INSTITUTO FEDERAL GOIANO - CAMPUS MORRINHOS

CURSO SUPERIOR DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

Matheus Fellipi Ribeiro de Oliveira

REENGENHARIA DE *SOFTWARE* DA FERRAMENTA CONTROLA

MORRINHOS - GO

2025

Matheus Fellipi Ribeiro de Oliveira

REENGENHARIA DE *SOFTWARE* DA FERRAMENTA CONTROLA

Monografia apresentada ao Curso Superior de Bacharelado em Ciência da Computação do Instituto Federal Goiano – Campus Morrinhos, como requisito parcial para obtenção de título de Bacharel em Ciência da Computação.

Área de concentração: Ciência da Computação

Orientador: Me. Norton Coelho Guimarães

MORRINHOS - GO

2025

Ficha de identificação da obra elaborada pelo autor, através do Programa de Geração Automática do Sistema Integrado de Bibliotecas do IF Goiano - SIBi

R484r

Ribeiro de Oliveira, Matheus Fellip
Reengenharia de software da ferramenta controla / Matheus Fellip Ribeiro de Oliveira. Morrinhos 2025.

54f. il.

Orientador: Prof. Me. Norton Coelho Guimarães.
Monografia (Bacharel) - Instituto Federal Goiano, curso de 0419204 - [MO.GRAD] Bacharelado em Ciência da Computação - Morrinhos (Campus Morrinhos).
1. Reengenharia de software. 2. Engeharia reversa. 3. Engenharia de requisito. I. Título.

SERVIÇO PÚBLICO FEDERAL
MINISTÉRIO DA EDUCAÇÃO
SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA GOIANO

Ata nº 23/2025 - CCEPTNM-MO/CEPTNM-MO/DE-MO/CMPMHOS/IFGOIANO

ATA DE DEFESA DE TRABALHO DE CURSO

Aos 16 dias do mês de setembro de 2025, às 16 horas, reuniu-se, via Google Meeting, a banca examinadora composta pelos docentes: Norton Coelho Guimarães (orientador), José Pereira Alves (membro), Marcel da Silva Melo (membro), para examinar o Trabalho de Curso intitulado "REENGENHARIA DE SOFTWARE DA FERRAMENTA CONTROLA" do estudante Matheus Fellipi Ribeiro de Oliveira, matrícula nº 2018104201940356, do Curso de Bacharelado em Ciência da Computação do IF Goiano – Campus Morrinhos. A palavra foi concedida ao estudante para a apresentação oral do TC, houve arguição do candidato pelos membros da banca examinadora. Após tal etapa, a banca examinadora decidiu pela APROVAÇÃO do estudante COM RESSALVA. Ao final da sessão pública de defesa foi lavrada a presente ata que segue assinada pelos membros da Banca Examinadora.

(Assinado Eletronicamente)

Norton Coelho Guimarães

Orientador

(Assinado Eletronicamente)

José Pereira Alves

Membro

(Assinado Eletronicamente)

Marcel da Silva Melo

Membro

Observação:

( ) O(a) estudante não compareceu à defesa do TC.

Documento assinado eletronicamente por:

- **Norton Coelho Guimaraes**, **PROFESSOR ENS BASICO TECN TECNOLOGICO** , em 16/09/2025 17:57:24.
- **Marcel da Silva Melo**, **PROFESSOR ENS BASICO TECN TECNOLOGICO** , em 16/09/2025 18:02:05.
- **Jose Pereira Alves**, **PROFESSOR ENS BASICO TECN TECNOLOGICO** , em 17/09/2025 13:57:02.

Este documento foi emitido pelo SUAP em 20/05/2025. Para comprovar sua autenticidade, faça a leitura do QRCode ao lado ou acesse https://suap.ifgoiano.edu.br/autenticar-documento/ e forneça os dados abaixo:

Código Verificador: 708734
Código de Autenticação: 06beab0e6f

INSTITUTO FEDERAL GOIANO
Campus Morrinhos
Rodovia BR-153, Km 633, Zona Rural, SN, Zona Rural, MORRINHOS / GO, CEP 75650-000
(64) 3413-7900

INSTITUTO FEDERAL
Goiano

Repositório Institucional do IF Goiano - RIIF Goiano
Sistema Integrado de Bibliotecas

TERMO DE CIÊNCIA E DE AUTORIZAÇÃO

PARA DISPONIBILIZAR PRODUÇÕES TÉCNICO-CIENTÍFICAS NO REPOSITÓRIO INSTITUCIONAL DO IF GOIANO

Com base no disposto na Lei Federal nº 9.610, de 19 de fevereiro de 1998, AUTORIZO o Instituto Federal de Educação, Ciência e Tecnologia Goiano a disponibilizar gratuitamente o documento em formato digital no Repositório Institucional do IF Goiano (RIIF Goiano), sem ressarcimento de direitos autorais, conforme permissão assinada abaixo, para fins de leitura, download e impressão, a título de divulgação da produção técnico-científica no IF Goiano.

IDENTIFICAÇÃO DA PRODUÇÃO TÉCNICO-CIENTÍFICA

- ☐ Tese (doutorado)
- ☐ Dissertação (mestrado)
- ☐ Monografia (especialização)
- ☑ TCC (graduação)
- ☐ Artigo científico
- ☐ Capítulo de livro
- ☐ Livro
- ☐ Trabalho apresentado em evento
- ☐ Produto técnico e educacional - Tipo:

Nome completo do autor: Matheus Fellipi Ribeiro de Oliveira

Matrícula: 2018104201940356

Título do trabalho: REENGENHARIA DE SOFTWARE DA FERRAMENTA CONTROLA

RESTRIÇÕES DE ACESSO AO DOCUMENTO

Documento confidencial: ☑ Não ☐ Sim, justifique:

Informe a data que poderá ser disponibilizado no RIIF Goiano: 05 /10 /2025

O documento está sujeito a registro de patente? ☐ Sim ☑ Não

O documento pode vir a ser publicado como livro? ☐ Sim ☑ Não

DECLARAÇÃO DE DISTRIBUIÇÃO NÃO-EXCLUSIVA

O(a) referido(a) autor(a) declara:

• Que o documento é seu trabalho original, detém os direitos autorais da produção técnico-científica e não infringe os direitos de qualquer outra pessoa ou entidade;

• Que obteve autorização de quaisquer materiais inclusos no documento do qual não detém os direitos de autoria, para conceder ao Instituto Federal de Educação, Ciência e Tecnologia Goiano os direitos requeridos e que este material cujos direitos autorais são de terceiros, estão claramente identificados e reconhecidos no texto ou conteúdo do documento entregue;

• Que cumpriu quaisquer obrigações exigidas por contrato ou acordo, caso o documento entregue seja baseado em trabalho financiado ou apoiado por outra instituição que não o Instituto Federal de Educação, Ciência e Tecnologia Goiano.

Morrinhos — Local
28 /09 /2025 — Data

gov.br Documento assinado digitalmente
MATHEUS FELLIPI RIBEIRO DE OLIVEIRA
Data: 01/10/2025 10:21:04-0300
Verifique em https://validar.iti.gov.br

Assinatura do autor e/ou detentor dos direitos autorais

Ciente e de acordo:

gov.br Documento assinado digitalmente
NORTON COELHO GUIMARAES
Data: 01/10/2025 16:29:56-0300
Verifique em https://validar.iti.gov.br

Assinatura do(a) orientador(a)

DEDICATÓRIA

Dedico este TCC ao meu irmão que me deu todo o suporte necessário, financeiro e emocional, para que eu pudesse terminar a faculdade.

AGRADECIMENTOS

Agradeço meus familiares e o Luan Henrique, Patrícia do Carmo e Valnir Rodrigues que me deram apoio para a conclusão deste trabalho.

Agradeço ao meu professor, Norton Guimarães, por ser meu orientador neste trabalho e pela a paciência.

Agradeço aos meus amigos Bárbara, Leonardo, Emanuelly e Júlia que me ajudaram no decorrer do processo deste trabalho.

RESUMO

Esse trabalho aborda a reengenharia de *software* aplicada na ferramenta Controla v10. É um sistema de apoio para gerenciamento de requisitos, com a necessidade de modernizar e a ausência de acesso ao código-fonte. O estudo buscou compreender o comportamento e a estrutura do sistema por meio de técnicas próprias da engenharia de requisitos e da engenharia reversa. *Para* que isso ocorresse, foram realizadas revisões bibliográficas e documentais fundamentadas em autores como Bezerra (2015) e Sommerville (2016). Além disso, foram feitas análises práticas das interfaces gráficas e dos fluxos de interação do sistema legado. O levantamento de requisitos funcionais e não funcionais, bem como as regras de negócios, permitiu a criação da nova ferramenta UCase para gerenciamento de requisitos. Também houve mudanças nas interfaces, a reestruturação de dados, modelados usando diagramas de UML. O resultado obtido por meio da abordagem permitiu encontrar pontos críticos e pontos a serem melhorados no sistema, contribuindo para a modernização do sistema de gestão de requisitos. O trabalho viabiliza a utilização de métodos técnicos para reengenharia de sistema legado que pode ser usado para modelar um sistema.

Palavras-chave: Reengenharia, Engenharia de Software, Ferramenta Controla

ABSTRACT

This paper addresses software reengineering applied to the Controla v10 tool. It is a requirements management support system that requires modernization and lacks access to source code. The study sought to understand the system's behavior and structure through techniques specific to requirements engineering and software backup engineering. To achieve this, bibliographic and documentary reviews were conducted based on authors such as Bezerra (2015) and Sommerville (2016). Furthermore, practical analyses of the legacy system's graphical interfaces and interaction flows were performed. The identification of functional and non-functional requirements, as well as business rules, enabled the creation of the new UCase tool for requirements management. Interface changes were also made, as well as data restructuring, modeled using UML diagrams. The results obtained through this approach identified critical points and areas for improvement in the system, contributing to the modernization of the requirements management system. The work enables the use of technical methods for legacy system reengineering that can be used to model a system.

Keywords: Reengineering, Software, Controls

SUMÁRIO

LISTA DE FIGURA

LISTA DE QUADRO

INTRODUÇÃO

A engenharia reversa começou inicialmente na área de hardware, em que se utiliza para compreender a estrutura do dispositivo sem acesso à documentação original. Segundo o Eilam (2005), engenharia reversa é o processo de extrair o conhecimento ou projetos de *design* de qualquer coisa feita pelo homem. O conceito existe desde muito antes dos computadores ou da tecnologia moderna e provavelmente remonta aos dias da revolução industrial (Eilam, 2005, p. 3).

