



UNIVERSIDADE FEDERAL DE CAMPINA GRANDE
CENTRO DE ENGENHARIA ELÉTRICA E INFORMÁTICA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

**Relatório de Estágio Supervisionado
Laboratório EMBEDDED.**

Stayner Nóbrega Barros

Campina Grande - PB

Outubro de 2021

Stayner Nóbrega Barros

Relatório de Estágio Supervisionado Laboratório EMBEDDED.

Relatório de Estágio Supervisionado submetido à Coordenação do Curso de Graduação em Engenharia Elétrica da Universidade Federal de Campina Grande como parte dos requisitos necessários para a obtenção do grau de Bacharel em Engenharia Elétrica.

Área de Concentração: Controle e Automação

Prof. Jaidilson Jó da Silva, D.Sc.

Orientador

Campina Grande - PB

Outubro de 2021

Stayner Nóbrega Barros

Relatório de Estágio Supervisionado Laboratório EMBEDDED.

Relatório de Estágio Supervisionado submetido à Coordenação do Curso de Graduação em Engenharia Elétrica da Universidade Federal de Campina Grande como parte dos requisitos necessários para a obtenção do grau de Bacharel em Engenharia Elétrica.

Aprovado em 27 / 05 / 2021

Prof. Gutemberg Gonçalves dos Santos Junior, D.Sc.
Universidade Federal de Campina Grande
Avaliador

Prof. Jaidilson Jó da Silva, D.Sc.
Universidade Federal de Campina Grande
Orientador

Campina Grande - PB
Outubro de 2021

Este trabalho é dedicado aos meus pais, Erica e Stêfenes, que sempre incentivaram o estudo, e me deram as condições para isto, e a todos os meus queridos amigos que me influenciaram em quem sou hoje.

Agradecimentos

Agradeço a Deus, pois sem Ele eu não teria conseguido chegar onde cheguei.

Agradeço a meus pais que sempre se esforçaram bastante para que eu tivesse as condições de ter a melhor educação, sempre me apoiando e me auxiliando. A toda minha família por toda a torcida, e especialmente a minha avó Socorro que me auxiliou com moradia durante toda a graduação em outra cidade.

A minha namorada Vanessa que sempre acreditou nas minhas conquistas e me apoiou em todos os momentos.

Aos amigos que ingressamos juntos no curso (Evellyne, Lizandra, João Marcos, Vicente) que me acompanharam até o fim. Aos amigos adquiridos durante a jornada (Clara, Giordano, Giovanny, Laécio, Marcus, Marianna, Mirelle, Paula, Samuel, Vagne e Valmir), por toda ajuda com coisas internas e externas ao curso, por tantos momentos de felicidade, e que mesmo com poucos anos de convivência já representam boa parte do que sou hoje. Aos meus amigos de longa data (Alessandra, Daniel, Gabrielly, Ingrid, Rafael, Rayff e Vitor) por todas as aventuras vividas.

A meu orientador Jaidilson pela disponibilidade prestada. Aos companheiros que participamos de trabalhos juntos (Arthur e Victor) pelas publicações realizadas, e a todos os professores que me auxiliaram na minha formação de alguma forma.

Epígrafe

*“Desistir de aprender é egoísmo.
Quando acalentamos o desejo de
aprender mais, nossas vidas estarão
repletas de genuína vitalidade e brilho”.*
(Daisuka Ikeda)

Lista de Ilustrações

Figura 1 – Fotografia da fachada do Laboratório.	2
Figura 2 – Diagrama da Estrutura Hierárquica da Equipe de Projeto.	3
Figura 3 – Diagrama com o ciclo de funcionamento da metodologia ágil Scrum. . .	5
Figura 4 – Representação do quadro Scrum.	5
Figura 5 – Representação do sistema Digital de TV terrestre.	8
Figura 6 – Encapsulamento de dados TCP.	11

Lista de abreviaturas e siglas

ABNT	- Associação Brasileira de Normas Técnicas
TCP	- <i>Transmission Control Protocol</i>
CEEI	- Centro de Engenharia Elétrica e informática
UFCG	- Universidade Federal de Campina Grande
SBTVD	- Sistema Brasileiro de Televisão Digital
TS	- <i>Transport Stream</i>
IP	- <i>Internet Protocol</i>
NCL	- Nested Context Language

Sumário

1	INTRODUÇÃO	1
2	O AMBIENTE DE TRABALHO	2
2.1	O Laboratório Embedded	2
2.2	Projeto de Pesquisa e Desenvolvimento	3
3	METODOLOGIA DE DESENVOLVIMENTO	4
3.1	Metodologia Ágil - Scrum	4
3.1.1	A vivência	5
4	FUNDAMENTAÇÃO TEÓRICA	7
4.1	Televisão Digital Brasileira	7
4.1.1	História	7
4.1.2	O Sistema Brasileiro de Televisão Digital (SBTVD)	8
4.2	O Middleware	9
4.2.1	Ginga	9
4.2.2	Ambientes de Programação de Aplicações	10
4.3	Normatização ABNT	10
4.4	<i>Transmission Control Protocol (TCP)</i>	11
5	ATIVIDADES DESENVOLVIDAS	13
5.1	Ferramentas utilizadas	13
5.1.1	Git	13
5.1.2	Clion	13
5.1.3	Microsoft Teams	13
5.1.4	Linguagem C++	13
5.1.5	NCLua	14
5.1.6	Wireshark	14
5.1.7	Google C++ Mocking Framework	14
5.2	Descrição das Atividades Desenvolvidas	14
5.2.1	Suporte a comunicação TCP no Ginga	15
5.2.2	Suporte a arquivos Zip no Ginga	15
5.2.3	Interpretação e interação de propriedades de elementos da tela descritos em NCLua no Ginga	16
5.2.4	Criação de Aplicações em NCLua para validação de funções do Ginga	16
5.2.5	Criação de testes de unidades em C++ para validação de funções no Ginga	16

