



UNIVERSIDADE TÉCNICA DE ANGOLA
FACULDADE DE ENGENHARIAS
DEPARTAMENTO DE ENSINO E INVESTIGAÇÃO DAS
TECNOLOGIAS DE INFORMAÇÃO E COMUNICAÇÃO DEITIC

Sistema de Gestão de Trabalho de Fim de Curso

ADILSON MENDONÇA DA SILVA

LUANDA-ANGOLA

MAIO/2024

UNIVERSIDADE TÉCNICA DE ANGOLA
FACULDADE DE ENGENHARIAS
DEPARTAMENTO DE ENSINO E INVESTIGAÇÃO DAS
TECNOLOGIAS DE INFORMAÇÃO E COMUNICAÇÃO DEITIC

CURSO DE ENGENHARIA INFORMÁTICA

Sistema de Gestão de Trabalho de Fim de Curso

ADILSON MENDONÇA DA SILVA

*Trabalho de Fim de curso apresentado a
Universidade Técnica de Angola, como
requisito para obtenção do título de
licenciado em Engenharia Informática.*
Orientador: Eng. Kwenda Mayeye

LUANDA-ANGOLA

MAIO/2024

ADILSON MENDONÇA DA SILVA

Sistema de Gestão de Trabalho de Fim de Curso

Aprovado: ____/____/____

BANCA EXAMINADORA

Presidente do Júri..... Assinatura.....

Primeiro Vogal..... Assinatura.....

Segundo Vogal..... Assinatura.....

DEDICATÓRIA

A minha mãe “*Bibiana Domingos Adão Mateus*” por mostrar que apenas acreditando e trabalhando conseguimos o sucesso na vida.

A meus irmãos que não têm medido esforços para ajudar de forma directa ou indirecta.

Aos meus amigos...

AGRADECIMENTO

À Deus o Altíssimo – pois sem ele eu não teria capacidade para desenvolver este trabalho.

A minha família por seu inquebrável apoio e compreensão durante todo este processo. Vossa força e ânimo foram cruciais para me manter motivado focado no alcance de minhas metas acadêmicas.

Além disso, desejo agradecer a meus amigos e companheiros por suas palavras revigorantes e por compartilharem seus conhecimentos e experiências, o que enriqueceu significativamente este trabalho.

A todos os professores que de forma directa contribuíram com a sua sabedoria para a consolidação da minha formação.

Aos meus irmãos queridos que com muita paciência apoiaram-me incondicionalmente em toda a minha trajetória até chegar aqui.

Aos meus colegas, em especial a turma EIN4.1, pelo apoio que me foi fornecido nessa dura e longa caminhada!

A todos que de forma directa ou indirecta contribuíram para o sucesso deste trabalho.

Muito obrigada...

EPÍGRAFE

*“A educação é a arma mais poderosa que você pode usar
para mudar o mundo.”*

Nelson Mandela

RESUMO

Atualmente, muitas empresas estão reconhecendo a importância dos softwares de gestão para otimizar suas operações. A necessidade de implementar sistemas de gestão é cada vez mais evidente, impulsionada pelo aumento constante das expectativas em relação à qualidade dessas soluções.

Este trabalho tem como objectivo criar um software de gestão de Trabalhos de Fim de Curso(TFC) para Universidade Técnica de Angola(UTANGA), pois o mesmo será uma ferramenta para facilitar a organização e acompanhamento das etapas do TFC, desde a escolha até a apresentação final, promover a comunicação eficiente entre estudantes e orientadores, possibilitando feedbacks regulares e esclarecimentos de dúvidas, monitorar e alertar sobre os prazos importantes, como datas de entregas de capítulos, revisões e apresentações, proporcionar um meio para acompanhar o progresso do TFC, tanto para estudantes quanto orientadores garantindo o avanço contínuo.

Portanto, o sistema de gestão de TFC visa proporcionar uma experiência mais eficiente, organizada e colaborativa para todos os envolvidos no processo de elaboração e avaliação de trabalhos de fim de curso.

Palavras-chave: Software de Gestão, TFC, Web Service.

ABSTRACT

Currently, many companies are recognizing the importance of management software to optimize their operations. The need to implement management systems is becoming increasingly evident, driven by the constant rise in expectations regarding the quality of these solutions.

This project aims to create a management software for Final Course Papers (FCP) for the Technical University of Angola (UTANGA), as it will serve as a tool to facilitate the organization and monitoring of FCP stages, from selection to final presentation, promote efficient communication between students and supervisors, enabling regular feedback and clarification of doubts, monitor and alert about important deadlines, such as chapter submission dates, revisions, and presentations, provide a means to track FCP progress for both students and supervisors, ensuring continuous advancement.

Therefore, the FCP management system aims to provide a more efficient, organized, and collaborative experience for all involved in the process of elaborating and evaluating final course papers..

Key-Words: Management Software, FCP, Web Service.