Com o avanço da computação, a prática expandiu-se para o campo da área de desenvolvimento de *software, transformando-se* em uma ferramenta indispensável para análise e compreensão de sistemas complexos. Nos primeiros momentos, a engenharia reversa de *software* era envolvida com técnicas manuais de desmontagem de código e análises de binários, sendo muito trabalhoso e suscetível a muitos erros. Com o passar do tempo, foram desenvolvidas ferramentas automatizadas, descompilação, análise estática e dinâmica. Fazendo com que esse processo evolua significativamente (Eilam, 2005).

A engenharia reversa em sistema legado desempenha um papel fundamental, permite a compreensão detalhada do sistema e possibilita a identificação de vulnerabilidades e recuperação de código perdido.

No cenário atual, em que a tecnologia evolui rapidamente, as empresas precisam modernizar seus sistemas para acompanhar essas inovações. Muitos sistemas legados continuam em uso porque, embora tenham sido desenvolvidos com tecnologias antigas, são essenciais para as operações e para o domínio de negócios das organizações. Em muitos casos, é mais econômico e menos arriscado manter e modernizar um sistema existente do que recriá-lo do zero (Eilam, 2005).

Ela não se limita apenas à recuperação de código-fonte, mas abrange também a extração de modelos conceituais de identificação de arquitetura e documentação. Que desempenham um papel para documentar os sistemas que carregam um vasto acúmulo de conhecimento que, muitas vezes, não está sendo documentado de maneira adequada.

1.1 Os problemas da manutenção de software e a necessidade de refazer o software legado

A manutenção de um sistema trata-se da adaptação, atualização e a otimização desse *software* para que elas operem de acordo com as demandas do mercado atual. Além disso, ela também abrange as melhorias e adaptações que são necessárias para o sistema ficar alinhado com a empresa e as inovações tecnológicas.

> "…As alterações feitas no software podem ser simples mudanças para correção de erros de codificação, até mudanças mais extensas para correção de erros de projeto, ou melhorias significativas para corrigir erros de especificação ou acomodar novos requisitos. As mudanças são implementadas por meio da modificação de componentes do sistema existente e, quando necessário, por meio da adição de novos componentes". (Sommerville, 2018 ,p. 170)

Um dos desafios da manutenção de *software é o seu alto custo*. Sommerville (2018) aponta que a manutenção consumiu média de dois terços do orçamento de um desenvolvimento (Sommerville, 2018).

No caso dos sistemas legados — definidos como *softwares* ou tecnologias que, embora baseados em ferramentas ultrapassadas, continuam fundamentais para as operações diárias da empresa —, a manutenção se torna ainda mais desafiadora com o passar do tempo. O custo para se manter o sistema aumenta, pois as ferramentas de suporte são descontinuadas e também há escassez de profissional especializado (Sommerville, 2018).

Neste cenário, muitas vezes surgem a necessidade de recriar o próprio sistema utilizando tecnologias mais modernas para garantir a segurança e desempenhar a melhor facilidade de manutenção; entretanto, essa transição é complexa e custosa, principalmente porque os sistemas raramente têm um tempo de vida definido e permanecem em uso diante dos desafios acrescentados até pela manutenção (Sommerville, 2018, p. 170).

1.2 Importância das ferramentas de Gestão de Requisitos de *Software*

As ferramentas de gestão auxiliam na elaboração e organização de processos internos da empresa. Essas ferramentas não apenas apoiam os gestores no acompanhamento e gerenciamento dos processos, mas também ajudam a evitar problemas que podem surgir durante o desenvolvimento do *software.* Uma ferramenta de gestão de requisito, por exemplo, pode garantir a rastreabilidade dos requisitos levantados permitindo com o controle maior sobre todo o processo do projeto, dessa forma todos envolvidos conseguem visualizar o que está sendo implementado, categorizando as demandas com base em suas urgências (Sommerville, 2018).

O gerenciamento eficaz dos requisitos é extremamente importante, especialmente em projetos de grandes escalas, pois um controle inadequado pode levar ao desvio de escopo e, consequentemente, ao fracasso do projeto. Portanto, definir e gerenciar os requisitos de forma sistemática é essencial para que os engenheiros mantenham controle do escopo do projeto, direcionando corretamente o ciclo de vida do projeto (Sommerville, 2018).

1.3 Evolução constante do software

O cenário atual da computação há constante evolução da tecnologia e faz com que seja necessário os sistemas terem manutenções e atualizações de forma que acompanhem as inovações atuais. *Softwares* legados, por sua vez, frequentemente apresentam dificuldade de manutenção, o que leva ao seu abandono por parte de empresas ou à reescrita completa do sistema, utilizando, quando disponível, a própria documentação original. Por sua vez, a implementação do zero pode replicar os mesmos erros e implicar custos elevados (Sommerville, 2018).

A engenharia reversa demonstra-se uma abordagem estratégica para a análise detalhada do *software* existente, permitindo identificar pontos críticos e recuperar informações a partir do código-fonte. Esse processo preserva o conhecimento acumulado ao longo da sua utilização. Dessa forma, a engenharia reversa é um passo essencial para dentro da reengenharia de software,

possibilitando reorganizar, modularizar e melhorar programas legado, ele apoia a incorporação de novas tecnologias e a prática do moderno no desenvolvimento, reduzindo custo, minimizando os riscos associados e é reescrita completa do sistema (Sommerville, 2018).

1.4 Objetivos

O trabalho presente tem como o propósito principal orientar um processo de reengenharia de software usando a ferramenta Controla V10, identificando pontos de melhoria, requisitos essenciais e funcionalidades que podem ser otimizadas para um desenvolvimento de uma nova ferramenta

1.4.1 Objetivo Geral

Realizar a reengenharia de software Controla v10.

1.4.2 Objetivos Específicos

Desenvolver a prototipação visual do novo software com base no Controla v10.

Desenvolver a documentação de um novo software com base no Controla v10.

2 BIBLIOGRAFIA CORRELATA

A Ferramenta Controla foi desenvolvida pelo Prof. Clayton Vieira, Doutor em Ciências Florestais pela Universidade Federal do Espírito Santo (UFES), em 2005. A ferramenta foi desenvolvida para apoiar o processo de desenvolvimento de *software* em pequenas empresas, com foco em gerenciamento de requisitos que permite a identificação das requisições junto ao *stakeholder* (cliente/usuário do sistema), a sua matriz de rastreabilidade e o gerenciamento das mudanças. A ferramenta está sendo utilizada por vários usuários, no Brasil, entre estudantes de graduação e pós-graduação, professores e pequenas empresas de desenvolvimento de software (Fraga Filho & Reis, 2005).

O *IBM Rational RequisitePro* é um código de fonte fechado desenvolvido pela IBM que permite o gerenciamento de requisitos do projeto, ou seja, a ferramenta dá suporte para uma complexidade de grande escala, fornecendo para os *designers,* os desenvolvedores ou as partes interessadas uma visão organizada do objetivo do projeto que está sendo desenvolvido, fornecendo também rastreabilidade de mudanças de requisitos ao longo do tempo (IBM, 2018).

O *Visure Requirements* é uma ferramenta desenvolvida pela Visure de código fechado. É uma plataforma de *ALM* (gerenciamento do ciclo de vida das aplicações). A ferramenta permite a criação e a edição do requisito facilmente, permitindo que simplifique o processo de captura dos requisitos complexos e precisos nas necessidades exigidas pelo cliente (VISURE, 2024).

O *Helix* é um conjunto de ferramentas de código fechado desenvolvida pela *Perforce* para o gerenciamento de ciclo de vida de uma aplicação, tendo um módulo como gerenciamento de requisitos, gerenciamento de caso de testes e gerenciamento de problemas. Ele visa organizar os requisitos de sistemas para as partes interessadas visualizarem o projeto, oferecendo a rastreabilidade dos requisitos solicitados, podendo saber quais passaram ou falharam no teste, ou se ocorreu alguma mudança no decorrer do desenvolvimento (PERFORCE, 2025).

3 REFERENCIAL TEÓRICO

3.1 Reengenharia de *Software*

Reengenharia de *software* são processos de remodelação para sistemas que já estão em uso, podendo envolver tanto a re-documentação do sistema quanto a refatoração da arquitetura, modernização de linguagem, entre outras. Geralmente são realizadas em aplicações ou sistemas legado onde há uma dificuldade de execução de manutenção (Mens & Tourwe, 2004).

A realização dos processos possuem vários benefícios, ao invés de substituir o sistema podendo ter mais custo e mais risco, a reengenharia pode reduzir os riscos de erros de levantamento de requisitos ou de regras de negócios, reduzindo tais riscos e custos:

> “Risco reduzido. Existe um alto risco em desenvolver novamente um software crítico de negócios. Podem ocorrer erros na especificação de sistema ou pode haver problemas de desenvolvimento. Atrasos no início do novo software podem significar a perda do negócio e custos adicionais.
>
> Custo reduzido. O custo de reengenharia pode ser significativamente menor do que o de desenvolvimento de um novo software. Ulrich (1990) cita um exemplo de um sistema comercial cujos custos de re-implementação foram estimados em 50 milhões de dólares. O sistema foi reconstruído com sucesso por 12 milhões de dólares. Suspeito que, com a tecnologia moderna de software, o custo relativo de re-implementação é provavelmente inferior a esse, mas ainda consideravelmente superior aos custos da reengenharia” (Sommerville, 2018 ,p. 174).

3.2 Engenharia de Requisitos

A engenharia de requisitos é o processo de analisar, descobrir, documentar e verificar os serviços e as restrições. Tendo diferentes níveis de requisitos para a comunicação. As informações sobre o sistema devem ser escritas para diferentes tipos de leitores, a divisão se dá por requisito do usuário e requisito do sistema (Sommerville, 2018).

Os requisitos do usuário usam a linguagem mais natural e diagramas, que especificam de forma simples as necessidades que o sistema deve fornecer para o usuário final (Sommerville, 2018).

Os requisitos do sistema, também chamados de especificações funcionais, definem exatamente o que deve ser implementado no sistema e são escritos de forma robusta. As funcionalidades de cada requisito são especificadas com clareza, permitindo aos leitores entender de forma precisa como o sistema funcionará e como apoiará as regras de negócios. Esse detalhamento é essencial para os desenvolvedores, que utilizarão essas especificações para implementar as funcionalidades do sistema de maneira eficaz (Bezerra, 2015; Sommerville, 2018).

Os requisitos funcionais descrevem o que o sistema fornecerá ao usuário em termos de finalidades e serviços oferecidos. Eles determinam o comportamento do sistema para cada entrada de dados e como ele deve realizar as operações com essas entradas específicas. Além disso, podem ser usados para descrever o que o sistema não deve fazer em determinados casos de entrada de dados (Bezerra, 2015).