5.2.6	Relatórios	16
6	CONCLUSÕES	18
	REFERÊNCIAS BIBLIOGRÁFICAS	19

1 Introdução

Nesse relatório são descritas as principais atividades realizadas pelo estudante Stayner Nóbrega Barros durante o estágio supervisionado no Laboratório de Sistemas Embarcados e Computação Pervasiva (Embedded), que foi realizado no período de 07 de junho a 15 de outubro de 2021, com uma carga horária de 20 horas semanais, totalizando 374 horas, sob a orientação do professor Jaidilson Jó da Silva e a supervisão do professor George Acioli Júnior.

Conforme previsto no Projeto Pedagógico do Curso de Engenharia Elétrica do Centro de Engenharia Elétrica e Informática (CEEI) da Universidade Federal de Campina Grande (UFCG) é obrigatório a elaboração, desenvolvimento, orientação, apresentação e avaliação da disciplina Estágio Integrado ou Estágio Supervisionado para os alunos concluintes. O objetivo é proporcionar ao estudante uma experiência profissional, em que seja possível conhecer e executar atividades associadas a Engenharia Elétrica, consolidando assim os conhecimentos apreendidos durante o curso.

As atividades foram realizadas no contexto de um projeto de PDI mediante uma parceria entre o Laboratório Embedded e o VIRTUS UFCG - Núcleo de Pesquisa, Desenvolvimento e Inovação em Tecnologia da Informação, Comunicação e Automação, ambos da Universidade Federal de Campina Grande (UFCG). O VIRTUS é um órgão suplementar da Universidade Federal de Campina Grande, especificamente do Centro de Engenharia Elétrica e Informática (CEEI).

A atuação do estagiário durante o período em questão incluíram a exploração das normas técnicas de software das TV digitais brasileiras, desenvolvimento de código em C++, Lua e NCL, avaliações de conexões TCP, e imersão em processos de gestão e desenvolvimento ágil de sistemas.

2 O Ambiente de Trabalho

Nesta seção serão descritos o laboratório, o qual foi realizado o estágio, de que tipo era e a estrutura da equipe.

2.1 O Laboratório Embedded

O Laboratório de Sistemas Embarcados e Computação Pervasiva (Embedded) faz parte do Centro de Engenharia Elétrica e Informática (CEEI) da Universidade Federal de Campina Grande (UFCG), em Campina Grande, Paraíba e foi fundado em dezembro de 2005. Na Figura 1 é apresentada uma fotografia da fachada do laboratório.

Figura 1 – Fotografia da fachada do Laboratório.



Fonte: embedded.ufcg.edu.br

Através da UFCG, o Laboratório Embedded é credenciado no Comitê da Área de Tecnologia de Informação (CATI) para receber recursos da Lei de Informática, tendo o Parque Tecnológico da Paraíba como interveniente financeiro também credenciado no CATI.

A equipe do Laboratório Embedded é formada por professores e pesquisadores do Centro de Engenharia Elétrica e Informática da Universidade Federal de Campina Grande. O laboratório conta com cerca de 60 colaboradores, incluindo alunos de doutorado, mestrado e graduação, dos cursos de Engenharia Elétrica e Ciência da Computação.

A missão do Embedded é avançar no estado da arte nas áreas de sistemas embarcados e computação pervasiva, promovendo ações que permitam que tais avanços

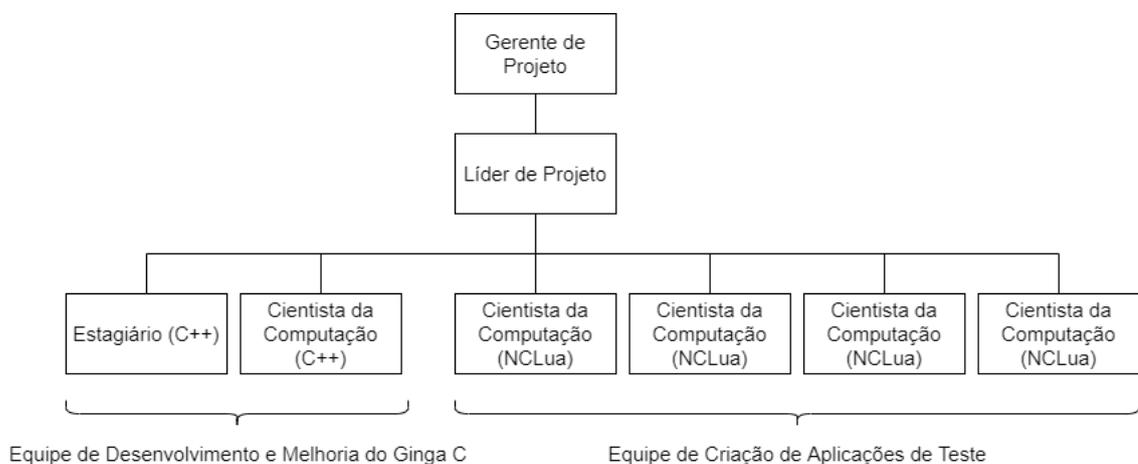
tragam benefícios para a sociedade através de parcerias com grandes empresas. Para isso, tem-se uma equipe formada por pesquisadores focados na produção de conhecimento e na aplicação deste conhecimento na resolução de problemas reais da indústria, equilibrando perspectivas acadêmicas com as necessidades de mercado.