ÍNDICE

| | |
|------------------------------------------------|-----------|
| DEDICATÓRIA | i |
| AGRADECIMENTO | ii |
| EPÍGRAFE..... | iii |
| RESUMO | iv |
| ABSTRACT | v |
| ÍNDICE DE FIGURAS | ix |
| ÍNDICE DE TABELAS | x |
| 1. INTRODUÇÃO | 1 |
| 1.1 Definição do problema..... | 2 |
| 1.2 Hipótese..... | 2 |
| 1.3 Justificativa..... | 2 |
| 1.4 Objectivos | 3 |
| 1.4.1 Objectivo Geral | 3 |
| 1.4.2 Objectivos específicos..... | 3 |
| 1.5 Organização do trabalho..... | 3 |
| 2. FUNDAMENTAÇÃO TEÓRICA | 5 |
| 2.1 Web Services | 5 |
| 2.1.1 Definição e Conceitos Fundamentais | 5 |
| 2.1.2 Funcionamento e Arquitetura..... | 5 |
| 2.1.3 Vantagens e Benefícios | 5 |
| 2.1.4 Aplicações e Casos de Uso | 6 |
| 2.2 Sistema de Gestão | 6 |
| 2.2.1 Vantagens de um sistema de gestão | 7 |
| 2.2.2 Sistemas de Gestão de TFCs..... | 7 |
| 2.3 Engenharia de Software | 10 |
| 2.3.1 Requisitos do Sistema..... | 10 |
| 2.3.2 Modelação de Sistemas | 11 |
| 2.4 Base de Dados..... | 12 |

| | | |
|--------|------------------------------------------------------------------------|----|
| 2.4.1 | Sistemas de base de dados | 13 |
| 2.4.2 | Modelo Entidade Relacionamento (MER) | 13 |
| 2.4.3 | SQL..... | 14 |
| 2.5 | Processo de desenvolvimento de sistemas | 15 |
| 2.5.1 | Processo Incremental | 17 |
| 2.6 | Qualidade de Software | 18 |
| 2.7 | Arquitetura de Software..... | 19 |
| 2.7.1 | Linguagem de Programação..... | 19 |
| 2.7.2 | Linguagem C#..... | 20 |
| 2.7.3 | Programação em camadas | 20 |
| 2.8 | Linguagens e Tecnologias Web..... | 21 |
| 2.8.1 | ASP.NET | 22 |
| 2.8.2 | CSS | 22 |
| 3. | METODOLOGIA | 24 |
| 3.1 | Metodologia de Investigação Científica (Metodologia de Pesquisa) | 24 |
| 3.2 | Descrição do Campo de estudo | 24 |
| 3.3 | Delimitação do Estudo..... | 25 |
| 3.4 | Processo de Desenvolvimento | 25 |
| 3.5 | Análise de Requisitos | 26 |
| 3.5.1 | Requisitos Funcionais | 27 |
| 3.5.2 | Requisitos não funcionais | 29 |
| 3.5.3 | Regras de negócio do nosso sistema..... | 29 |
| 3.6 | Diagramas UML do nosso Sistema..... | 30 |
| 3.6.1 | Diagrama de Casos de Uso | 30 |
| 3.6.2 | Descrição dos principais casos de uso do nosso sistema..... | 32 |
| 3.6.3 | Diagrama de Actividades..... | 40 |
| 3.6.4 | Diagrama de Sequência | 43 |
| 3.6.5 | Diagrama de Classe | 46 |
| 3.5.7. | Diagrama de Instalação | 48 |
| 3.5.8. | Modelo conceitual..... | 48 |
| 3.5.9. | Modelo lógico | 50 |

| | | |
|-------|-----------------------------------------------|----|
| 3.6. | Ferramentas utilizadas | 52 |
| 4. | RESULTADOS | 54 |
| 4.1 | Descrição do projecto | 54 |
| 4.2 | Interface do Sistema | 54 |
| 4.2.1 | Página Inicial do nosso sistema | 54 |
| 4.2.2 | Página de Cadastro de Proposta | 55 |
| 4.2.3 | Página de Actividades | 56 |
| 4.2.4 | Página de Login | 56 |
| 4.2.5 | Página de Cadastro de Área Investigação | 57 |
| 5 | CONCLUSÃO | 58 |
| 6 | RECOMENDAÇÕES | 59 |
| 7 | REFERÊNCIAS BIBLIOGRÁFICAS | 60 |

ÍNDICE DE FIGURAS

| | |
|----------------------------------------------------------------------------------|----|
| Figura 2. 1 O SIGAA (Sistema Integrado de Gestão de Atividades Acadêmicas) | 8 |
| Figura 2. 2 Sistemas de Gestão TIDIA-AE(Ambiente de Ensino)..... | 9 |
| Figura 2. 3 Processo Incremental | 17 |
| Figura 2. 4 Arquitetura de Software | 19 |
| | |
| Figura 3. 1 Diagrama de Caso de Uso do projecto. | 31 |
| Figura 3. 2 Diagrama de Actividade Área de Investigação | 41 |
| Figura 3. 3 Diagrama de Actividade Enviar Proposta de TFC | 42 |
| Figura 3. 4 Diagrama de Sequência Enviar Proposta de TFC. | 44 |
| Figura 3. 5 Diagrama de Sequência Visualizar Tema. | 45 |
| Figura 3. 6 Diagrama de Classe..... | 47 |
| Figura 3. 7 Diagrama de instalação do sistema. | 48 |
| Figura 3. 8 Modelo Conceitual | 49 |
| Figura 3. 9 Diagrama Lógico..... | 51 |
| | |
| Figura 4. 1 Página inicial | 55 |
| Figura 4. 2 Cadastro de Proposta..... | 55 |
| Figura 4. 3 Visualizar Tarefas | 56 |
| Figura 4. 4 Login | 57 |
| Figura 4. 5 Cadastro Área de Conhecimento..... | 57 |

ÍNDICE DE TABELAS

| | |
|---------------------------------------------|----|
| Tabela 3. 1 Requisitos Funcionais..... | 28 |
| Tabela 3. 2 Requisitos Não Funcionais | 29 |
| Tabela 3. 3 Regras de Negócio..... | 30 |

1. INTRODUÇÃO

O crescimento e aperfeiçoamento da tecnologia fez com que nos últimos tempos ela se tornasse constantemente presente no cotidiano das pessoas, desde o acordar com o despertador até o enviar uma mensagem online. Com o avanço tecnológico o que não se realizava na rede, se tornou seguro e habitual para as pessoas como movimentações bancárias, compras em sites e aplicativos, agendamento de serviços e armazenamento de arquivos em servidores online, por exemplo. Antes, muitas atividades exigiam que as pessoas se deslocassem para um determinado local e aguardassem pelo atendimento dos funcionários, o que demandava grande quantidade de tempo. A automatização dessas atividades gerou comodidade e agilidade para que a população pudesse obter maior rendimento na rotina, pois operações que demandavam tempo passaram a ser feitas em poucos instantes com o uso de softwares (KOHN, MORAES, 2007).