Os requisitos não funcionais, por sua vez, variam significativamente dependendo da área de atuação da empresa, e a abordagem para escrevê-los pode mudar conforme o contexto. É crucial que sejam escritos de forma clara para evitar ambiguidades e confusões entre os *stakeholders* (Bezerra, 2015).

Em muitos casos, esses requisitos não funcionais são mais críticos que os requisitos funcionais, pois descrevem como o sistema deve agir, geralmente relacionados a aspectos como desempenho, segurança, disponibilidade, tempo de resposta, entre outros. Diferentemente dos requisitos funcionais, que se referem diretamente aos serviços específicos oferecidos pelo sistema, os requisitos não funcionais podem incluir restrições baseadas em políticas organizacionais, *software* ou *hardware* dos equipamentos, e fatores internos como legislações de privacidade (Sommerville, 2018).

Atender a um requisito não funcional pode acarretar mudanças na arquitetura do sistema, dependendo da sua natureza. Esses requisitos também podem ser

subdivididos em categorias como requisitos de produto, requisitos de desempenho, requisitos de confiabilidade, entre outros (Bezerra, 2015).

3.2.1 Elicitação e análise de requisito

Elicitação são técnicas de coleta, descoberta e definição usadas na engenharia de requisitos, como entrevistas, observação e questionários, para compreender claramente as etapas do desenvolvimento. Nesta etapa, os engenheiros, junto aos seus clientes e usuários finais, obtêm informações por meio de observações, reuniões, análise de telas, entrevistas, cenários e engenharia reversa do sistema. Essas técnicas são usadas para obter os requisitos. Eles não apenas analisam todo o progresso do começo ao fim, mas também consideram o *hardware* das máquinas que possivelmente vão rodar o sistema (Bezerra, 2015).

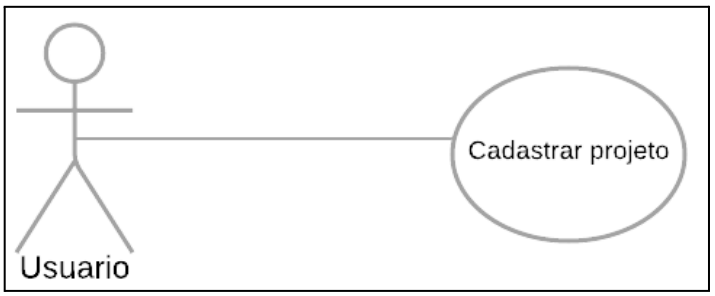
3.2.2 Os artefatos de requisitos de *software*

Na engenharia de requisitos, também há a parte de documentação, que atribui praticamente todos os requisitos funcionais e não funcionais do sistema. As informações da documentação vão depender do sistema a ser desenvolvido, sendo levantadas em reuniões com os *stakeholders* das empresas. Em geral, a documentação é usada para contratos, escrever testes, ver como o sistema vai funcionar, planejar propostas e processos, e entender os requisitos e os relacionamentos entre eles [Mens & Tourwe, 2004].

3.2.2.1 Caso de uso

O caso de uso é uma técnica característica de modelagem unificada (UML — do inglês *Unified Modeling Language*). Ela apresenta de forma simples os serviços e funcionalidades do *software* que irá disponibilizar. No diagrama de caso de uso, é possível identificar autores, representados por bonecos de palitos, que podem ser outros sistemas ou *stakeholders*, e cada interação no sistema é representada por uma elipse que se liga por meio de uma linha, como na Figura 1 [Guedes, 2007].

Figura 1 - Exemplo de um use case.



Fonte: Produção Própria.

3.2.2.2 Regras de Negócio

> “Regras do negócio são políticas, condições ou restrições que devem ser consideradas na execução dos processos existentes em uma organização" (Gottesdiener, 2001 apud Bezerra, 2015 pg. 85).

As regras de negócios são políticas ou diretrizes de uma organização que devem ser seguidas. No desenvolvimento de *software,* sendo geralmente levantada com a análise de requisitos. E também tem um papel crucial para garantir que a implementação atende às necessidades da organização e garantir o alinhamento e o objetivo estratégico da organização e também garante que a legislação legal do sistema (Bezerra, 2015).

Geralmente a regra que se conecta na descrição de um caso de uso influencia a lógica de execução ou nos fluxos de como o sistema deve se comportar na entrada de dados, gerando fluxos de exceções baseados na regra (Bezerra, 2015).

3.2.3 Análise de interface do sistema

Uma das técnicas usadas para elicitação de requisitos de sistema é a análise de interface. Essa técnica revela os requisitos funcionais, funcionalidades e trocas de dados entre sistemas (IIBA, 2009). Usando a análise de interface do sistema, você pode obter requisitos de outro sistema para adicionar ao seu, no caso de sistemas semelhantes. Descrevendo dados de entrada, as regras para esses dados e suas validações, é possível identificar funcionalidades que não precisam ser implementadas no sistema com base na análise de interface. Através dessa análise, você pode aprender mais sobre o sistema, utilizando outro sistema como referência,

ou localizar requisitos que não são necessários na interface (Wiegers & Beatty, 2013).

4 METODOLOGIA

Neste trabalho foi realizadas técnicas práticas descritas em literaturas como do Bezerra (2007; 2015) e do Sommerville (2018). Essas técnicas possibilitam compreender o comportamento do *software* por meio da engenharia de requisitos e da engenharia reversa. Como o código fonte não está disponível, adotamos abordagens alternativas, tais como: reconhecimento de informação em interface gráfica e teste prático no sistema.

4.1 O levantamento bibliográfico

Nesta etapa, houve a revisão da literatura, que foi extremamente importante para estudar as técnicas que seriam usadas nas etapas seguintes, pois demonstra como devem ser feitos os procedimentos de levantamento de requisitos e a análise no sistema, usando o fundamento teórico dos autores como Bezerra (2007; 2015) e Sommerville (2018). Também houve a análise do documento autor do Clayton Vieira Fraga Filho (2005), que possibilitou a compreensão de qual seria o escopo do Controla.

4.2 Análise do sistema legado

A Análise do sistema legado foi realizado com base nas técnicas de elicitação de requisitos apresentados no capítulo anterior com análise de interface, observação de interações e levantamento de dados de entrada e saída.

A Etapa da análise do sistema foi basicamente observar e interagir com o sistema para analisar os comportamento de cada interface gráfica e os fluxos de interações, permitindo analisar a estrutura de dados de saída e entrada de cada interface. Nesta etapa permitiu levantar os requisitos funcionais e as regras de negócio com base nas suas relações entre as telas e em seus fluxos.

4.3 Levantamento de requisito e regras de negócio

Com base na etapas anteriores, foram extraídos os seguintes artefatos de aplicação:

- Requisitos funcionais: São funcionalidades do sistema (ex.: cadastro de usuário, geração de relatórios).
- Requisitos não funcionais: aspectos relacionados a desempenho, segurança e tecnologia usada no sistema (ex.: linguagem de programação, gerenciamento do banco SGBD)
- Regras de negócios: Adiciona limitações políticas e não políticas no próprio sistema impactando em seu desenvolvimento (ex.: Requisitos devem ser categorizados como **funcionais** ou **não funcionais** no momento do cadastro.)

4.4 Proposição da nova ferramenta *CASE* para Gerência de Requisitos

Com base nos levantamentos de requisitos, foi desenvolvida uma proposta de prototipação de uma nova ferramenta, usando ferramentas auxiliares para criar a interface gráfica e para criação do diagrama de UML, como o Figma (2025) e o Astah (2025).

4.5 Comparação dos Artefatos

Após a proposta da nova ferramenta, foi realizada uma comparação com os requisitos levantados no sistema legado do Controla v10 e da nova proposta ao sistema. Essa comparação envolveu a criação de quadros de requisitos funcionais e não funcionais, da regra de negócio, a construção do diagrama de classe e como funcionará o novo sistema em comparação ao sistema legado. Essa análise permitiu identificar pontos de melhoria nos novos módulos a serem implementados na aplicação.

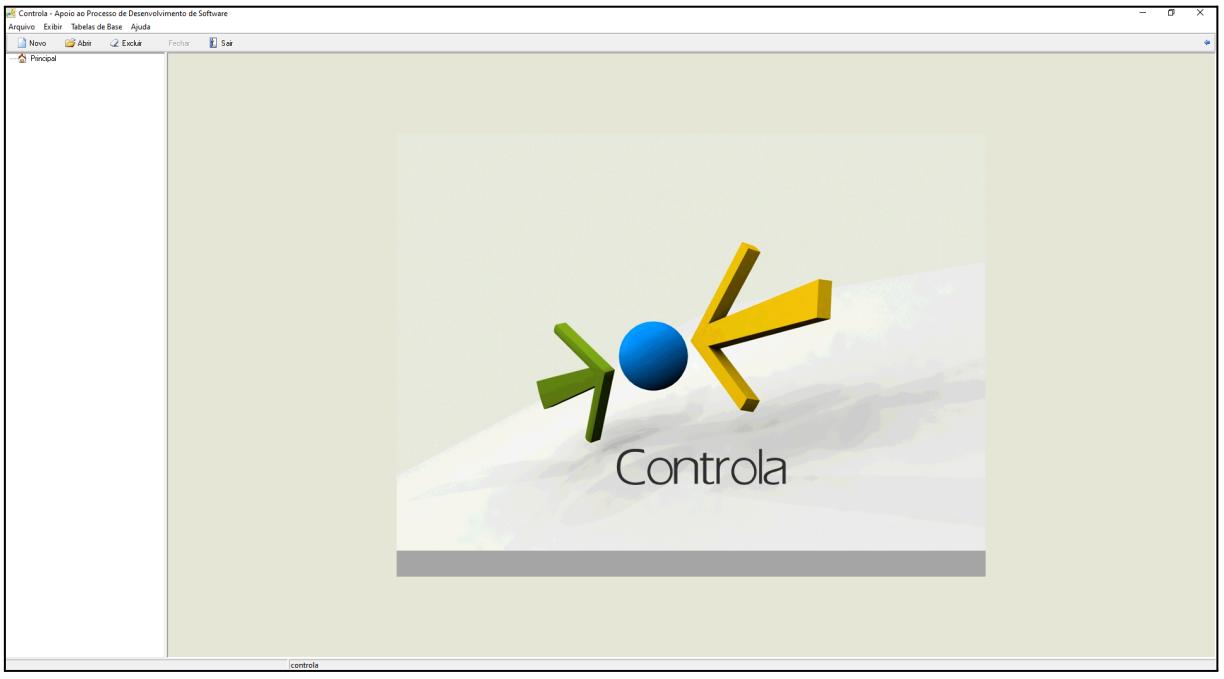
5 RESULTADOS E DISCUSSÕES

5.1 - Definição da Ferramenta Controla

Controla é uma ferramenta de apoio ao processo de desenvolvimento de *software* criado pelo Dr. Clayton Vieira Fraga Filho em 2005, como TCC na sua graduação, tendo como foco o gerenciamento de requisitos. Em seu artigo (Fraga Filho & Reis, 2005) ele descreve quais possibilidades a ferramenta oferece, sendo elas: Levantamento e registro dos requisitos do usuário; Projeto e especificação do sistema; Registro de usuário, entre outros.