O foco está na concepção, implementação e experimentação de soluções para sistemas embarcados e computação pervasiva, incluindo métodos, técnicas, arcabouços, ferramentas e aplicações, com o objetivo de proporcionar tecnologias invisíveis para o usuário, acessíveis a todo momento, em todo lugar e enquanto o usuário se movimenta.

2.2 Projeto de Pesquisa e Desenvolvimento

No contexto de atuação do Embedded, o estagiário realizou as atividades em um projeto de cooperação técnico e científico entre o laboratório e uma empresa parceira. Nesse projeto são executadas atividades de pesquisa e desenvolvimento de desenvolvimento e testes de software. Atualmente a equipe é composta por 8 pessoas e organizadas de acordo com a estrutura hierárquica apresentada na Figura 2.

Figura 2 – Diagrama da Estrutura Hierárquica da Equipe de Projeto.



Fonte: Autoria Própria.

As tarefas e decisões do direcionamento do projeto eram elaboradas pelo líder e repassadas aos outros membros. Diariamente ocorriam reuniões com todos os membros da equipe para atualização e repasse de novas tarefas, foi utilizado a metodologia ágil Scrum durante todo o projeto pelos membros, esta metodologia será descrita no próximo capítulo.

3 Metodologia de Desenvolvimento

Esta seção descreve as etapas e papéis envolvidos na metodologia utilizada para a realização das atividades de estágio.

3.1 Metodologia Ágil - Scrum

No ambiente de desenvolvimento do projeto, as atividades foram desenvolvidas utilizando a metodologia Scrum.

Scrum é uma metodologia ágil para gestão e planejamento de projetos de software. No Scrum, os projetos são divididos em ciclos chamados de Sprints. O Sprint representa um Time Box dentro do qual um conjunto de atividades deve ser executado. Metodologias ágeis de desenvolvimento de software são iterativas, ou seja, o trabalho é dividido em iterações, que são chamadas de Sprints no caso do Scrum. O tempo de Sprint no Embedded utilizado foi de duas semanas.

A unidade fundamental do Scrum é o Scrum Team. Ele é composto por Developers, um Scrum Master e um Product Owner. Dentro de um time, não há sub-times ou hierarquias. Trata-se de uma unidade coesa de profissionais focados no objetivo atual do produto (SCHWABER; SUTHERLAND, 2020).

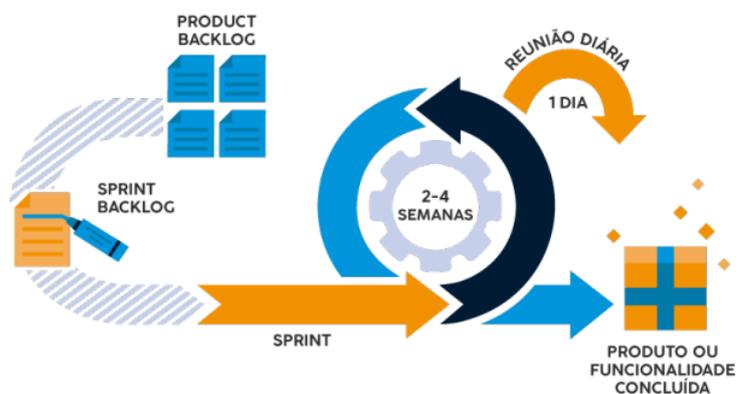
Os Developers são responsáveis pela criação de qualquer aspecto incremental da solução ao longo da Sprint. No geral, as atividades dessa parcela do time estão associadas à criação de um plano para atender as demandas e à adaptação diária deste em direção ao objetivo da Sprint. O Scrum Master é responsável por garantir a execução do método ágil seguindo as diretrizes Scrum. Ele lidera o time e o conduz, garantindo que haja auto-gerenciamiento, entregas parciais com incremento de valor dentro dos prazos estimados e auxilia na resolução de possíveis impedimentos. Por fim, o Product Owner é responsável por maximizar o valor do produto resultante do trabalho do Scrum Team. Define e explicita o objetivo do produto, comunica os requisitos relacionados às entregas parciais e garante que as requisições são claras, visíveis e compreendidas por todos.

As funcionalidades a serem implementadas em um projeto são mantidas em uma lista que é conhecida como Product Backlog. No início de cada Sprint, faz-se um *Sprint Planning Meeting*, ou seja, uma reunião de planejamento na qual o Product Owner prioriza os itens do *Product Backlog* e a equipe seleciona as atividades que ela será capaz de implementar durante o Sprint que se inicia. As tarefas alocadas em um Sprint são transferidas do Product Backlog para o Sprint Backlog.

A cada dia de uma Sprint, a equipe faz uma breve reunião (normalmente de manhã),

chamada Daily Scrum. O objetivo é disseminar conhecimento sobre o que foi feito no dia anterior, identificar impedimentos e priorizar o trabalho do dia que se inicia. Ao final de um Sprint, a equipe apresenta as funcionalidades implementadas em uma Sprint Review Meeting. Finalmente, faz-se uma Sprint Retrospective e a equipe parte para o planejamento do próximo Sprint. Assim reinicia-se o ciclo (BRASIL,), apresentado na Figura 3.

