MARICK B.; MARICK B.; MARTIN R.C.; MELLOR S.J.; SCHWABER K.; SUTHERLAND J.; THOMAS D. **Manifesto for agile software development.** [S.1: s.n], 2001. Disponível em: <http://www.agilemanifesto.org>. Acesso: 25 mar. 2013.

BORGES, A. P. **Instrumentação virtual aplicada a um laboratório com acesso pela internet**. 2002. Dissertação (Mestrado em Engenharia Elétrica) - Escola Politécnica da Universidade de São Paulo-USP, São Paulo, 2002.

BOTTENTUIT JUNIOR, J. B.; COUTINHO, C. P. Projeto e Desenvolvimento de um laboratório virtual na plataforma moodle. In: CONFERÊNCIA INTERNACIONAL DE TECNOLOGIAS DE INFORMAÇÕES E COMUNICAÇÃO NA EDUCAÇÃO, 5., 2007, Portugal. **Anais...** Portugal: Universidade do Minho, 2007.

CARVALHO, B. V.; MELLO, C. H. P. Revisão, análise e classificação de desenvolvimento de produtos ágil Scrum. In: SIMPÓSIO DE ADMINISTRAÇÃO DA PRODUÇÃO, LÓGISTICA E OPERAÇÕES INTERNAICIONAIS – SIMPOI, 12. 2009, São Paulo. **Anais...** Itajubá: Universidade Federal de Itajubá, 2009.

CENTRO DE ESTUDOS, RESPOSTAS E TRATAMENTO DE INCIDENTES DE SEGURANÇA NO BRASIL - CERT. **Cartilha de segurança para internet – criptografia.** [S.1.: s.n]. 2003. Disponível em: <http://www.cartilha.cert.br>. Acesso: 16 abr. 2014.

COCKBURN, A. Caracterização de pessoas como não-linear, componentes de primeira ordem em desenvolvimento de software. In: MULTI-CONFERÊNCIA INTERNACIONAL SOBRE SISTEMAS, CIBERNÉTICA E INFORMÁTICA, 21., 1999. Florida. **Anais...** Orlando: [s.n.], 2000. Disponível em: <http://alistair.cockburn.us/index.php/Characterizing_people_as_non-linar_first-order_components_in_software_development>. Acesso em: 12 dez. 2013.

DASYLAB. **Data acquisition system laboratory:** data acquisition, controlling, and monitoring. USA: National Instruments. 2013. Disponível em: < http://www.dasylab.com>. Acesso em: 10 jul. 2013.

DANTAS, V. F. **Uma metodologia para o desenvolvimento de aplicações Web num cenário global**. 2003. Dissertação (Mestrado em Informática)-Centro de Ciências e Tecnologia, Universidade Federal de Campina Grande, Campina Grande, 2003.

DEEMER, P.; BENEFIELD, G.; LARMAN, C.; VODDE, B. **The scrum primer**. [S.1.]: Scrum Training Institute. 2009. Disponível em http://www.ScrumTI.com. Acesso em: 20 dez. 2013.

DOF. **Easy to use, quality free software:** kid-key-lock. [S.1.]: Software Bloqueio**.** 2013. Disponível em: <http://www.100dof.com/products/pro-key-lock> Acesso em: 10 maio. 2013.

FACULDADE DE ENGENHARIA-FEIS. **Laboratório de qualidade de energia elétrica.** Ilha Solteira: UNESP, [200-]. Disponível em: <http://www.feis.unesp.br/#!/departamentos/engenharia-eletrica/pesquisas-e-projetos/lqee1668/>. Acesso em: 16 fev. 2014.

FERREIRA, F. N. F.; ARAÚJO, M. T. **Política de segurança da Informação:** guia prático para elaboração e implementação. Rio de Janeiro: Ciência Moderna, 2006.

FERREIRA, M. F. T.; ROCHA, T. S.; MARTINS, G. B.; FEITOSA, E.; SOUTO, E. Análise de vulnerabilidades em sistemas computacionais modernos: conceitos, exploits e proteções. In: SIMPÓSIO BRASILEIRO EM SEGURANÇA DA INFORMAÇÃO E DE SISTEMAS COMPUTACIONAIS- SBSEG, 12., 2012, Curitiba. **Minicursos...** [S.l.]: Instituto de Computação (IComp), Universidade Federal do Amazonas, 2012.

FREIRE, N. F. C. **Controle remoto de um PC através de uma plataforma 3G em comutação de pacotes.** 2007. Dissertação (Mestrado em Engenharia Eletrotécnica e de Computadores)- Instituo Superior Técnico, Universidade Técnica de Lisboa, Lisboa. 2007.

FONTES, E. **Políticas e normas de segurança da informação.** Rio de Janeiro. Editora Brasport, 2012.

FOWLER, M. **The new methodology.** [S.1.: s.n.] 2000. Disponível em: <www.martinfowler.com/articles/newMethodology.html>. Acesso em: 10 nov. 2013.

FOWLER, M. **Refactoring:** improving the design of existing code. Upper Saddle River: Addison-Wesley, 2000. 421 p.