O instalador do Controla v10 de 2010 está disponível no drive do próprio autor no DropBox (Fraga Filho, 2010). Após a instalação será exibido uma Tela, conforme a Figura 2.

Figura 2 - Interface do *software* Controla



Fonte: Produção do autor (Fraga Filho & Reis, 2005).

5.1.1 Escopo da Análise e Relevância dos Requisitos

Com base no referencial teórico de Bezerra, que modela aplicações utilizando engenharia de requisitos com UML, e de Sommerville, especialmente o Capítulo 4 de *Engenharia de Requisitos* do livro *Engenharia de Software*, que aborda o levantamento de requisitos a partir de documentação, toda a ferramenta Controla foi analisada. Para a proposta de desenvolvimento de uma nova ferramenta, optou-se por concentrar a análise exclusivamente nas funcionalidades relacionadas à engenharia de requisitos, priorizando aspectos relevantes para a nova solução. Os aspectos relacionados a casos de teste, encerramento de teste, liberação e simulações foram excluídos da nova proposta.

5.1.2 Métodos Utilizados no Levantamento de Requisitos

Para obter a melhor clareza de como o sistema está funcionando utilizamos abordagens e metodologias conforme recomendada nos livros do Bezerra (2007) e do Sommerville (2018).

5.1.3 - Levantamento dos Requisitos Funcionais

Antes de fazer o levantamento de requisitos do Controla, é de extrema importância compreender o domínio em que o sistema se aplica e o objetivo principal. De acordo com Bezerra (2007), o entendimento do domínio é uma etapa fundamental para identificar os requisitos, pois isso possibilita identificar os requisitos de forma mais precisa. Ele define domínio como:

> “A área de conhecimento ou de atividade específica é caracterizada por um conjunto de conceitos e de terminologia compreendidos por especialistas nessa área” (Bezerra, 2007, p. 22-23).

Isso significa que entender o domínio onde o controle se aplica é possível levantar os requisitos mais precisos garantindo que o sistema atenda as necessidades de cada empresa no apoio de desenvolvimento de um determinado *software*.

5.1.3.1 Análise documental

Umas das técnicas utilizadas foi análise documental. O principal documento utilizado nesta análise foi o próprio artigo desenvolvido por Fraga Filho, o autor do sistema controla, especificamente o artigo “Controla: Ferramenta de Apoio ao Processo de Desenvolvimento de *Software* em Pequenas Empresas”, nesse documento descreve a ferramenta e as suas funcionalidades e seu foco operacional, permitindo a compreensão mais detalhada do sistema.

No Bezerra (2007), análise documental não é explicitamente citada, mas ele menciona que a leitura de obras de referência constitui uma abordagem válida para o levantamento de requisitos. No livro de Engenharia de *Software* Sommerville (2018), explica que durante o processo de levantamento de requisitos uma documentação existente fornecida pelos os *stakeholders* pode fornecer informações valiosas a ser descoberto (Sommerville, 2018).

5.1.3.2 Análise do sistema existente

A principal abordagem utilizada no levantamento de requisitos foi a análise do sistema existente já implementado. Uma abordagem que o Bezerra (2007) cita é a comparação com o sistema preexistente no mesmo domínio de negócio, sendo uma abordagem válida para desenvolvimento de novos *software* no mesmo negócio.

Dessa forma, foi utilizada uma abordagem para analisar a interface gráfica, fluxo de interações e estrutura de dados. O objetivo foi observar como cada interface, seja com formulários ou não, como os dados são armazenados, realizando as interações no sistema, permitindo avaliar quais requisitos podem ser extraídos, adicionados ou removidos, podendo até evitar redundância futura no sistema.

Com os métodos de análise sistema existentes e a própria análise documental foi possível fazer o levantamento do requisito funcional (Quadro 01) e não funcionais (Quadro 02).

Quadro 01 - Lista de Requisitos Funcionais do Controla

ID	Descrição

RF01	Manter usuário
RF02	Manter projeto
RF03	Manter empresa
RF04	Manter cliente
RF05	Manter os usuários do cliente
RF06	Manter Pessoa da equipe
RF07	Manter funções das pessoas da equipe
RF08	Manter expectativa do usuário
RF09	Manter requisitos não funcionais
RF10	Manter requisitos funcionais
RF11	Disponibilizar uma tabela dos requisitos
RF12	Manter fases do projeto cadastrado
RF13	Manter os tipos de requisitos
RF14	Manter prioridade dos requisitos
RF15	Manter os estados do requisitos
RF16	Manter estabilidade do requisito
RF17	Manter tipo de implementação
RF18	Manter tipo de estado
RF19	Manter tipo de categoria
RF20	Manter a implementação
RF21	Manter a estados do caso de testes

RF22	Manter os casos de testes
RF23	Manter os Erros
RF24	Manter o estado do erros
RF25	Manter o tipo de erros
RF26	Manter o tipo de caso de uso
RF27	Manter a liberação do projeto
RF28	Emitir relatórios da tabela de rastreamento separadamente
RF29	Visualizar a matriz de rastreamento caso de uso x requisitos
RF30	Visualizar a matriz de rastreamento caso de uso e x implementação
RF31	Visualizar a matriz de rastreamento caso de uso x caso de testes
RF32	Visualizar a matriz de rastreamento caso de testes x Erros
RF33	Visualizar a matriz de rastreamento Implementações x Erros
RF34	Visualizar a matriz de rastreamento liberação em caso de uso
RF35	Visualizar a matriz de rastreamento liberação em caso de teste
RF36	Visualizar a matriz de rastreamento, erros ou liberação
RF37	Visualizar a métrica dos requisitos
RF38	Visualizar as métricas dos requisitos por mudança solicitada
RF39	Visualizar as métricas do requisito por mudança solicitada em gráficos
RF40	Visualizar as métricas do requisito por mudança solicitada em resumo
RF41	Visualizar as métricas do requisito por estados do requisitos
RF42	Visualizar as métricas do requisito por alterações proposta

RF43	Visualizar as métricas do requisito por estado ao longo do projeto
RF44	Visualizar a métrica dos caso de uso
RF45	Visualizar a métrica dos caso de uso com os tipo de caso de uso
RF46	Visualizar a métrica dos caso de uso como estados
RF47	Visualizar a métrica das implementações
RF48	Visualizar a métrica das implementações com gráfico de gantt
RF49	Visualizar a métrica das implementações com tipo de implementações
RF50	Visualizar a métrica dos testes
RF51	Visualizar métrica dos testes caso de testes / período
RF52	Visualizar a métrica dos testes com teste x liberação
RF53	Visualizar a métrica dos erros
RF54	Visualizar a métrica dos erros com descoberta de erros
RF55	Visualizar a métrica dos erros com números de erros por fase
RF56	Visualizar a métrica dos erros com estados de erros
RF57	Visualizar a métrica das liberações
RF58	Visualizar a métrica das liberações com erros por liberações
RF59	Visualizar a métrica das liberações com gráfico de gantt
RF60	Visualizar o geral do projeto
RF61	Exportar plano de projeto
RF62	Exportar os requisitos do projeto
RF63	Exportar com as descrições dos caso de uso

RF64	Emitir a simulação de modelo de priorização
RF65	Emitir a simulação de estimativa de esforço baseado em ponto de caso de uso

5.1.4 - Levantamento dos requisitos não funcional

Além dos levantamentos de requisitos funcionais, com o mesmo método da seção 5.1.3.2 *Análise do sistema existente,* é possível também identificar os requisitos não funcionais. Isso pode ser feito por meio da observação da interface gráfica, do fluxo de interações e da estrutura de dados, permitindo identificar aspectos relacionados à usabilidade, segurança e desempenho no sistema.

Também foi realizada uma análise documental com o objetivo de extrair informações adicionais para o levantamento de requisitos não funcionais. No entanto, observou-se que o documento cita mais funcionalidades do que aspectos não funcionais.

Mesmo assim, a combinação da análise do sistema existente com a análise documental permite um levantamento mais abrangente e detalhado dos requisitos, assegurando que a ferramenta atenda adequadamente às necessidades do usuário.

Com base nesse levantamento, foi possível elaborar o quadro de requisitos não funcionais (Quadro 02).

Quadro 02 - Lista de Requisitos Não Funcionais do Controla

ID	Requisitos não Funcionais
RNF01	A persistência do sistema deve ser feita através do Sistema Gerenciador de Banco de Dados da Microsoft MDB.
RNF02	A linguagem de programação utilizada é o *Borland Delphi* 7.
RNF03	Usa biblioteca para gerar PDF no *Borland Delphi* 7 *QuickPDF* ou o *PDF toolkit.*

RNF04	O sistema deve permitir a rastreabilidade entre artefatos do ciclo de desenvolvimento sem impactar significativamente o tempo de resposta.
RNF05	A ferramenta deve manter histórico de alteração nos requisitos evitando a perda de informações críticas.
RNF06	Permite a emissão de relatórios detalhados de requisitos edificações de requisitos Matriz de estabilidade plano de projeto.
RNF07	Emissão de relatórios em diversos formatos (XML e CSV).
RNF08	O *Controla* deve fornecer visualização gráfica das matrizes de rastreabilidade.
RNF09	O sistema será distribuído em versão desktop.

5.1.3 - Levantamento das Regras de Negócio

O levantamento da regra de negócio foi realizado de forma semelhante ao levantamento de requisitos funcionais e não funcionais descritos anteriormente. O processo primeiramente inclui a análise de interface gráfica. Inicialmente, realizou-se uma análise de interface, observando as funcionalidades apresentadas pelo sistema para identificar se cada uma delas possui alguma regra associada.

Além da análise de interface gráfica, também foi realizada a análise de documentos, utilizando o próprio material do autor. Nesse processo, buscou-se identificar a presença de regras de negócio que orientam e regem o funcionamento do sistema, de modo que ajuda a compreender a sua lógica interna. A partir dessa metodologia, torna-se possível estruturar o quadro (Quadro 3) Contando as regras de negócio levantadas, eu dou como cada uma delas de maneira clara e organizada, conforme a abordagem sugerida do Eduardo Bezerra em sua obra.