Figura 3 – Diagrama com o ciclo de funcionamento da metodologia ágil Scrum.



Fonte: www.tecnicon.com.br/

Basicamente, existem três estados para descrever as tarefas propostas. *To Do* no caso das tarefas ainda não iniciadas, *Doing* para as tarefas em andamento e *Done* para as tarefas já realizadas. Estes compõem o quadro Scrum, que pode ser ilustrado na Figura 4. O quadro condensa as informações essenciais do projeto.

Figura 4 – Representação do quadro Scrum.



Fonte: (OLIVEIRA, 2021)

3.1.1 A vivência

O estagiário estava enquadrado na equipe de Developers. Especificamente, interagiu com equipes de desenvolvimento em c++, absorvendo muito da experiência prática do

time. Imerso no processo de desenvolvimento ágil, foi responsável pela investigação de problemas no código desenvolvido do Ginga, especialmente das classes de comunicação TCP e Zip.

O estagiário participou de todo o processo de Scrum já descrito, o que foi de grande importância para experiência profissional do estagiário, visto que era possível validar as próprias atividades contando estas para o resto do time durante as Daily, assim como entender o que os outros membros estavam realizando para entender o processo como um todo.

O registro contínuo de atividades foi feito utilizando a ferramenta de gestão de projetos Turmalina. Esta ferramenta possibilita aos *Scrum Masters* uma visão geral do andamento do projeto, compreendendo as etapas *To Do*, *Doing* e *Done*, além dos impedimentos associados.

Neste capítulo foi descrito a metodologia utilizada para desenvolvimento das atividades, e a experiência que o estagiário teve utilizando-a. No próximo capítulo serão descritos os conhecimentos técnicos essenciais para a realização das atividades do estágio.

4 Fundamentação Teórica

4.1 Televisão Digital Brasileira

Esta seção introduz os conhecimentos técnicos necessários para realização das atividades do estagiário, e consistem em entender o Sistema Brasileiro de Televisão Digital Terrestre e suas normas, além do protocolo de comunicação TCP.

4.1.1 História

Em 1980, surgiu no Japão a proposta de desenvolvimento de uma televisão de alta definição, quando a tecnologia ainda era analógica. A idéia, capitaneada pela emissora pública japonesa NHK, era aumentar a área de exibição em tela e dobrar a resolução da imagem. Apesar de não ter vingado, entre outras razões, pela dificuldade de alocação de espectro compatível com tamanho incremento de qualidade (o formato analógico consome em demasia banda de transmissão), a proposta serviu para preocupar americanos e europeus a respeito da perda de mercado para a indústria japonesa (FREITAS, 2004).

É oportuno esclarecer outros conceitos básicos. A expressão TV Digital refere-se à digitalização do meio de transmissão dos sinais de vídeo e áudio que compõem o sinal de TV. O meio de transmissão pode ser aberto, ou seja, a recepção dos sinais ocorre de maneira livre, ou fechado, quando a emissão do sinal é precedida de “codificação” que impede sua livre recepção. Tal característica diferencia, em termos técnicos e jurídicos, a TV aberta (serviço de radiodifusão de sons e imagens, em meio aberto) da TV por assinatura (serviço de telecomunicações, em meio fechado).

Entre 1991 e 2000, os americanos, os europeus e, por último, os japoneses desenvolveram suas próprias soluções de radiodifusão digital terrestre, cujos padrões técnicos são identificados pelos acrônimos ATSC, DVB-T e ISDB-T, respectivamente. Em função do lapso de tempo entre eles, das realidades socioeconômicas e do contexto regulatório local, os critérios que moldaram o desenvolvimento de cada padrão foram distintos.

No caso do Brasil sob iniciativa e gestão do Grupo Abert/SET, foram realizados, nos anos de 1998 e 1999, vários testes de campo com as três tecnologias disponíveis. Os resultados foram encaminhados à ANATEL no ano seguinte, de forma a instruir o processo de tomada de decisão. Portanto, até 2000, todo o debate sobre TV Digital esteve restrito à escolha do padrão tecnológico de transmissão e à alocação de um espaço no espectro para que as emissoras pudessem proceder à migração.

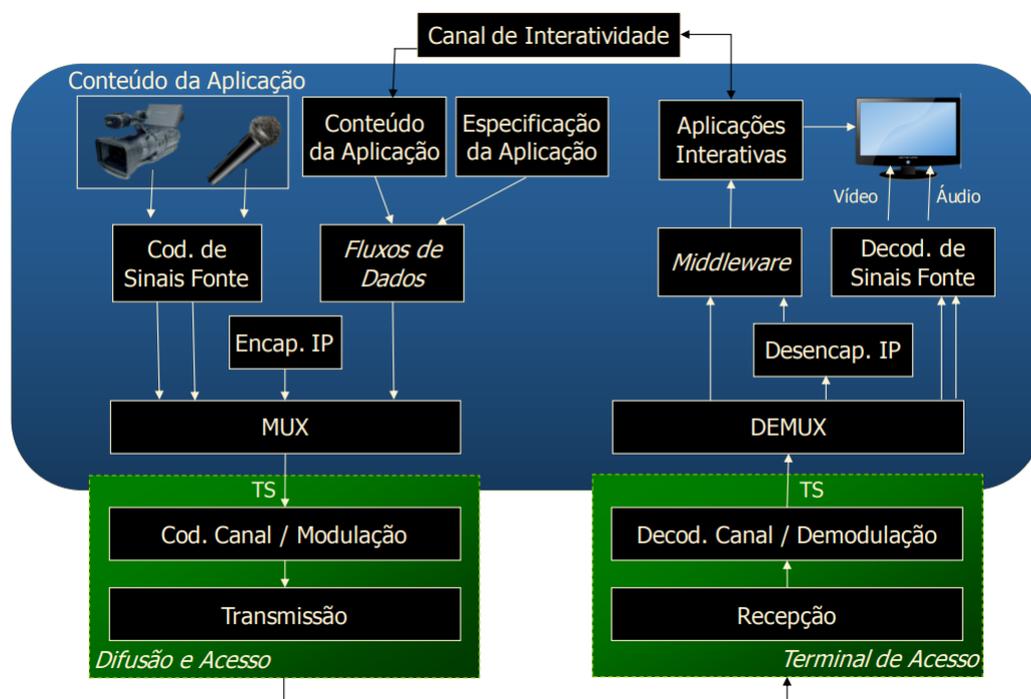
Entretando apenas em 2003 instituiu-se o Sistema Brasileiro de Televisão Digital