O TFC, que significa Trabalho de Fim de Curso, é uma avaliação que acontece quando a licenciatura está chegando ao fim. Ele tem o objetivo de fazer com que o aluno demonstre o que aprendeu, desde o início dos estudos mediante um trabalho. Sendo uma parte muito importante de vários cursos, o aluno se esforça para fazê-lo da maneira correta, pois serve para aferir o conhecimento e vê por outro ângulo como foi a própria evolução durante todo o processo de aprendizagem. Desta forma, é possível saber se tudo foi absorvido e entendido corretamente.

A elaboração do Trabalho de Fim de Curso (TFC) na Universidade Técnica de Angola (UTANGA) é um processo comum para os alunos que estão prestes a terminar a sua licenciatura e que queiram obter o diploma. Este processo embora sendo pessoal, envolve geralmente os mesmos elementos: coordenador, orientador, aluno, trabalho proposto e concretizado e uma mesa de júris.

Actualmente, todo o processo de acompanhamento dos trabalhos de fim de curso é feito manualmente, através de formulários e relatórios preenchidos pelos alunos, orientadores e coordenador do TFC. Algumas informações consideradas importantes são postadas no grupo de whatsapp criado para o efeito ou são enviados por email. Essa sistemática de trabalho acaba ocasionando alguns problemas, dentre eles a descentralização das informações e o grande volume de papel manuseado pelos envolvidos. No entanto desta situação advem certas dificuldades dos orientadores, coordenadores de cursos e alunos, para gerir Trabalhos de Fim de Curso (TFC). Os cronogramas costumam atrasar e os coordenadores e orientadores têm dificuldades para controlar o progresso dos projetos.

1.1 Definição do problema

A falta de um sistema digital para o acompanhamento e controle dos Trabalhos de Fim de Curso na Universidade Técnica de Angola(UTANGA) tem dificultado a instituição no acompanhamento e controle dos TFC, bem como na partilha de informações entre orientadores e orientandos.

- **Como melhorar a gestão e o controle dos trabalhos de fim de curso na UTANGA garantindo a centralização de dados e a gestão eficiente?**

1.2 Hipótese

Se implementarmos um sistema web para gestão e controle dos trabalhos de fim de curso (TFC), irá possibilitar que os intervenientes tenham uma melhor interação, e informação necessárias disponível a tempo para garantir um melhor acompanhamento dos TFC.

1.3 Justificativa

Para quem passa ou já passou pelo mesmo, sabe o quão desafiador é o processo do TFC, do início até a conclusão do trabalho, é desafiante não apenas para quem o faz, mas também para quem coordena, orienta. Os desafios começam desde a escolha do tema, interação entre os intervenientes deste processo (coordenador, orientador, aluno, etc...), bem como ter a informação a tempo.

Sendo parte constituinte do processo, passando por dificuldades e observando os meus colegas passarem pelas mesmas, motivou-me a acolher este desafio. Por outra, aplicar na prática os conhecimentos adquiridos tanto dentro como fora da instituição dando resolução a um problema real tendo em mente que vai beneficiar tanto a instituição como os alunos, e a possibilidade de conhecer outras ferramentas científicas, tem funcionado como combustível para continuar essa empreitada.

Com o sistema de gestão de trabalho de fim do curso (TFC) implementado facilitará a universidade acompanhar os alunos inscritos na disciplina TFC, acompanhar progresso do TFC, comunicar aos envolvidos sobre assuntos relacionados ao mesmo, bem como visualizar a

interação entre alunos e orientadores. Para alunos dará a eles a possibilidade de enviar seu tema para aprovação, escolher tutor, receber tarefas do seu tutor.

1.4 Objectivos

1.4.1 Objectivo Geral

- Desenvolver um sistema web de gestão de trabalho de fim do curso (TFC).

1.4.2 Objectivos específicos

A fim de alcançar o resultado esperado, foi definido alguns objectivos específicos:

- Estudar o processo básico de elaboração, desenvolvimento e apresentação de monografias;
- Compreender o funcionamento de um sistema web de gestão de TFC;
- Estudar metodologias de desenvolvimento de software;
- Realizar levantamentos de requisitos;
- Elaborar os diagramas UML, com base os requisitos;
- Desenhar as interfaces (Layout) para aplicação;
- Elaborar a arquitetura de Software;
- Codificar a aplicação;
- Efectuar testes;

1.5 Organização do trabalho

Para melhor enquadramento e situação no relatório, o presente trabalho foi organizado por capítulos, como se segue a abaixo:

- **Capítulo 1** – Neste capítulo é feita uma breve introdução sobre o trabalho realizado apresentado elementos importantes como a problemática, justificativa e os objectivos a serem atingidos.
- **Capítulo 2** - Neste capítulo abordar-se-á a fundamentação teórica onde será reservado a abordagem mais teórica sobre o projecto desenvolvido, apresentando algumas tecnologias envolvidas no projecto, arquitecturas e metodologias usadas para a construção de softwares.