Quadro 03 - Lista de Requisitos Funcionais do Controla

ID	Regras

RN01	Para criar um novo projeto é necessário ter o nome do projeto, o cliente já pré-cadastrado no sistema, a data de início, a empresa pré-cadastrada no sistema e a descrição como obrigatório e nos campos, data final, metas gerais e limitações como opcional.
RN02	Para adicionar novo requisito é necessário tem pelo menos uma pessoa na equipe do projeto
RN03	Para adicionar um novo requisito é necessário selecionar o autor, estabilidade, prioridade, e estado de requisito antes de salvar
RN04	Para gerar a matriz de dependência dos requisito é necessário criar um requisito funcional e não funcional
RN05	Não é possível remover ou alterar a pessoa da equipe quando estiver dado relacionado a eles
RN06	Para adicionar um novo caso de uso tem pelo menos uma pessoa na equipe do projeto
RN07	Para adicionar um novo caso de uso e preciso de ter um a descrição e o autor selecionada para criação do caso de uso
RN08	Para adicionar um novo caso de uso a descrição não pode ser numero e nem números sequenciais e um responsável
RN09	Para gerar a matriz de rastreamento de caso de uso é requisito é necessário ter cadastrado um caso de uso e um requisitos
RN10	Para gerar a matriz de rastreamento de caso de uso é implementação é necessário ter cadastrado um caso de uso e uma implementação.
RN11	Para salvar a nova implementação é necessário selecionar a categoria e responsável o estado e o tipo de implementação
RN12	Um projeto deve conter pelo menos um membro na equipe de desenvolvimento

RN13	Requisitos devem ser categorizados como **funcionais** ou **não funcionais** no momento do cadastro.
RN14	Requisitos não podem ser removidos se já estiverem vinculados a casos de uso ou implementações

5.2 - Proposta da Ferramenta UCase para Gerência de Requisitos

A proposta da ferramenta a seguir tem como objetivo apenas mostrar mudanças de requisitos funcionais e não funcionais de mudanças de interfaces gráficas. Uma comparação entre as duas ferramentas demonstrando a eficácia da reengenharia de software em um sistema legado.

5.2.1 - Ferramentas Utilizadas

Google Docs: É uma ferramenta para criação de documentos disponível via web, permite que o usuário crie e edite documentos com outras pessoas em conjunto, armazenados na nuvem, criando e disponibilizando de forma gratuita para o usuário (Google, 2025).

O Figma é uma ferramenta de *design* que combina a acessibilidade da web com as funcionalidades de um aplicativo nativo, ou seja, a plataforma é disponibilizada pela web de qualquer navegador, sem a necessidade de instalação. A plataforma disponibiliza várias ferramentas de forma gratuita, protagonização de sistema e design de telas de cooperativas, possibilitando aos últimos usuários trabalhar simultaneamente no mesmo projeto (Figma, 2025).

O Astah é uma ferramenta que fornece meios de modelagem para diagramas de UML, ER, diagramas de fluxo de dados, fluxogramas, mapas mentais, entre outros, para indivíduos ou para equipes. A ferramenta possui a versão paga e a versão de comunidade, mas também possui a versão de estudante, que permite um ano de uso (Change Vision, 2025).

5.3 - Requisitos funcionais - Proposição

A proposta dos requisitos tem como finalidade a revisão dos requisitos funcionais para nova proposta da ferramenta do gerenciamento de requisitos. Com isso, foram identificados requisitos redundantes os, quais foram os mesclados, obsoletos que foram excluídos e novos requisitos funcionais que foram adicionados para contemplar a necessidade da versão atual do sistema.

A apresentação dos requisitos foram adicionados no (Quadro 04), Demonstra o relacionamento dos requisitos originais com a nova ferramenta indicando pelos seus status (mantido, mesclado, alterado, excluído ou novo)

Quadro 04 - Comparação dos requisitos funcionais

ID	Descrição	Novo ID	Nova Descrição	Status
-	-	RF01	Manter contas	Novo
RF01	Manter(A persistência de dados no sistema) usuário	RF02	Manter usuário	Mesclado
RF02	Manter projeto	RF03	Manter projeto	Manter
RF03	Manter empresas	RF04	Manter usuário	Mesclado
RF04	Manter cliente	RF05	Manter usuário	Mesclado
RF05	Manter os usuários do cliente	RF06	Manter os usuários do cliente	Excluído
RF06	Manter Pessoa da equipe	RF07	Manter usuário da equipe	Mesclado
RF07	Manter funções das pessoas da equipe	RF08	Manter usuário da equipe	Mesclado
RF08	Manter expectativa	RF09	Manter projeto	Mesclado

	do usuário			
RF09	Manter requisitos não funcionais	RF10	Manter requisitos	Mesclado
RF10	Manter requisitos funcionais	RF11	Manter requisitos	Mesclado
RF11	Disponibilizar uma tabela de dependência de requisitos	RF12	Disponibilizar uma tabela de dependência de requisitos	Manter
RF12	Manter fases do projeto cadastrado	RF13	Manter projeto	Mesclado
RF13	Manter os tipos de requisitos	RF14	Manter requisitos	Mesclado
RF14	Manter prioridade dos requisitos	RF15	Manter requisitos	Mesclado
RF15	Manter os estados do requisitos	RF16	Manter requisitos	Mesclado
RF16	Manter estabilidade do requisito	RF17	Manter requisitos	Excluído
RF17	Manter os tipos de caso de uso	RF18	Manter caso de uso	Alterado
RF18	Emitir relatórios das tabelas de rastreamento separadamente	RF19	Emitir relatórios das tabelas de rastreamento separadamente	Manter
RF19	Visualizar a matriz de	RF20	Visualizar a matriz de	Manter

	rastreamento caso de uso x requisitos		rastreamento caso de uso x requisitos	
RF20	Visualizar a matriz de dependência entre requisitos	RF21	Visualizar a matriz de dependência entre requisitos	Manter
RF21	Visualizar as métricas do requisitos por mudança solicitada em resumo	RF22	Manter mudanças solicitadas pelos os stakeholder	alterado
RF22	Visualizar o geral do projeto	RF23	Visualizar o geral do projeto	Manter
RF23	Exportar para PDF o plano de projeto	RF24	deve gerar um PDF com informação geral do projeto	alterado
RF24	Exportar para PDF com os requisitos do projeto	RF25	deve gerar PDF com os requisitos funcionais e não funcionais.	Manter
RF25	Exportar para PDF com as descrições dos caso de uso	RF26	Deve gerar PDF com as informações dos casos de uso simples.	Manter
		RF27	Deve gerar PDF com as informações dos casos de uso com todos os fluxos.	Novo
RF26	Manter o de caso de uso	RF28	Manter o de caso de uso	Manter

RF27	Manter fluxos de caso de uso principal	RF29	Manter fluxos de caso de uso	Alterar
RF28	Manter fluxos de caso de uso alternativos	RF30	Manter fluxos de caso de uso	Alterar
RF29	Manter fluxos de caso de uso Erros	RF31	Manter fluxos de caso de uso	Alterar
		RF32	Integração com API de assinatura eletrônica	Novo
		RF33	Importar planilhas de caso de uso não detalhados	Novo
		RF34	Importar planilhas de requisitos funcionais e não funcionais	Novo
		RF35	Adicionar diagramas de caso de uso	Novo
		RF36	Realizar login com GitHub	Novo
		RF37	Realizar login com Google	Novo
		RF38	Realizar login com o GitLab	Novo

5.4 - Requisitos não funcionais - Proposição

A propósito tem como finalidade a revisão dos requisitos não funcionais para nova versão da ferramenta de gerenciamento de requisitos. Tem como objetivo a revisão dos requisitos de forma semelhante à aplicação aos requisitos funcionais.

Porém com o foco na atualização das tecnologias, substituindo os requisitos originais da ferramenta controla com os novos requisitos não funcionais mais atuais atendendo a tecnologias moderna conforme demonstrado no (Quadro 05).

A nova proposta de alterar a arquitetura para um *API* (Interface de Programação de Aplicações) baseada no estilo arquitetural *REST* (*Representational State Transfer*). A arquitetura permite comunicação entre diferentes aplicações por meio do protocolo HTTP (*Hypertext Transfer Protocol* – Protocolo de Transferência de Hipertexto), o mesmo utilizado em navegação de sites.

Quadro 05 - Comparativa entre a Ferramenta Controle e a nova Ferramenta.

ID	Descrição Controla	Nova Descrição	
RNF01	A persistência do sistema deve ser feita através do Sistema Gerenciador de Banco de Dados da microsoft MDB	A persistência do sistema deve ser feita através do Sistema Gerenciador de Banco de Dados postgresql	Alterado
RNF02	A linguagem de programação utilizada foi Borland Delphi 7	A linguagem de programação utilizada deve ser DotNet versão 8	Alterado
RNF04		Criar um api (Interface de Programação de Aplicação)	Incluído
RNF05	Usa biblioteca para gerar PDF no Borland Delphi 7	Usar a biblioteca PDFkit para gerar PDF no nodejs	Alterado
RNF06		Criar interfaces gráficas em react,	Incluído
RNF07		Usar conexão https para se	Incluído

		conectar com a api	
RNF08		Cada conta pode fazer parte de vários projeto	Incluído
RNF09		Uma conta só pode pertencer a um usuário.	Incluído
RNF10		o sistema deve permitir formas alternativas de login como alternativa para o login tradicional por meio e senha.	Incluído
RNF11		Conectar os caso de uso com um kanban	Incluído
RNF12		Apenas administradores podem alterar os projetos das equipes.	Incluído
RNF13		Poder escolher o que vai está no relatório exportado	Incluído
RNF14		Qualquer mudança vai ter que passar pela assinatura dos administradores do projeto	Incluído
RNF15		O time/equipe pode ter vários usuários (stakeholders).	Incluído
RNF16		o projeto criado em times os administradores poderão escolher quem os membros que têm acesso nos projetos. e só podem editar ou apenas visualizar	Incluído
RNF17		O projeto pode ter vários usuários.	Incluído

RNF18		Você pode selecionar usuário para convidar para o projeto.	Incluído

5.5 - Lista de Casos de Uso - Proposição

Diferente do da lista de requisitos funcionais e não funcionais A apresentada anteriormente, O (Quadro 06) demonstra o referenciamento entre caso de uso e os requisitos funcionais. Esse mapeamento tem como finalidade de simplificar a identificação das funcionalidade do sistema relacionamento em cada caso de uso aos requisitos funcionais correspondentes, dessa forma podemos garantir a rastreabilidade mais clara dos requisitos levantados e as funcionalidades.