(SBTV D), que se tornou responsável pela condução do debate com a sociedade e pela produção de um modelo de referência a ser utilizado como norteador do processo de implantação da TV Digital a partir de 2005.

4.1.2 O Sistema Brasileiro de Televisão Digital (SBTV D)

Um sistema de TV digital é um sistema típico cliente/servidor. Em um sistema de TV digital terrestre, o servidor compõe o ambiente de uma radiodifusora (parte esquerda da Figura 5) ou de um provedor de conteúdo, e o cliente, o ambiente do usuário telespectador (parte direita da 5).

Figura 5 – Representação do sistema Digital de TV terrestre.



Fonte: (SOARES; BARBOSA, 2009)

Um programa é composto de um áudio principal e um vídeo principal, capturado ao vivo de uma câmera, como na Figura 5 (lado esquerdo) ou proveniente de um servidor de vídeo. Um programa pode também conter dados adicionais, incluindo o documento da aplicação que define o relacionamento entre os vários objetos de mídia definidos nesses dados. Os dados adicionais podem vir encapsulados no formato IP ou em outro formato, como veremos oportunamente. O vídeo e o áudio são entregues aos codificadores digitais, responsáveis pela geração dos respectivos fluxos de vídeo principal e áudio principal comprimidos. Esses fluxos, mais os fluxos dos outros dados, são então multiplexados em um único sinal, denominado fluxo de transporte (TS — *Transport Stream*). Após, no caso de um sistema terrestre, o fluxo de transporte é modulado para um canal de frequência e

transmitido no ar. Já em um serviço IPTV, em geral, o fluxo de transporte é transmitido diretamente em pacotes IP sem qualquer modulação. Do lado da recepção (lado direito da Figura 5), o sinal é recebido e demodulado (retirado do canal de frequência sintonizado), no caso da TV terrestre, e entregue ao demultiplexador, que separa os fluxos de áudio principal e vídeo principal (entregando-os a decodificadores apropriados) dos fluxos de dados, que são entregues para processamento.

O processamento dos dados recebidos pode demandar novos dados, obtidos pelo canal de interatividade. Dados também podem ser enviados pela aplicação à emissora ou outro destino qualquer na rede do canal de interatividade.

4.2 O Middleware

4.2.1 Ginga

Uma aplicação poderia ser executada diretamente sobre o sistema operacional de um receptor. No entanto, os sistemas operacionais de propósito geral não estão preparados para dar um bom suporte às aplicações de TV digital. Além disso, uma aplicação de TV deve ser capaz de ser executada em qualquer plataforma de hardware e sistema operacional.

Para tornar as aplicações independentes da plataforma de hardware e software de um fabricante de receptor específico e para dar melhor suporte às aplicações voltadas para a TV, uma nova camada é acrescentada nos padrões de referência de um sistema de TV digital. A essa camada denominamos *middleware*, no caso do Brasil o *middleware* utilizado é o Ginga.

Uma das funções de um *middleware* é fornecer suporte às aplicações. O suporte é fornecido através de uma interface de programação de aplicações (API — *Application Programming Interface*), cuja funcionalidade oferecida deve ser regida pelas necessidades das aplicações a serem executadas no ambiente de TV digital. Dentre essas aplicações, obviamente, estão os programas não-lineares, foco principal de qualquer sistema de TV digital.

Assim pode-se enumerar os requisitos de *middleware*, sendo estes (SOARES; BARBOSA, 2009):

1. suporte ao sincronismo de uma forma geral e, como caso particular, à interação do usuário;
2. Suporte à adaptação de conteúdo e da forma como um conteúdo é exibido;
3. suporte a múltiplos dispositivos de exibição;
4. Suporte à edição ao vivo (em tempo de exibição);

5. suporte à definição de relacionamentos de sincronismo espacial e temporal separado da definição do conteúdo dos objetos de mídia relacionados.

4.2.2 Ambientes de Programação de Aplicações

As aplicações de TV digital vão desde aquelas em que a aplicação transmitida não tem nenhuma relação semântica com o programa principal em exibição (como, por exemplo, a exibição de mensagens de notícias urgentes durante a exibição de um programa de TV) até os programas não-lineares (onde todos os objetos de mídia são sincronizados entre si, incluindo o áudio principal e o vídeo principal), passando pelas aplicações com vários objetos de mídia relacionados entre si e que só fazem sentido dentro do contexto do programa sendo assistido, mas não tendo nenhum relacionamento temporal com o áudio principal e o vídeo principal

Aplicações são usualmente desenvolvidas usando dois paradigmas de programação distintos: o declarativo e o não-declarativo.