HIGHSMITH, J. A. **Agile software development ecosystems**. Boston: Addison-Wesley, 2002.

HRUSHA, V.; KOCHAN, R.; KURYLYAK, Y.; OSOLINSKIY, O. Development of measurement system with remote access based on internet. In: INTERNACIONAL DE AQUISIÇÃO DE DADOS INTELIGENTES E SISTEMAS DE COMPUTAÇÃO AVANÇADOS: TECNOLOGIA E APLICAÇÕES-IDAACS, 4., 2007. **Workshop...** [S.1.]: EEE, 2007. p.126, 128.

HUNT, J. **Agile software construction**. London: Springer, 2006.

INTERNATIONAL BUSINESS MACHINES- IBM. **Autenticação.** [S.1.: s.n., 200-]. <http://publib.boulder.ibm.com/infocenter/tivihelp/v27r1/index.jsp?topic=%2Fcom.ibm.itam.doc%2Fsecurity%2Fc_security_tamit.html&lang=pt_BR>. Acesso: 16 fev. 2014.

JEFFRIES, R.; ANDERSON, A.; HENDRICKSON, C. **Extreme programming installed.** Boston: Addison-Wesley, 2001.

JURIC, R**.** Extreme programming and its development practices. In: CONFERÊNCIA INTERNACIONAL DE INTERFACES NA TECNOLOGIA DA INFORMAÇÃO, 22., 2000, London. **Anais**... London: SEI OAA, 2000.

KAZIENKO, J. F. **Assinatura digital de documentos eletrônicos através da impressão digital.** 2003. Dissertação (Mestrado em Ciência da Computação), Universidade Federal de Santa Catarina, Florianópolis. 2003.

LEE, K. W.; TRUONG, Nguyen-Vu; RHODES, B.; MCLAREN, J.; WANG, L. Development of a remote access control laboratory using xpc target and virtual reality modeling language. In: CONFERÊNCIA INTERNACIONAL DE SISTEMAS AVANÇADOS INTELIGENTES- ICIAS, 3., 2010. **Anais...** Malésia: Universiti Teknologi Petronas –UTP, 2010.

MARIN, R.; SANZ, P.J.; NEBOT, P.; WIRZ, R. A multimodal interface to control a robot arm via the web: a case study on remote programming. **Industrial Electronics**, **IEEE Transactions on,** New York, v.52, n.6, p.1506,1520, Dez. 2005. doi: 10.1109/TIE.2005.858733.

MATTIOLI, F. E. R.; LAMOUNIER JUNIOR, E. A.; CARDOSO, A.; ALVES, N. M.M. **Uma proposta para o desenvolvimento ágil de ambientes virtuais.** Uberlândia: Universidade Federal de Uberlândia - Instituto Federal do Triângulo Mineiro. 2009. Disponível em: < http://www.lbd.dcc.ufmg.br/colecoes/wrva/2009/0021.pdf>. Acesso em: 9 out. 2013.

MICROSOFT. **Microsoft windows server 2008 R2.** [S.1.:s.n], 2009. Disponível em: <http://technet.microsoft.com/pt-br/library/dd578336(v=ws.10).aspx>. Acesso: 3 abr. 2013.

MICROSOFT. **Criptografia.** [S.1.: s.n.], 2009. Disponível em: <http://windows.microsoft.com/pt-br/windows/what-is-encryption#1TC=windows-7>. Acesso em: 16 fev. 2014.

MOREIRA, R.S. **Certificados digitais**. Rio de Janeiro: Escola de Engenharia - Universidade Federal do Rio de Janeiro, 2003.

NACIONAL INSTITUTE OF STANDARDS TECHNOLOGY - NIST. **Special publication 800-53 revision 2:** recommnedation Information Security. – [S.1.: s.n], agust, 2009. Disponível em: < http://www.nist.gov/>. Acesso em: 10 jun. 2013.

NACIONAL INSTITUTE OF STANDARDS TECHNOLOGY - NIST. **Special publication 800-82:** recommnedation computer security - guide to industrial control systems(ICS) security. [S.1.: s.n], June, 2011. Disponível em: < http://www.nist.gov/>. Acesso em: 10 jun. 2013.

OLIVEIRA, J. M. C.; MELO, S. S. J.; CALAZANZ. J. R. G.; SILVA, J. B.; RODRIGUES, W. M. R.; SAMAPAIO, R. B. Desenvolvimento da Plataforma do Laboratório de Acesso Remoto e Instrumentação virtual via web. In: CONGRESSO BRASILEIRO DE EDUCAÇÃO EM ENGENHARIA-COBENGE, 2012, Belém. **Congresso...** Manaus: Instituto Federal de Educação, Ciência e Tecnologia do Amazonas – Campus Manaus Distrito Industrial – IFAM/CMDI. 2010.

OLIVEIRA, R. R. Criptografia simétrica e assimétrica: os principais algoritmos de cifragem. **Revista Segurança Digital,** Brasília, v. 2, n. 3, p. 21-24 - 2012. Disponível em: < http://www.ronielton.eti.br/blog/2012/06/02/artigo-revista-segurana-digital/>. Acesso em: 30 ago. 2013.