Quadro 06 - Lista dos casos de uso com referência com os requisitos funcionais.

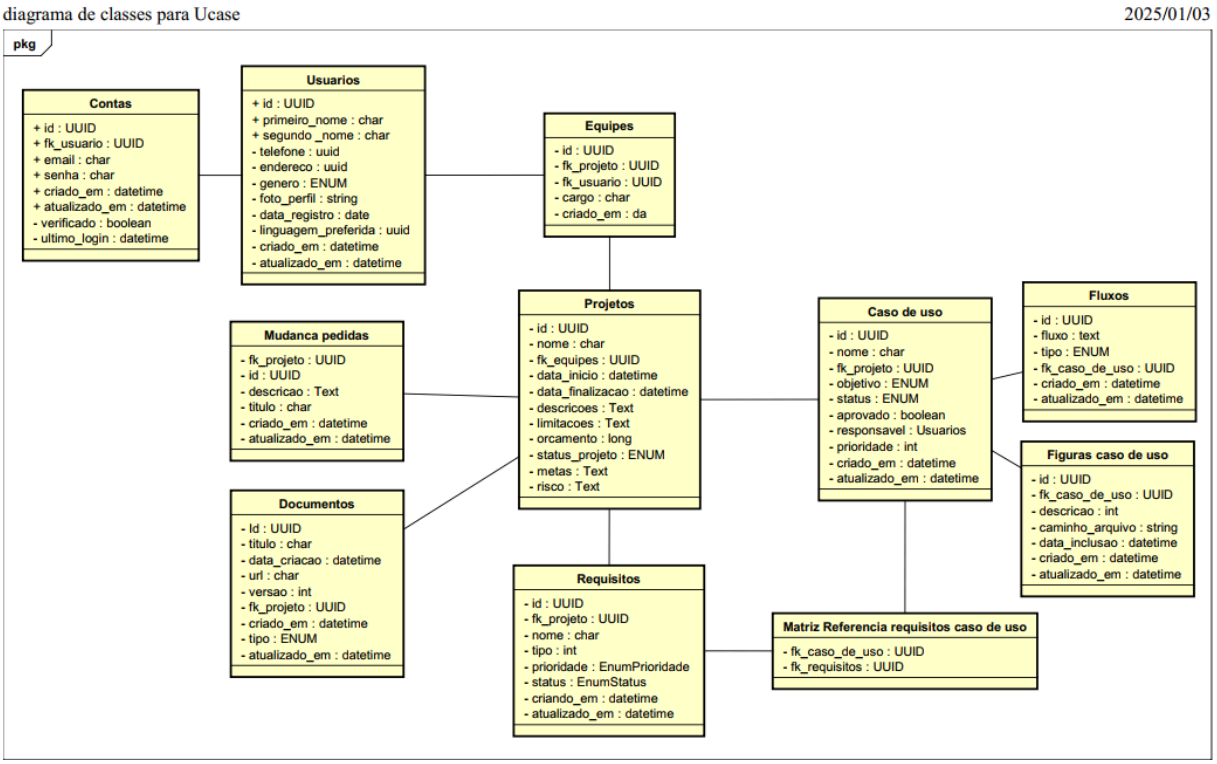
ID	Descrição	ID Referência
UC 01	Manter projeto	RF01, RF03, RF09, RF13
UC 02	Manter usuários	RF02, RF04, RF05, RF06, RF07, RF08
UC 03	Manter usuários no projeto	RF05
UC 04	Emitir relatórios gerais.	RF23, RF24, RF25, RF26, RF27
UC 05	Assinar documentos	RF23, RF24, RF25, RF26, RF27
UC 06	Emitir documentos assinados	RF23, RF24, RF25, RF26, RF27
UC 07	Manter requisitos funcionais	RF10, RF11, RF14, RF15, RF16, RF17
UC 08	Manter requisitos não funcionais	RF14, RF15, RF16, RF17
UC 09	Manter caso de usos	RF18

UC 10	Manter fluxos de caso de uso	RF28
UC 11	Visualizar tabelas de Requisitos	RF12
UC 12	Visualizar a tabela de caso de uso	RF28
UC 13	Visualizar matriz de requisito com requisito não funcional	RF19
UC 14	Visualizar matriz de requisitos com caso de uso	RF19, RF20
UC 15	Emitir informação geral do projeto	RF22
UC 16	Manter diagramas de caso de uso	RF28

5.6 - Diagrama de Classe

O diagrama de classe é uma ferramenta UML usada para representar aspectos estáticos de um sistema orientado a objeto (Bezerra, 2015). O diagrama na (Figura 3) ilustra como funciona a relação das entidades, tanto em banco de dados como em classe no desenvolvimento do sistema, mostrando a os atributos e e o tipo de cada atributo, inicialmente, foram desenvolvidas 11 classe, garantindo que, em um desenvolvimento futuro, seja possível expandir o sistema, mas já cobrindo todas as funcionalidades básicas necessárias.

Figura 3 - Diagrama de classes para Ucase.



Fonte: Elaborado pelo autor.

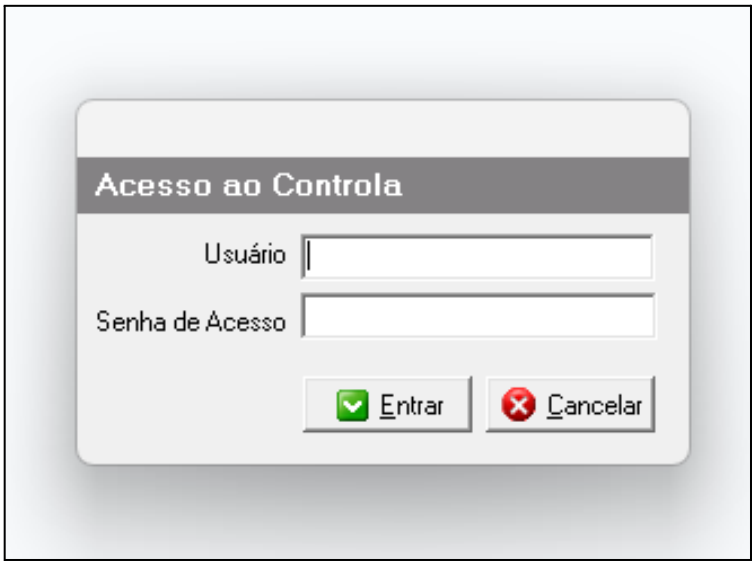
5.7 - Protótipo das novas Telas

Nessa seção, é apresentada a comparação entre as telas do originais do sistema Controla com as novas interfaces proposta, mostrando as melhorias das interface gráficas de cinco telas do sistema como: tela de login, tela de cadastro de caso de uso, tela de relatório, tela de cadastro de novos projetos, tela de projeto salvos.

5.7.1 Tela de Login

A interface original apresenta um visual simples, com a caixa de texto usuário e senha, dois botões de ações de um interface de caixa com predominância de tons cinza, uma característica comum em software da década de 2000 (Figura 4).

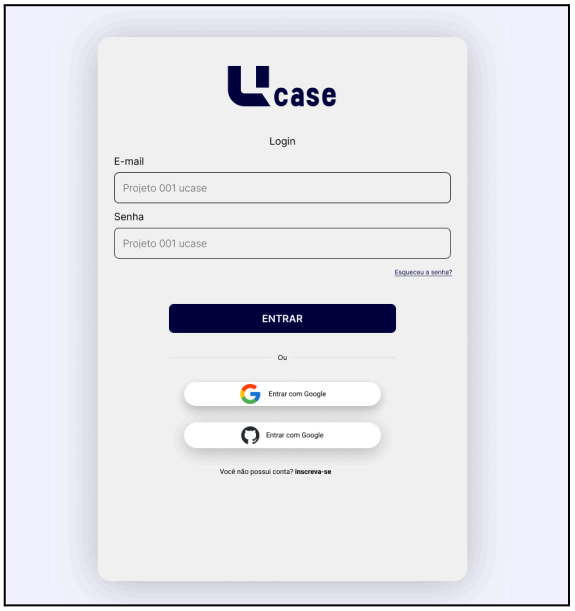
Figura 4 - Interface de login do Controla



Fonte: Própria do autor (Fraga Filho & Reis, 2005).

Na nova proposta há uma modernização das interfaces, adotando o campo mais usuais, como e-mail e senha, tendo outra possibilidade de autenticação alternativa (ex.: *Google, Github*). A interface utiliza cores leves priorizando acessibilidade destacando botões de acesso principais (Figura 5).

Figura 5 - Interface de login.

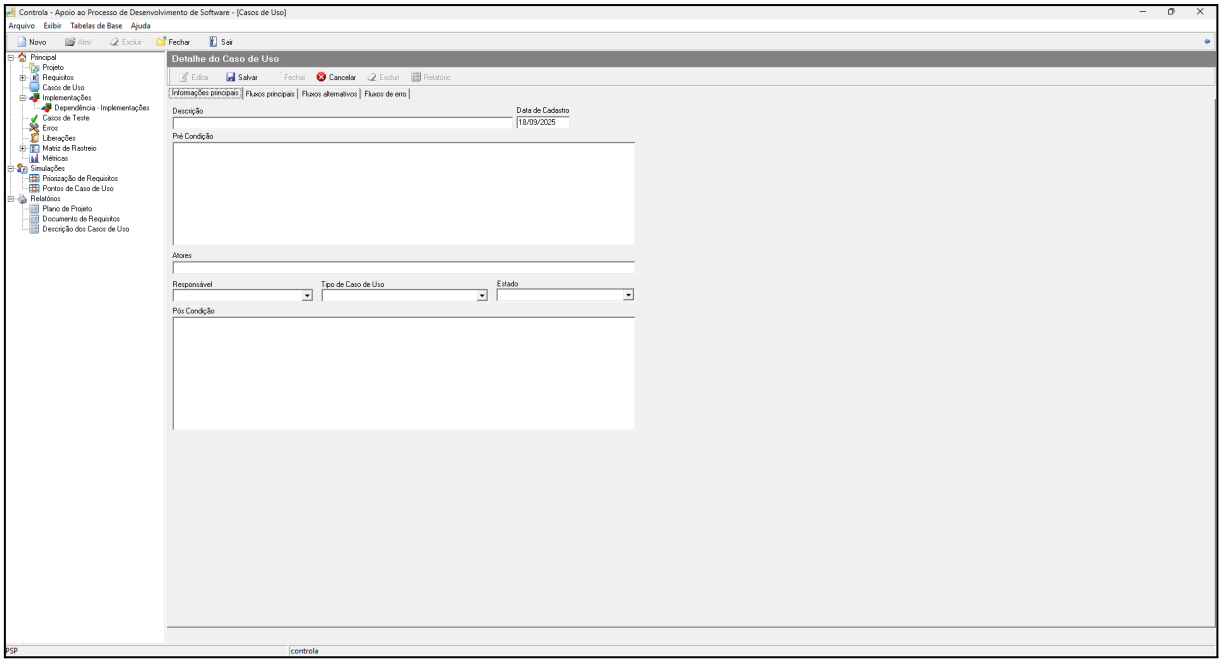


Fonte: Produção Própria.