O middleware Ginga obrigatoriamente requer o ambiente declarativo Ginga-NCL, que utiliza a linguagem declarativa NCL *Nested Context Language*, e geralmente a linguagem não declarativa LUA, mas admite extensões. Uma dessas extensões é o ambiente imperativo Ginga-J e mais recentemente o Ginga-HTML5.

4.3 Normatização ABNT

O desenvolvimento do Sistema Brasileiro de TV Digital Terrestre conhecido como ISDB-Tb, pois é baseado no sistema Japonês, requer normatizações dos padrões utilizados em cada parte que compõem a TV digital, para padronizações de *Hardware* e *Software* presentes nas TV's, estas normas são regidas pela Associação Brasileira de Normas Técnicas e são apresentadas na ABNT NBR 15606, sob o título geral "Televisão Digital Terrestre - Codificação de dados e especificações de radiodifusão digital", e tem como partes:

- Parte 1: Codificação de dados;
- Parte 2: Ginga-NCL para receptores fixos e móveis - Linguagem de aplicação XML para codificação de aplicações;
- Parte 3: Especificação de transmissão de dados;
- Parte 4: Ambiente para execução de aplicações procedurais;
- Parte 5: Ginga-J para receptores portáteis - Linguagem de aplicação XML para codificação de aplicações;
- Parte 6: Java DTV 1.3;

- Parte 7:Ginga NCL - Diretrizes operacionais para as ABNT NBR 15606-2 e ABNT NBR 15606-5;
- Parte 8: Ginga J - Diretrizes operacionais para a ABNT NBR 15606-5;
- Parte 9: Diretrizes operacionais para a ABNT NBR 15606-1 (cancelada);
- Parte 10: Ginga-HTML5 - Especificação do perfil HTML5 no Ginga;
- Parte 11: Ginga CC WebServices - Especificação de Web-Services do Ginga Common Core;
- Parte 12: Ginga CC WebServices - Diretrizes operacionais para a ABNT NBT 15606-11.

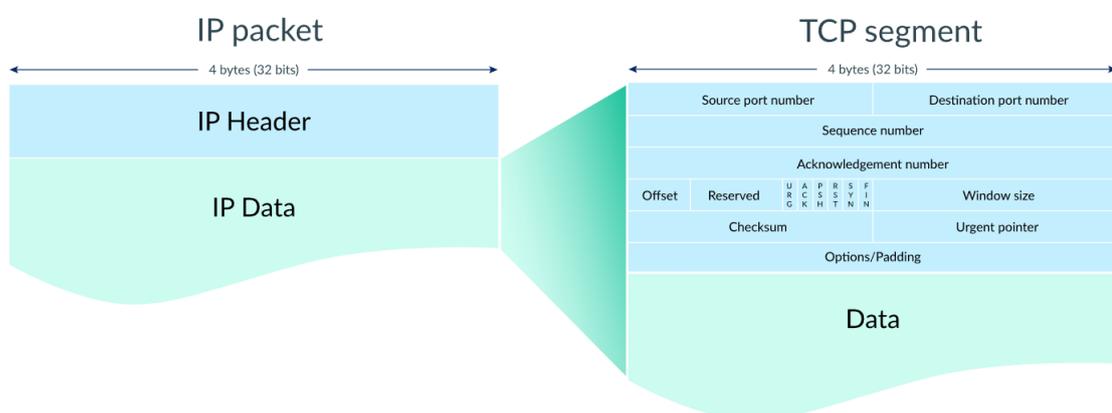
4.4 *Transmission Control Protocol (TCP)*

O *Transmission Control Protocol (TCP)* é um protocolo da camada de transporte da internet, o qual é usada em cima da camada de rede IP, para garantir uma transmissão de pacotes confiável.

O TCP inclui mecanismos para resolver muitos dos problemas que surgem utilizando mensagens baseadas em pacotes, como perdas de pacote, pacotes fora de ordem, pacotes duplicados, e pacotes corrompidos. Dado que o TCP é mais comumente utilizado em cima do pacote IP, o protocolo é referido como TCP/IP

Os pacotes transmitidos pelo TCP/IP, possuem uma porção de dados do cabeçalho do protocolo IP, e outra porção de dados que incluem o cabeçalho do protocolo TCP e os dados a serem transmitidos, como apresentado na Figura 6.

Figura 6 – Encapsulamento de dados TCP.



Fonte: (FOX,)

Estes conhecimentos técnicos descritos neste capítulo serão necessários para o entendimento das atividades realizadas durante o tempo de estágio. No próximo capítulo serão descritas as principais ferramentas e atividades realizadas pelo estagiário.

5 Atividades Desenvolvidas

Neste capítulo serão apresentadas as principais ferramentas e atividades desenvolvidas durante o tempo de estágio.

5.1 Ferramentas utilizadas

5.1.1 Git

Git é um sistema de controle de versão de arquivos. Através deles podemos desenvolver projetos na qual diversas pessoas podem contribuir simultaneamente no mesmo, editando e criando novos arquivos e permitindo que os mesmos possam existir sem o risco de suas alterações serem sobrescritas.

Foi utilizado o Gitlab em um servidor local, de forma que este era acessado por meio de uma VPN pelos desenvolvedores.