- **Capítulo 3** - Este capítulo apresenta a análise metodológica, onde são apresentadas as técnicas, os procedimentos e os métodos de pesquisa usados para a elaboração deste projecto, apresentamos os diagramas UML em função da análise que foi feita, fizemos a descrição das tecnologias envolvidas.
- **Capítulo 4** - Neste capítulo é apresentado o nosso projecto de forma funcional, a interface juntamente com alguma explicação do funcionamento do mesmo.

2. FUNDAMENTAÇÃO TEÓRICA

2.1 Web Services

Os Web Services revolucionaram a forma como os sistemas se comunicam e trocam informações na era digital. Através de protocolos padronizados e modelos cliente-servidor, eles facilitam a integração entre diferentes plataformas e linguagens de programação, promovendo interoperabilidade e flexibilidade.

2.1.1 Definição e Conceitos Fundamentais

De acordo com Kremmer (2003), um Web Service é um componente de software modular que oferece uma funcionalidade específica e acessível através da rede. Ele segue protocolos padronizados como SOAP, XML e REST, permitindo a comunicação entre diferentes sistemas, independentemente da linguagem de programação ou plataforma utilizada.

2.1.2 Funcionamento e Arquitetura

O modelo cliente-servidor é fundamental para o funcionamento dos Web Services. Um cliente inicia a interação enviando uma solicitação a um servidor. O servidor processa a solicitação, executa a tarefa desejada e retorna uma resposta ao cliente. Essa troca de informações é realizada através de protocolos padronizados, garantindo a interoperabilidade entre os sistemas.

2.1.3 Vantagens e Benefícios

A adoção de Web Services oferece diversos benefícios, como:

- **Interoperabilidade:** Facilita a integração entre diferentes sistemas e plataformas, independentemente da linguagem de programação ou arquitetura.
- **Reutilização:** Permite que funcionalidades sejam reutilizadas em diferentes aplicações, otimizando o desenvolvimento de software.

- **Flexibilidade:** Promove flexibilidade na arquitetura de sistemas, permitindo a adaptação a novas necessidades e mudanças no negócio.
- **Padronização:** Garante a interoperabilidade através de protocolos padronizados como SOAP, XML e REST.
- **Acessibilidade:** Facilita o acesso a recursos e funcionalidades remotamente, através da rede.

2.1.4 Aplicações e Casos de Uso

Os Web Services são utilizados em diversas áreas, como:

- **Comércio eletrônico:** Integração de sistemas de pagamento, catálogos de produtos e gerenciamento de pedidos.
- **Serviços financeiros:** Acesso a informações bancárias, transações financeiras e serviços de investimento.
- **Saúde:** Compartilhamento de registros médicos, agendamento de consultas e telemedicina.
- **Logística:** Rastreamento de encomendas, gerenciamento de estoque e otimização de rotas.
- **Governo eletrônico:** Acesso a serviços públicos, consulta de informações e pagamento de impostos.

Em resumo, os Web Services se consolidaram como uma ferramenta poderosa para a integração de sistemas e troca de informações na era digital. Sua flexibilidade, interoperabilidade e padronização os tornam essenciais para o desenvolvimento de soluções escaláveis e robustas em diversos setores.

2.2 Sistema de Gestão

Para Rodriguez y Rodrigues (2002) a gestão de um modo abrangente, tem a ver com a forma como os relacionamentos entre as pessoas se estabelecem na busca de um objetivo comum.

A apresentação de forma estruturada e organizada de como ocorre a integração entre os seus sistemas internos, formais e informais, que fazem com que seja assegurado o atendimento às estratégias de negócio suportadas pelas pessoas dentro de uma Organização formal de poder e sistemas.

Um sistema de gestão é um programa de computador que ajuda a cuidar das atividades de uma empresa. Ele é um software inteligente e que tem como objetivo facilitar as atividades do dia a dia, automatizando o máximo de processos quanto for possível.

Os resultados apresentados por um sistema de gestão extrapolam o carácter financeiro. Pois podemos obter alguns benefícios da implementação de um SG, que, por conseguinte, poderão ser representados por meio de indicadores de qualidade:

- Maior segurança para os funcionários;
- Melhoria nos índices de satisfação interna;
- Aumento da qualidade de vida na comunidade na qual a empresa atua.

2.2.1 Vantagens de um sistema de gestão

Com a delimitação de indicadores e o acompanhamento constante das métricas relacionadas à implantação de um sistema de gestão em um negócio, diversas vantagens podem ser observadas. Eis as principais:

- Aumenta a transparência;
- Diminui os riscos de acidentes de trabalho;
- Reduz a burocracia no trâmite de processos;
- Aprimora o clima organizacional da empresa;
- Reduz os danos causados ao meio ambiente;
- Torna o negócio mais competitivo e mais próximo da excelência;
- Padroniza processos em consonância aos padrões internacionais;
- Possibilita um ambiente de trabalho mais seguro, agradável e produtivo;
- Fortalece a percepção de marca tanto entre o público interno, quanto entre o externo.

2.2.2 Sistemas de Gestão de TFCs

Os Sistema de Gestão de Trabalhos de Fim de Curso de um modo geral é uma ferramenta de muita utilidade pós, é possível obter informações actualizadas sobre os trabalhos, controlar os alunos que estão trabalhando nos projectos e obter informações extras valiosas para auxiliar na tomada de decisões, isto tudo em tempo real.