5.7.2 Interface de cadastro de caso de uso

Na interface original é uma interface um pouco mais dados e têm as mesmas características de tons acinzentados com a pegada de software de característica de 2000 com poucos recursos visuais (Figura 6).

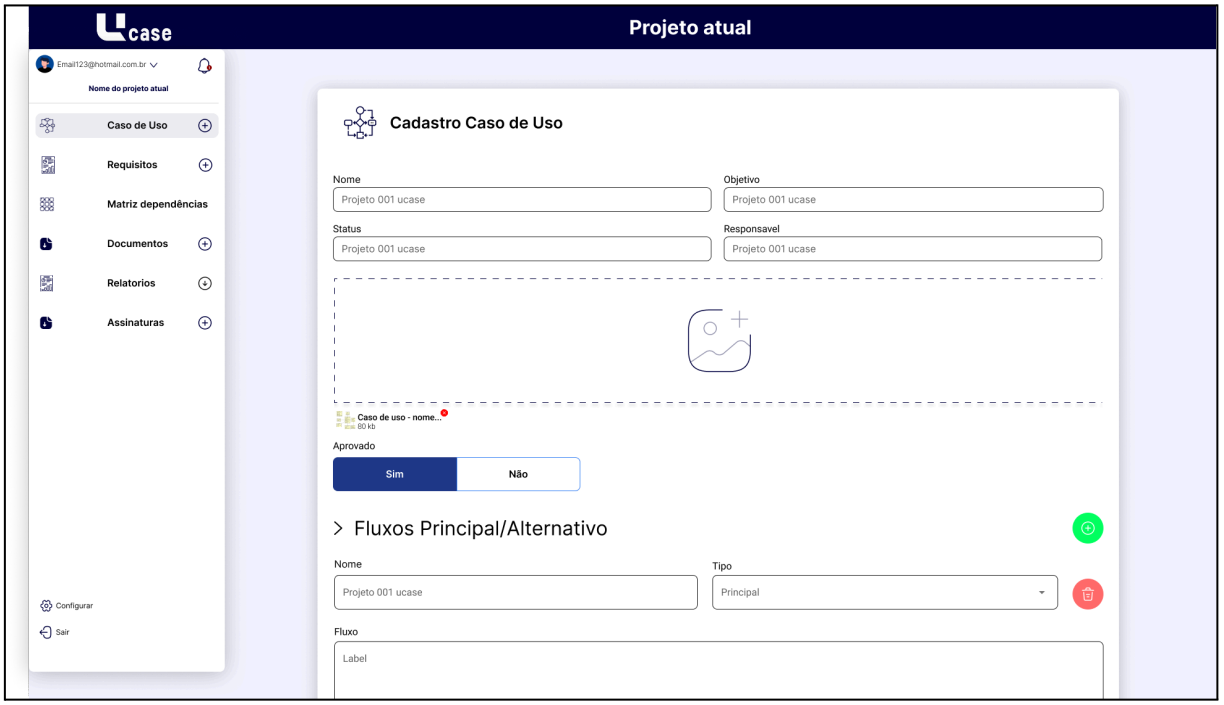
Figura 6 - Interface de cadastro de caso de uso do Controla



Fonte: Própria do autor (Fraga Filho & Reis, 2005).

Na nova proposta organiza as Interface por blocos e divisões eu já bem mais definidas, usa as mesmas cores como padrão (branco e azul), utiliza botões com com cores semântico, utiliza ícones para guiar o usuário, inclui funções adicionais, possível cadastrar a imagens, e então cadastro de fluxo alternativo na mesma página (Figura 7).

Figura 7 - Interface de cadastro de um novo projeto.

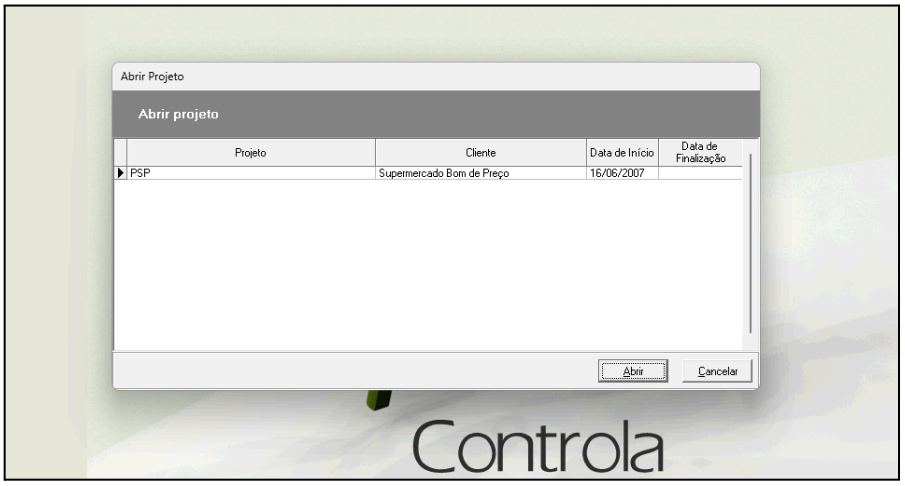


Fonte: Produção Própria.

5.7.3 Interface de Projetos salvos

A versão original, representa o modal no meio da tela com tonalidades cinzas assim como representada anteriormente sobre as outras telas um formato de tabela textual onde tem pouca visualização gráfica muito baseado naquela época de 2000 (Figura 8).

Figura 8 - Interface de projetos salvo no Controla



Fonte: Própria do autor (Fraga Filho & Reis, 2005).

Na nova proposta da interface os projetos são exibidos em cartões visíveis visuais contendo informações resumida facilitando a identificação de cada relatório. Além disso, a interface conta com uma barra lateral de acesso rápido e de menu principal o que permite ao usuário navegar diretamente para relatórios e as demais funcionalidades (Figura 9).

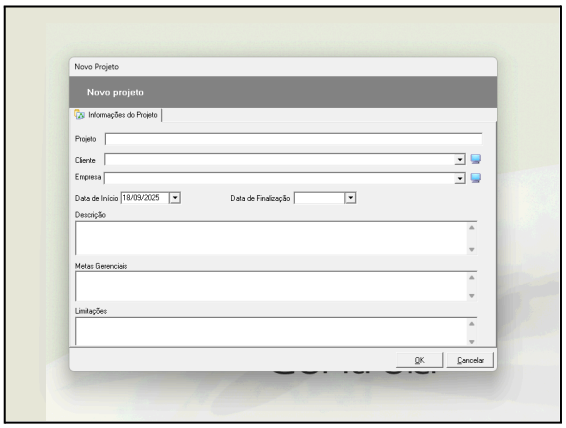
Figura 9 - Interface de listagem do projeto recentes.

Fonte: Produção Própria.

5.7.4 Interface de Novos projetos

A interface antiga segue o padrão rígido, como anteriormente, um campo organizado por lista em verticais dentro de um modal utilizando o mesmo padrão de todas as telas anteriores que foram citadas (Figura 10).

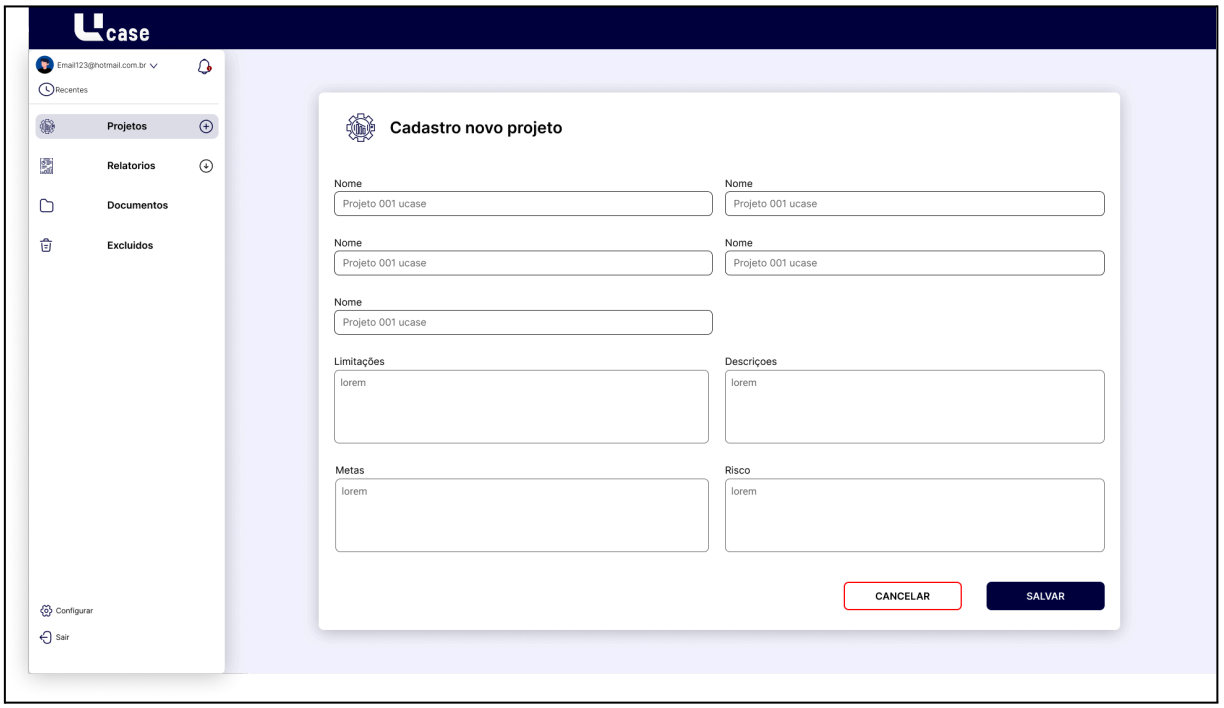
Figura 10 - Interface de cadastro de um novo projeto Controla



Fonte: Própria do autor (Fraga Filho & Reis, 2005).

Na proposta atual, removemos o modal e organizamos os campos, os mesmo que tem na antiga interface mantendo o formulário em *card* (cartão) que fica centralizado e seus botões de ações destacados (Figura 11).