5.1.2 Clion

Clion é uma IDE de desenvolvimento em C++, que fornece facilidades na hora de desenvolver algoritmos, como assistência de código, geração de código, refatoração segura, análise estática do código realizada enquanto o código é escrito, busca automática de classes utilizando cliques em objetos descritos no código, além de possuir um depurador interno.

5.1.3 Microsoft Teams

O Microsoft Teams é uma plataforma de comunicações unificada que combina batepapo persistente no local de trabalho, reuniões de vídeo, armazenamento de arquivos e integração de aplicativos.

Essa plataforma foi utilizada no projeto, o que facilita a comunicação com integrantes da mesma equipe em outras localizações, assim como o contato com integrantes de outros projetos e outros setores, facilitando o trabalho *Home Office*.

5.1.4 Linguagem C++

A linguagem C++ é uma linguagem de nível médio de abstração que possui vantagens de programação em baixo nível (*drivers*, *Kernels*) ou mesmo de aplicações de alto nível (Jogos, Interfaces gráficas de Usuário, aplicativos, etc). C++ é uma linguagem

orientada a objeto que fornece uma estrutura clara para programas e aceita reuso de código.

O middleware Ginga é desenvolvido na linguagem C++.

5.1.5 NCLua

A linguagem NCLua na verdade é a combinação das linguagens declarativa NCL (*Nested Context Language*) e não declarativa Lua. Estas duas linguagens combinadas são a principal forma de desenvolvimento de aplicações para TV digitais que irão rodar no *middleware* Ginga.

5.1.6 Wireshark

O Wireshark é uma ferramenta livre que permite a análise de protocolos, podendo auxiliar administradores de rede na resolução de problemas. Ele realiza a captura e análise de informações trafegadas em diferentes tipos de meios de transmissão, sejam eles cabeados ou não, com a vantagem de compatibilidade entre os mais diversos sistemas operacionais. Além disso, por meio do Wireshark, torna-se possível conhecer diversas características e o funcionamento de vários tipos de protocolos. O conhecimento aprofundado dos protocolos proporciona a criação de novas ferramentas, uma melhor utilização dos recursos de cada protocolo e maior compatibilidade entre tecnologias.

5.1.7 Google C++ Mocking Framework

Um objeto Mock implementa a mesma interface de um objeto real (para que possa ser usado como um), mas permite que você especifique em tempo de execução como ele será usado e o que deve fazer (quais métodos serão chamados? Em que ordem? Como muitas vezes? com quais argumentos? o que eles vão retornar? etc).

Google C++ Mocking Framework é uma biblioteca ou framework para criar classes Mockadas e usá-las. Com o Google Mock é possível definir facilmente classes Mockadas, assim como criar sem dificuldades testes de unidades, definindo Testes de espera dentro do teste. Teste de espera são testes que acontecem dentro do teste de unidade, que irão definir a falha ou sucesso do teste, o qual podem testar se determinada função foi chamada n vezes, o valor de retorno para cada chamada desta função, ou testes de valores de uma determinada variável.

5.2 Descrição das Atividades Desenvolvidas

Nesta seção a seguir, são descritos os processos iniciais e também todas as atividades realizadas pelo estagiário. Algumas informações não são disponibilizadas nesse relatório

por serem consideradas confidenciais.

A função principal do estagiário no projeto era de desenvolvimento e melhoria do código do *middleware* Ginga, desenvolvendo principalmente em C++, serão descritas alguns tipos de atividades realizadas no Ginga de forma geral, para não comprometer a confidencialidade do projeto. As áreas principais do Ginga o qual o estagiário atuou foram:

- Suporte a comunicação TCP no Ginga;
- Suporte a arquivos Zip no Ginga;
- Interpretação e interação de propriedades de elementos da tela descritos em NCLua no Ginga;
- Criação de Aplicações em NCLua para validação de funções do Ginga;
- Criação de testes de unidades em C++ para validação de funções no Ginga.

A seguir serão detalhadas melhor as atividades.

5.2.1 Suporte a comunicação TCP no Ginga

Grande parte do estágio foi focado na parte de suporte a comunicação TCP do Ginga, já existiam testes de unidades, e aplicações que validavam como o Ginga devia se comportar em algumas situações de acordo com as normas da ABNT NBR 15606-2 e ABNT NBR 15606-7. O estagiário utilizou estas normas e testes como base para verificar o funcionamento do Ginga, e realizar as modificações necessárias no código para que a adequação fosse realizada.

Muitas das vezes era necessário utilizar um servidor TCP simples, desenvolvido em Java para realizar a conexão com o Ginga e validar as funcionalidades, e conseqüentemente utilizar um analisador do tráfego TCP, para observar a comunicação entre o servidor TCP escrito na linguagem Java e o software Ginga, para assim identificar onde estava o problema.

Foram realizadas atividades de verificação das funcionalidades de estabelecer conexão, envio de dados e criação de servidores no Ginga.

5.2.2 Suporte a arquivos Zip no Ginga

O *middleware* Ginga também fornece funcionalidades para gestão de arquivos do tipo Zip, como salvamento, leitura e extração desses tipos de arquivo. O estagiário forneceu suporte para estas funcionalidades, também utilizando como guia testes já criados e das normas da ABNT NBR 15606-2 e ABNT NBR 15606-7, que descrevem como o funcionamento destes tipos de evento.