Mas apesar da reconhecida existência no mercado de alguns sistemas de Gestão de Trabalhos de Fim de Curso, tais soluções nem sempre são adotadas pelas universidades, tendo em vista que não se aplica o enquadramento dos processos feitos na mesma bem como as suas praticas. Assim, pretende-se com este projeto dar uma solução que se aplica no contexto local e possa responder as necessidades da instituição, também podem dispor de um sistema de gestão prático (pela facilidade de interação com o usuário), gratuito ou de custo extremamente reduzido e eficiente. Ou seja, mais controle, qualidade e produtividade com menos pessoal, tempo, tecnologia e recursos financeiros.

Mais a baixo apresentamos dois (2) exemplos de como funcionam alguns sistemas de Gestão de TFCs que serviram como base de estudo para os trabalhos relacionados:

- **SIGAA (Sistema Integrado de Gestão de Atividades Acadêmicas)**

O SIGAA (Sistema Integrado de Gestão de Atividades Acadêmicas) é uma plataforma de gestão acadêmica amplamente utilizada em instituições de ensino superior no Brasil. Desenvolvido pela Universidade Federal do Rio Grande do Norte (UFRN), o SIGAA oferece uma variedade de recursos para auxiliar na administração de atividades acadêmicas, incluindo o acompanhamento de Trabalhos de Fim de Curso (TFCs);

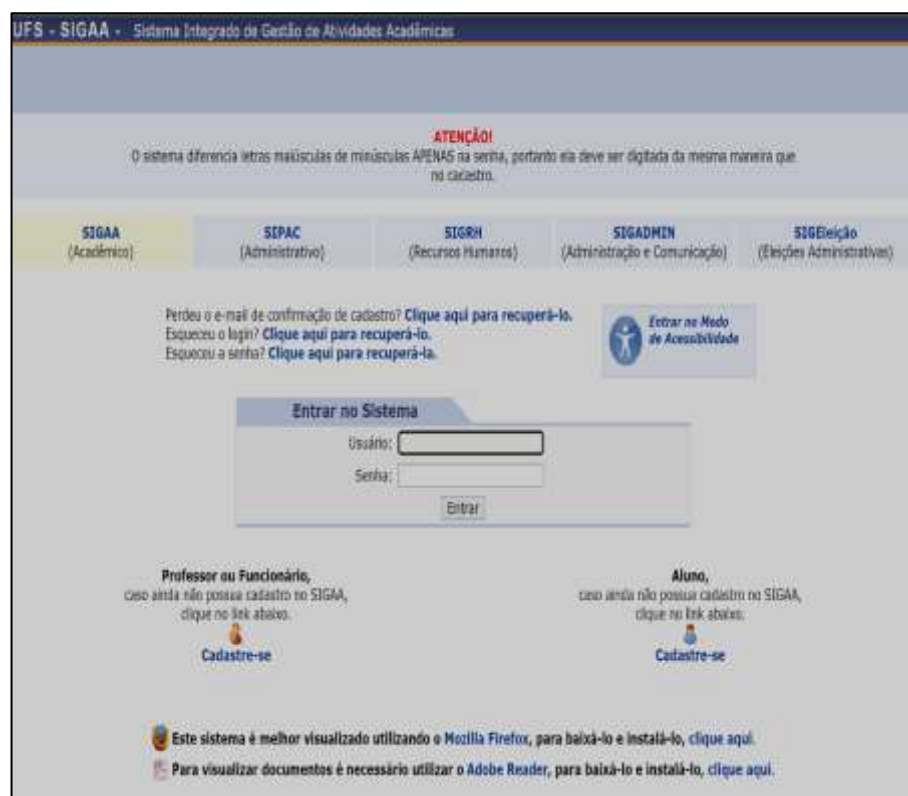


Figura 2. 1 O SIGAA (Sistema Integrado de Gestão de Atividades Acadêmicas)

- **TIDIA-AE(Ambiente de Ensino)**

O Tidia-ae é uma plataforma de ensino e aprendizado desenvolvida pela Universidade de São Paulo (USP), no Brasil. Seu nome deriva de "Tecnologias da Informação e Design Interativo para a Aprendizagem" (Tidia). O "ae" significa "ambiente de ensino".

Essa plataforma tem como objetivo principal oferecer suporte digital para as atividades de ensino, permitindo a criação e gestão de ambientes virtuais de aprendizagem. O Tidia-ae é uma ferramenta completa que pode ser usada por professores e alunos para diversas finalidades educacionais, desde a disponibilização de materiais didáticos até a realização de atividades interativas e colaborativas.