Figura 11 - Interface de cadastro de um novo projeto.

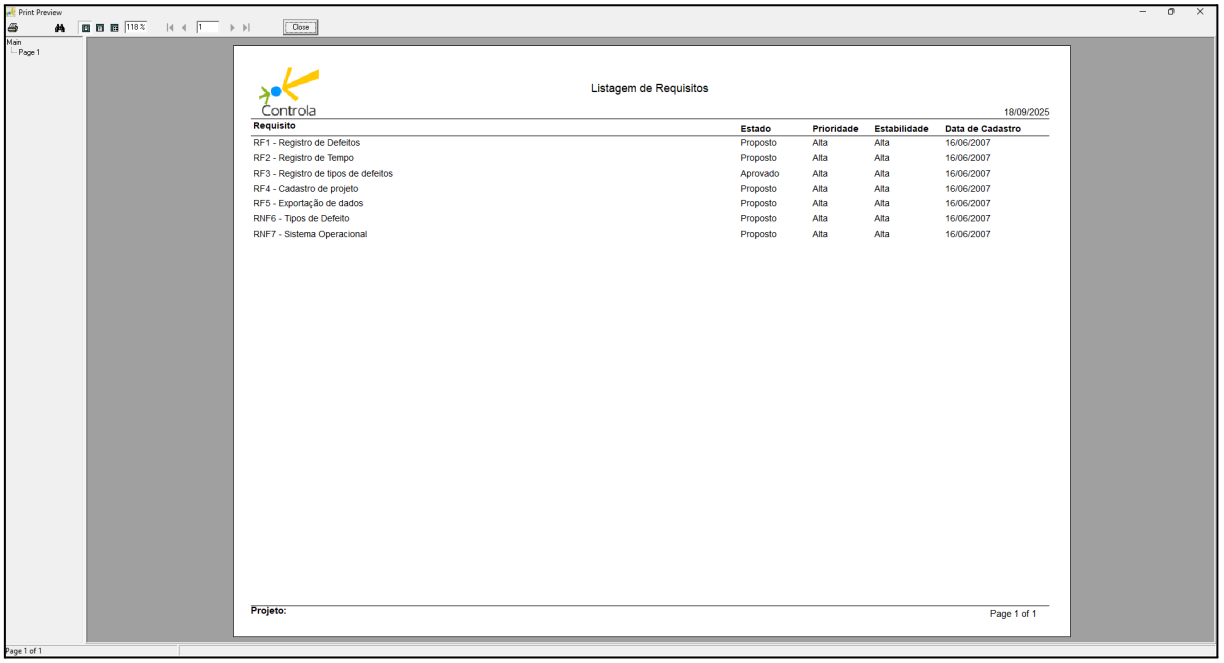


Fonte: Produção Própria.

5.7.5 Interface de Relatórios

Na interface original, o relatório apresenta uma forma simples e estática, lembrando saída de uma impressão, com listagens em tabelas. Vale ressaltar que esse relatório no Controla é separado por funcionalidades e cada função você pode gerar um relatório (Figura 12).

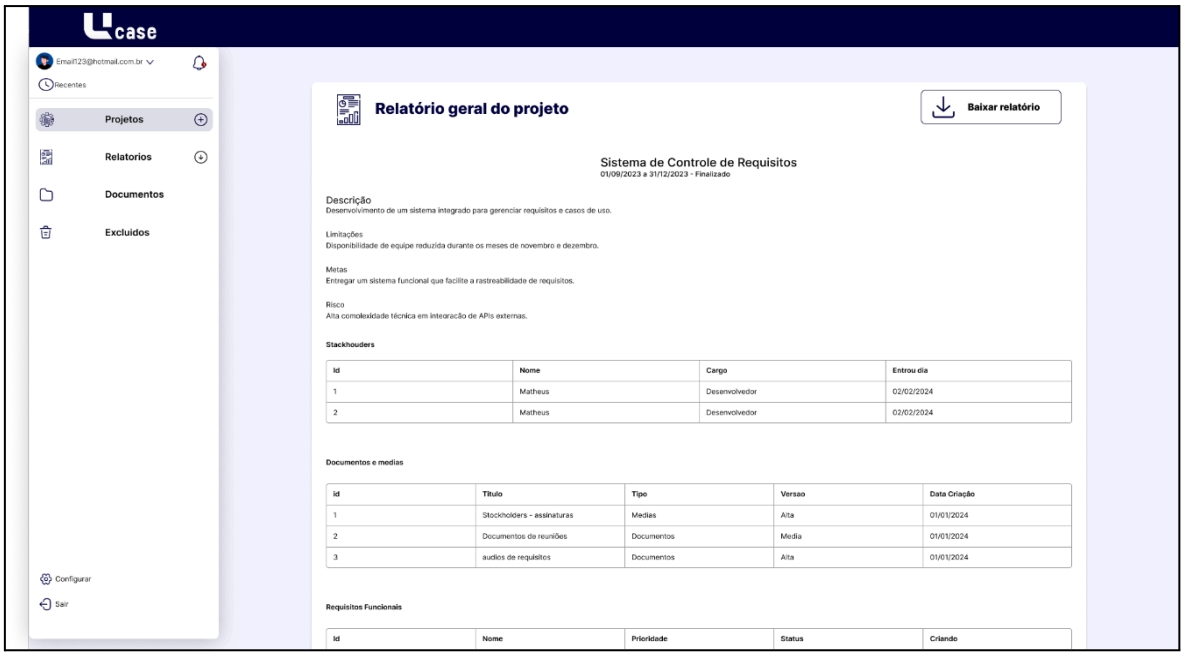
Figura 12 - Interface de relatório de requisitos do Controla



Fonte: Própria do autor (Fraga Filho & Reis, 2005).

A versão proposta, a Interface de relatório foi redesenhada para ser mais personalizada. O usuário seleciona brevemente quais informações deseja incluir, como matrizes de rastreabilidade, requisitos funcionais, diagramas de casos de uso, entre outros. Mantivemos o formato de lista de tabelas para a organização dos dados e cada relatório é um espaço reservado para assinaturas dos *stakeholders* envolvidos no projeto, conferindo formalidade e respaldo documental aos registros (Figura 13).

Figura 13 - Interface de relatório do projeto.



Fonte: Produção Própria.

6 CONCLUSÃO

O trabalho aborda a proposta da reengenharia reversa *com o* foco na reestruturação a um *software* legado (Controla v10) – originalmente desenvolvido pelo o Clayton Vieira Fraga Filho em seu bacharelado, no ano de 2005 – o trabalho demonstra como é possível revitalizar um sistema por meio remodelação, aplicando técnicas de engenharia de requisitos e de reengenharia de software.

A análise documental, a observação prática das interfaces e o estudo dos fluxos de interação permitiram o levantamento de requisitos funcionais não funcionais e das regras de negócios, com as técnicas utilizadas foi possível identificar uma redundância de requisito que podia ser mesclado ou até excluído do sistema sem comprometer o funcionamento na nova proposta da ferramenta.

Além disso, a reengenharia demonstra uma abordagem eficaz a remodelação do sistema legado, permitindo incorporar novas tecnologias e práticas atuais no desenvolvimento, melhorando aspectos essenciais, como estabilidade, usabilidade e manutenção.

Para futuros trabalhos recomendam a prática de implementação de protótipos utilizando a documentação levantada no trabalho, a proposta é desenvolver ao longo do tempo um protótipo funcional destinado a uso da instituição permitindo validar as melhorias e inovação incorporada no sistema.

REFERÊNCIAS

ASTAH. **Premier Diagramming, Modeling Software & Tools.** Disponível em: <https://astah.net>: Acesso em: 08 mar. 2025.

BEZERRA, Eduardo. **Princípios de análise e projeto de sistemas com UML**. 2. ed. Rio de Janeiro, RJ: Elsevier, 2015. 398 p.

EILAM, Eldad. ***Reversing: Secrets of Reverse Engineering***. Indianapolis: Wiley Publishing, 2005.

FIGMA. **A ferramenta de design de interface colaborativa e online**. Disponível em: <https://www.figma.com/pt-br/>: Acesso em: 08 mar. 2025.

FRAGA FILHO, Clayton Vieira. **Instalador do Controla v. 10 Free**. Dropbox, 2010. Disponível em: <https://www.dropbox.com/scl/fi/gnice8hmjwuyhafiya0f2/InstallControla10Free.zip?rlkey=v92gginy7wib1xh1jhcda6f11&e=3&st=f48h1wo5&dl=0>. Acesso em: 05 de fevereiro de 2025.

FRAGA FILHO, Clayton Vieira; REIS, José Maurício dos**. Controla: ferramenta de apoio ao processo de desenvolvimento de software em pequenas empresas**. Faculdade de Viçosa, Viçosa-MG, Brasil, 2005. Disponível em: <https://www.periodicosibepes.org.br/index.php/reinfo/article/view/156>. Acesso em: 26 jan. 2025.

GOOGLE. **Google Docs**. Disponível em: https://docs.google.com/. Acesso em: 08 mar. 2025.

Ellen Gottesdiener. 2001. **Rules rule: business rules as requirements. Beyond chaos: the expert edge in managing software development.** Association for Computing Machinery, New York, NY, USA, 219–225. https://doi.org/10.1145/379391.379419: Acesso em: 06 de fevereiro de 2025.

GUEDES, Gilleanes T. A. **UML 2: uma abordagem prática.** São Paulo: Novatec, 2007.

IBM. **Rational RequisitePro 7.1.4.** 2018. Disponível em: https://www.ibm.com/support/pages/rational-requisitepro-714. Acesso em: 05 de fevereiro de 2025.

IIBA. **INTERNATIONAL INSTITUTE OF BUSINESS ANALYSIS**. A Guide to the Business Analysis Body of Knowledge (BABOK Guide). 3. ed. Toronto: IIBA, 2015.

MENS, Tom; TOURWE, Tom. **A survey of software refactoring**, in IEEE Transactions on Software Engineering, vol. 30, no. 2, pp. 126-139, Feb. 2004, doi: 10.1109/TSE.2004.1265817.

PERFORCE. **HELIX-ALM.** 2025. Disponível em: https://www.perforce.com/products/helix-alm. Acesso em: 05 de fevereiro de 2025.

SOMMERVILLE, Ian. **Engenharia de software.** 10. ed. São Paulo, SP: Pearson Prentice Hall, 2018. 756 p.

VISURE. **Visure ALR Requisitos**. 2024. Disponível em: https://visuresolutions.com/pt/. Acesso em: 05 de fevereiro de 2025.

WIEGERS, Karl; BEATTY, Joy. ***Software Requirements***. 3. ed. Redmond: Microsoft Press, 2013.