PAULA FILHO, W. de P. **Engenharia de software:** fundamentos, métodos e padrões. 3. ed. Rio de Janeiro: LTC, 2009.

PAULA, R. S. A. Engenharia de software: scrum na melhoria do gerenciamento de projetos de software. **Revista Engenharia de Software,** São Paulo, 2010.

PRESSMAN, R. S. **Engenharia de software**. 3. ed. São Paulo: Makron Books, 1995.

RAPANELLO, R. M. **Laboratório remoto de qualidade da energia elétrica**. 2008. Dissertação (Mestrado em Engenharia Elétrica)- Faculdade de Engenharia, Universidade Estadual Paulista- UNESP, Ilha Solteira, 2008.

RICHTER, T.; WATSON, R.; KASSAVETIS, S.; KRAFT, M.; GRUBE, P.; BOEHRINGER, D.; DE VRIES, P.; HATZIKRANIOTIS, E.; LOGOTHETIDIS, S., The WebLabs of the University of Cambridge: a study of securing remote instrumentation, remote engineering and virtual instrumentation (REV). In: INTERNATIONAL CONFERENCE ON, 9., 2012. **Conference…** [S.1.: s.n], 2012.

RISING, L.; JANOFF, N.S. The scrum software development process for small teams. **SOFTWARE,** v. 17, n. 4, p. 26,32, Jul/Aug 2000
doi: 10.1109/52.854065

RUSSEL, C.; ZACKER, C. **Microsoft:** introducing windows server 2008 R2. Washington: Microsoft Press, 2010.

SANTOS JÚNIOR, D. **Implementação de processo de software para teste de equipamentos aeroespaciais.** 2007. Dissertação (Mestrado de Engenharia Elétrica)- Escola de Engenharia de São Carlos, Universidade de São Paulo, São Carlos, 2007.

SCHWABER, K.; SUTHERLAND, J. **The scrum guide:** the definitive guide to scrum. U.S.A.: *Scrum*, 2011.

SCHWABER, K; BEEDLE, M. **Agile software development with scrum**. [S.1.]: Prentice-Hall, 2001. ISBN-13:978-0130676344.

SOARES, M. S. Metodologias Ágeis Extreme Programming e Scrum para o Desenvolvimento de Software. **Revista eletrônica de sistemas de informação,** Minas Gerais,v. 3, n.1, 2004. ISSN 1677-3071 DOI: 10.5329/RESI. Disponível em: < http://revistas.facecla.com.br/index.php/reinfo/article/view/146/38>. Acesso em: 20 ago 2013.

SOUZA, L. M. Método ágil XP (extreme programming). **Revista Eletrônica da FIA,** v.3., n.3, p.3, 2007. Disponível em: < intranet.fainam.edu.br/acesso_site/fia/academos/revista3/6.pdf>. ISSN. 1809-3604.

STRAATSMA, M.; COX, D.; CTISTIS, C.; BARTZ, R. Development and Enhancement of RLab - a remote laboratory system. In: INTERNATIONAL CONFERENCE ON SYSTEMS AND NETWORKS COMMUNICATIONS – ICSNC - , 4., 2009, Porto. **Conference…** [S.1: s.n.], 2009. p. 159-164. Doi: 10.1109/ICSNC.2009.97.

STALLINGS, W. **Criptografia e segurança de redes:** princípios e práticas. 4. ed. São Paulo: Person Education do Brasil, 2008.

SUBRAMANIAM, V.; HUNT, A. **Practices of an agile developer**. Dallas: Pragmatic Bookshelf, 2006.

TEXEIRA FILHO, N. **Estudo do impacto do uso das metodologias ágeis na melhoria do planejamento e acompanhamento do processo de ensino e aprendizagem em sala de aula.** 2012. Dissertação (Mestrado em Computação Aplicada)- Instituto Federal de Educação Ciência e Tecnologia do Ceará, Universidade Estadual do Ceará, Fortaleza, 2012.

TELES, V. M. **Extreme programming:** aprenda como encantar seus usuários desenvolvendo software com agilidade e alta qualidade. Rio de Janeiro:Novatec Novatec Editora, 2004. ISBN: 85-7522-047-0.

TELES, V. M. **Um estudo de caso da adoção das práticas e valores do extreme programming**. 2005. Dissertação (Mestrado em Informática)- Instituto do Instituto de Matemática e Núcleo de Computação Eletrônica, Universidade Federal, Rio de Janeiro: 2005.

THOMPSON, M. A. **Windows server 2008 R2:** fundamentos**.** São Paulo: Érica, 2010.

WILLIAMS, L.; KESSLER, R. **Pair programming illuminated**. Boston: Addison-Wesley, 2003. 265 p.

WILLIAMS, S. P.; HARDY, C. A.; HOLGATE, J. A. **Information security governance practices in critical infrastructure organizations:** a socio-technical and institutional logic perspective. Berlin. Electron Markets- Springer, 2013. ISSN: 1422-8890.