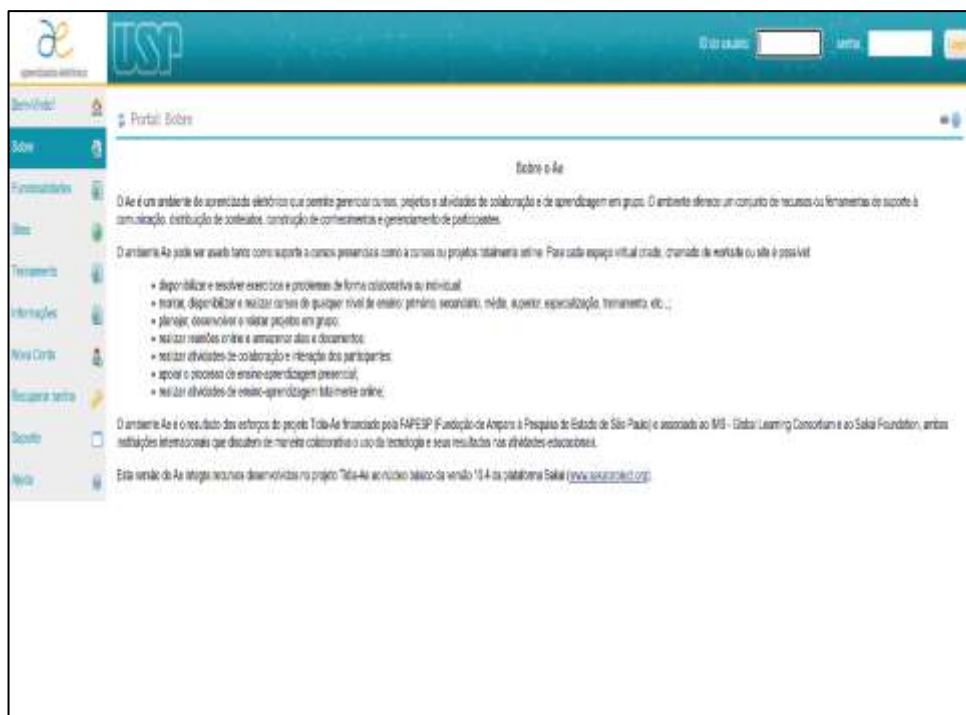


Figura 2. 2 Sistemas de Gestão TIDIA-AE(Ambiente de Ensino)

A mesma dispõe das seguintes funcionalidades: Submissão de trabalhos; Agendamento de defesas; Gestão de Alunos e Professores. O administrador pode ter acesso completo de onde estiver com tablet ou smartphone.

As duas soluções acima, apesar de não serem testadas por nós, apresentam boas funcionalidades que facilitam a gestão de TFCs. Mas uma das grandes desvantagens é que não estão disponíveis para o nosso país e as informações requisitos utilizados para o desenvolvimento dos mesmos são bastante genéricas e viradas para o público brasileiro. Sendo assim a nossa solução é uma

solução virada para o nosso país e que esta a ser desenvolvida não de forma genérica, mas sim tendo em conta a uma entidade real.

2.3 Engenharia de Software

É um conjunto integrado de métodos e ferramentas utilizadas para especificar, projetar, implementar e manter um sistema.

A Engenharia de Software é a área da computação que estabelece uma abordagem sistemática de desenvolvimento de software com qualidade envolvendo processos, técnicas e ferramentas apropriadas para uma ampla gama de aplicações, considerando prazos, restrições e recursos disponíveis (PRESSMAN, 2011).

A criação da Engenharia de Software surgiu no intuito de contornar a crise do software, dando um tratamento de engenharia ao desenvolvimento de sistemas complexos caracterizados por um conjunto de componentes abstratos (estrutura de dados e algoritmos) encapsulados na forma de procedimentos, funções, módulos, objetos ou agentes e interconectados entre si, compondo a arquitetura do software, devendo ser executados em sistemas computacionais.

2.3.1 Requisitos do Sistema

Para Silva e Videira (2001), um requisito é uma funcionalidade ou condição que o sistema deverá possuir. Para os identificar adequadamente, é aplicado um conjunto de técnicas de modo a obter a percepção detalhada daquilo que o sistema deverá efectuar.

E eles podem ser extraídos realizando reuniões com os interessados, a elaboração de questionários, a observação das actividades e do funcionamento do dia-a-dia, a recolha e análise de documentação diversa, a elaboração de pequenos protótipos do sistema que permitam validar mais facilmente a percepção obtida (seguindo o princípio que "uma imagem vale mais do que mil palavras"). Deve-se ter a preocupação de encontrar a melhor solução, pois às vezes aquilo que o utilizador pede não é sempre o que ele necessita (este facto está relacionado com o seu desconhecimento do que se pode obter de um sistema de informação). Outra questão a considerar tem a ver com a importância de identificar não apenas as funcionalidades actuais, mas sobretudo determinar a situação futura a atingir, (Silva e Videira, 2001).

2.3.1.1 Tipo de Requisitos

Esses requisitos também são impostos pelos diversos stakeholders do software e estão normalmente relacionados a interfaces com o usuário, capacidades, consumo de recursos e escalas de tempo.

Bennet, McRobbe Farmer (1999), identificam as seguintes categorias de requisitos:

- Requisitos funcionais - descrevem o que um sistema faz ou é esperado que faça. Estes são os requisitos que inicialmente serão levantados, abrangendo a descrição de processamentos a efectuar pelo sistema, entradas (inputs) e saídas (outputs) de informação em papel ou no ecrã que derivam da interacção com pessoas e outros sistemas.
- Requisitos não funcionais - relacionados com as características qualitativas do sistema, descrevendo a qualidade com que o sistema deverá fornecer os requisitos funcionais. Abrange medidas de desempenho como, por exemplo, tempos de resposta, volume de dados ou considerações de segurança.

2.3.2 Modelação de Sistemas

Para Lancaster (2000), como parte dos requisitos do sistema e da actividade de projectos, o sistema precisa ser modelado como um conjunto de componentes e de relações entre esses componentes. Isso é, normalmente, ilustrado graficamente em um modelo de arquitectura de sistema, que proporciona ao leitor uma visão geral da organização do sistema.

2.3.2.1 Linguagem UML

UML é a sigla de Unified Modelling Language, que pode ser traduzido por Linguagem de Modelação Unificada. A UML é uma linguagem que utiliza uma notação padrão para especificar, construir, visualizar e documentar sistemas de informação orientados por objectos segundo (Nunes e O'Neill).