5.2.3 Interpretação e interação de propriedades de elementos da tela descritos em NCLua no Ginga

O *middleware* Ginga como já citado em seção anteriores é o responsável por interpretar as aplicações que são executadas na TV digital, tais aplicações são descritas na linguagem NCLua, assim se faz necessário que Ginga interprete e interaja com esta. O estagiário atuou fornecendo suporte para funcionalidades das propriedades de elementos descritos em NCLua que podem aparecer na tela, como transparência do elemento e posição na tela, e nas funcionalidades de interação com a linguagem NCLua, como retorno correto das propriedades de elementos, este retorno correto também está descritos nas normas ABNT NBR 15606-2 e ABNT NBR 15606-7.

5.2.4 Criação de Aplicações em NCLua para validação de funções do Ginga

Afim de validar funcionalidades desenvolvidas ou correções realizadas, em algumas vezes se fazia necessário a criação de novas aplicações em NCLua para validar exatamente o que havia sido implementado, assim o estagiário também atuou na validação destas aplicações em NCLua para validar as próprias funcionalidades desenvolvidas.

Outra situação de desenvolvimento em NCLua ocorria quando um teste já criado já não estava mais de acordo com as normas da ABNT NBR 15606-2 e ABNT NBR 15606-7, e então o estagiário deveria observar a falha do código em NCLua, realizar a correção e reportar a equipe de desenvolvimento de aplicações para testes. Esta desatualização podia ocorrer pois a norma é periodicamente revisada pelo SBTVD, de forma que podiam haver alterações na norma que tornasse desatualizados alguns testes.

5.2.5 Criação de testes de unidades em C++ para validação de funções no Ginga

Assim como para validar as funcionalidades desenvolvidas era necessário aplicações em NCLua, também eram desenvolvidos testes de unidades de funções criadas no Ginga, a partir de Mocks das classes desenvolvidas ou modificadas em C++, para validar o seu funcionamento de forma isolada. Para a criação destes testes de unidades e Mocks das classes foi utilizado a ferramenta *Google C++ Mocking Framework*.

5.2.6 Relatórios

Após o estagiário ter desenvolvido as funcionalidades e melhorias, e seu respectivos testes de unidade em C++ e de aplicação em NCLua, caso não existam ainda, era realizado um relatório do que havia sido desenvolvido e repassado para o líder e realizado o commit dos códigos no gitlab do projeto, o qual seria passado pelo processo de revisão por outro

desenvolvedor mais capacitado. Este revisor poderia submeter solicitações de revisões no código que o estagiário deveria realizar, para assim submeter novamente o código para revisão, este processo continuaria até tudo está devidamente aceito pelo revisor.

Neste Capítulo foram apresentadas as atividades desenvolvidas durante o período de estágio e as ferramentas utilizadas para isto, que se referem ao desenvolvimento de algoritmos e melhorias aplicadas ao *middleware* Ginga presente no Sistema Brasileiro de Televisão Digital Terrestre. No próximo capítulo serão apresentadas as considerações finais deste trabalho.

6 Conclusões

Neste relatório foram descritas as principais atividades desenvolvidas em um projeto de pesquisa e desenvolvimento como parte da disciplina de Estágio Supervisionado. O aluno desenvolveu suas atividades no Laboratório de Sistemas Embarcados e Computação Pervasiva (Embedded). Foi cumprida a carga horária de aproximadamente 374 horas, ultrapassando a carga horária mínima para o estágio supervisionado matriculado (360 horas).

As atividades desenvolvidas foram solicitadas pelo líder de modelo de acordo com as demandas do projeto. Muitos desafios foram encontrados no desenvolvimento das atividades, principalmente pelo desconhecimento do sistema e normas do Sistema Brasileiro de Televisão Digital Terrestre e da linguagem de programação NCLua. Todavia, todas as atividades passadas pelo líder foram executadas, atingindo os objetivos propostos.

Ademais, o acompanhamento das etapas práticas envolvidas na execução de um projeto para desenvolvimento ágil de novas soluções, a necessidade de cumprimento de prazos, reuniões, discussões técnicas com profissionais da área e dificuldades encontradas nas atividades estimulam o desenvolvimento de competências e maturidade necessárias para a inserção no mercado de trabalho. Além disso, o projeto em que o estágio foi desenvolvido é alinhado com os interesses de mercado de uma grande empresa, o que proporcionou conhecer a um pouco da realidade da indústria.

Com a realização dessas atividades, o estagiário pôde colocar em prática os conceitos aprendidos durante a graduação, principalmente nas disciplinas Redes de Computadores e Informática Industrial.

Referências Bibliográficas

BRASIL, C. de Desenvolvimento Ágil do. Metodologia Ágil scrum. *Disponível em: <http://www.desenvolvimentoagil.com.br/scrum/>. Acesso em 12 de Outubro de 2021.* Citado na página 5.

FOX, P. Transmission control protocol (tcp). *Disponível em: <https://www.khanacademy.org/computing/computers-and-internet>. Acesso em 12 de Outubro de 2021.* Citado na página 11.

FREITAS, I. V. B. de. TELEVISÃO DIGITAL: QUE IMAGEM TERÁ O MODELO BRASILEIRO? *Consultoria Legislativa do Senado Federal*, 2004. Citado na página 7.

OLIVEIRA, A. D. B. Relatório de Estágio Supervisionado LIHM - Laboratório de Interface Homem-Máquina. *Relatório de Estágio*, 2021. Citado na página 5.

SCHWABER, K.; SUTHERLAND, J. The scrum guide. 2020. Citado na página 4.

SOARES, L. F. G.; BARBOSA, S. D. J. *Programando em NCL 3.0*. [S.l.]: Elsevier, 2009. 341 p. ISBN 9788535234572. Citado 2 vezes nas páginas 8 e 9.