Particularmente no que tange à engenharia de software, a linguagem UML pode ser utilizada para modelar todas as etapas do processo de desenvolvimento de software, bem como produzir todos os artefatos de software necessários à documentação dessas etapas, segundo (Gudwin, 2015)

Um modelo em UML é constituído por um conjunto de diagramas que representam aspectos complementares de um sistema de informação. E para cada um destes diagramas são utilizados símbolos que representam os elementos que estão a ser modelados (abstracções) e linhas que relacionam esses elementos. Os símbolos e as linhas têm significado específico e possuem formas distintas, constituindo uma forma de notação, segundo (Nunes e O'Neill).

Para (Nunes e O'Neill), a UML disponibiliza o seguinte conjunto mais importantes de diagramas:

- **Diagrama de Use Case** - serve para identificar as fronteiras do sistema e descrever os serviços (use cases) que devem ser disponibilizados a cada um dos diversos utilizadores (actores);
- **Diagrama de Classes** - através do qual descrevemos a estrutura de informação (classes e suas relações) que é utilizada no sistema;
- **Diagrama de Actividade** - pode ser utilizado para descrever cada um dos use cases, realçando o encadeamento de actividades realizadas por cada um dos objectos do sistema, numa óptica de fluxo de trabalho (work-flow).
- **Diagrama de Sequência** - O diagrama de sequência é um diagrama de interacção que realça a ordem cronológica das mensagens entre objectos.

2.4 Base de Dados

Para Coelho (2011) base de dados é um local onde pode ser guardada informação. A informação pode ser consultada, alterada, apagada, na totalidade ou parcialmente, através de uma aplicação conhecida como Sistema de Gestão de Base de Dados (SGBD), também chamada simplesmente de Base de Dados (BD).

Uma das principais finalidades de uma base de dados é que as informações que elas contêm sirvam para uma variedade de aplicações distintas. Para isso, é importante que a base de dados seja “substancialmente não-redundante, isto é, possuir o mínimo de duplicidade de dados idênticos, de preferência nenhuma” (Rowley, 2002, p. 125).

2.4.1 Sistemas de base de dados

Um sistema de base de dados tenta baixar os custos de manutenção através da separação entre a forma como os dados são percebidos pelo programador e a forma como esses dados são armazenados fisicamente.

De acordo com DATE (2004), um sistema de base de dados é “um sistema computadorizado cuja finalidade geral é armazenar informações e permitir que os usuários busquem e atualizem essas informações quando as solicitar”. Para o autor um sistema de base de dados é composto por dados, hardware, software e usuários.

Para Um Sistema Gerenciador de Base de Dados (SGBD) é uma coleção de programas que permitem aos usuários criarem e manipularem uma base de dados. Um SGBD é, assim, um sistema de software de propósito geral que facilita o processo de definir, construir e manipular bases de dados de diversas aplicações:

Definir uma base de dados envolve a especificação de tipos de dados a serem armazenados na base de dados. Construir uma base de dados é o processo de armazenar os dados em algum meio que seja controlado pelo SGBD.

Manipular uma base de dados indica a utilização de funções como a de consulta, para recuperar dados específicos, modificação da base de dados para refletir mudanças no mini-mundo (inserções, atualizações e remoções), e geração de relatórios.

2.4.2 Modelo Entidade Relacionamento (MER)

Segundo Takai, Italiano e Ferreira (2005) O MER é um modelo de dados conceitual de alto-nível, ou seja, seus conceitos foram projetados para serem compreensíveis a usuários, descartando detalhes de como os dados são armazenados.

E é actualmente usado principalmente durante o processo de projeto da base de dados. Segundo (Silva) O MER, ou simplesmente ER, foi desenvolvido com o objetivo de facilitar o projeto de base de dados por meio de um modelo independente de implementação e de fácil compreensão por parte do usuário

Os relacionamentos são descritos através da cardinalidade, que indica como as instâncias das entidades se relacionam. Os tipos utilizados na modelagem são (KORTH, SILBERCHATZ e SUDARSHAN, 2006):

- **Um-para-um (1:1):** uma instância em “A” está associada com no máximo uma instância em “B”, e uma instância em “B” está associada com no máximo uma instância em “A”;
- **Um-para-muitos (1:n):** uma instância em “A” está associada a qualquer número de instâncias em “B”, e uma instância em “B”, todavia, pode estar associado a no máximo uma instância em “A”;
- **Muitos-para-muitos (n:n):** uma instância em “A” está associada a qualquer número de instâncias em “B” e vice-versa. Alguns autores preferem chamar esta cardinalidade de m:n, por considerar que podem representar valores diferentes.

2.4.3 SQL

A Structured Query Language (SQL) ou Linguagem de Consulta Estruturada foi criada pela IBM Research, no início da década de 1970, para o protótipo de um sistema de banco de dados chamado System R (DATE, 2004).

Apesar de conhecida como uma “linguagem de consulta”, a SQL oferece também recursos para definir a estrutura dos dados, atualizar, incluir, excluir e alterar dados, especificar restrições de integridade e outros recursos mais (SILBERSCHATZ; KORTH; SUDARSHAN, 1999).

Ainda nesta senda, de acordo com DATE (2004), a SQL possui, entre outros, os seguintes componentes:

Linguagem de Definição de Dados (“Data Definition Language” - DDL): é utilizada pelo DBA e projetistas de base de dados para definir seus esquemas. O SGBD tem um compilador para processar descrições em DDL e construir a descrição do esquema armazenado no catálogo;

Linguagem de Manipulação de Dados (“Data Manipulation Language” - DML): uma vez que o esquema é compilado e a base de dados preenchida com dados, os usuários têm que ter algum modo de manipular os dados. Manipulações comuns como

recuperação, inserção, remoção e modificação de dados são realizadas pela DML (Takai, Italiano e Ferreira, 2005).

Segundo Battisti (2000), todas as instruções SQL são conduzidas com um único comando que contem uma descrição completa da informação exigida. Ao escrever uma instrução SQL, você não deve se preocupar em como os dados são recuperados, mas somente com o conteúdo do conjunto de dados. Este é o principal objectivo do SQL.

2.5 Processo de desenvolvimento de sistemas

Ainda que os processos tenham de ser definidos caso a caso, de maneira geral, o ciclo de vida de um software envolve, pelo menos, as seguintes fases (Falbo e Barcellos, 2011):

- **Planeamento:** O objetivo do planeamento de projeto é fornecer uma estrutura que possibilite ao gerente fazer estimativas razoáveis de recursos, custos e prazos. Uma vez estabelecido o escopo de software, com os requisitos esboçados, uma proposta de desenvolvimento deve ser elaborada, isto é, um plano de projeto deve ser elaborado configurando o processo a ser utilizado no desenvolvimento de software. À medida que o projeto progride, o planeamento deve ser detalhado e atualizado regularmente. Pelo menos ao final de cada uma das fases do desenvolvimento (análise e especificação de requisitos, projeto, implementação e testes), o planeamento como um todo deve ser revisto e o planeamento da etapa seguinte deve ser detalhado. O planeamento e o acompanhamento do progresso fazem parte do processo de gestão de projeto.
- **Análise e Especificação de Requisitos:** Nesta fase, o processo de levantamento de requisitos é intensificado. O escopo deve ser refinado e os requisitos mais bem definidos. Para entender a natureza do software a ser construído, o engenheiro de software tem de compreender o domínio do problema, bem como a funcionalidade e o comportamento esperados. Uma vez capturados os requisitos do sistema a ser desenvolvido, estes devem ser modelados, avaliados e documentados. Uma parte vital desta fase é a construção de um modelo descrevendo o que o software tem de fazer (e não como fazê-lo).
- **Projeto:** Esta fase é responsável por incorporar requisitos tecnológicos aos requisitos essenciais do sistema, modelados na fase anterior e, portanto, requer que a plataforma de implementação seja conhecida. Basicamente, envolve duas grandes etapas: projeto

da arquitetura do sistema e projeto detalhado. O objetivo da primeira etapa é definir a arquitetura geral do software, tendo por base o modelo construído na fase de análise de requisitos. Essa arquitetura deve descrever a estrutura de nível mais alto da aplicação e identificar seus principais componentes. O propósito do projeto detalhado é detalhar o projeto do software para cada componente identificado na etapa anterior. Os componentes de software devem ser sucessivamente refinados em níveis maiores de detalhamento, até que possam ser codificados e testados.

- **Implementação:** O projeto deve ser traduzido para uma forma passível de execução pela máquina. A fase de implementação realiza esta tarefa, isto é, cada unidade de software do projeto detalhado é implementada.
- **Testes:** inclui diversos níveis de testes, a saber, teste de unidade, teste de integração e teste de sistema. Inicialmente, cada unidade de software implementada deve ser testada e os resultados documentados. A seguir, os diversos componentes devem ser integrados sucessivamente até se obter o sistema. Finalmente, o sistema como um todo deve ser testado.
- **Entrega e Implantação:** uma vez testado, o software deve ser colocado em produção. Para tal, contudo, é necessário treinar os usuários, configurar o ambiente de produção e, muitas vezes, converter bases de dados. O propósito desta fase é estabelecer que o software satisfaz os requisitos dos usuários. Isto é feito instalando o software e conduzindo testes de aceitação. Quando o software tiver demonstrado prover as capacidades requeridas, ele pode ser aceito e a operação iniciada.
- **Operação:** nesta fase, o software é utilizado pelos usuários no ambiente de produção.
- **Manutenção:** Indubitavelmente, o software sofrerá mudanças após ter sido entregue para o usuário. Alterações ocorrerão porque erros foram encontrados, porque o software precisa ser adaptado para acomodar mudanças em seu ambiente externo, ou porque o cliente necessita de funcionalidade adicional ou aumento de desempenho. Muitas vezes, dependendo do tipo e porte da manutenção necessária, essa fase pode requerer a definição de um novo processo, onde cada uma das fases precedentes é reaplicada no contexto de um software existente ao invés de um novo.

2.5.1 Processo Incremental

Para Silva e Videira (2001) a noção de processo incremental corresponde à ideia de “aumentar (ou alargar) pouco-a-pouco” o âmbito do sistema. Uma boa imagem para este atributo é a de uma mansão que foi construída por sucessivos incrementos a partir de uma primeira casa com apenas duas divisões.

Segundo Lancaster (2000), a abordagem do desenvolvimento incremental foi sugerida por Mills [Millset al., 1980] como um meio de reduzir o ‘retrabalho’ no processo de desenvolvimento e de proporcionar aos clientes algumas experiencias de adiar decisões sobre seus requisitos detalhados, até que eles tenham alguma experiencia com o sistema.

Em um processo incremental, os clientes identificam, em um esboço, as funções a serem fornecidas pelo sistema. Eles identificam quais funções são mais importantes e quais são menos importantes para eles. Em seguida é definida uma série de estágios de entrega, com cada estágio fornecendo um subconjunto das funcionalidades do sistema. A alocação de funções aos estágios depende da prioridade da função. As funções prioritárias são entregues primeiramente ao cliente, (Lancaster, 2000).

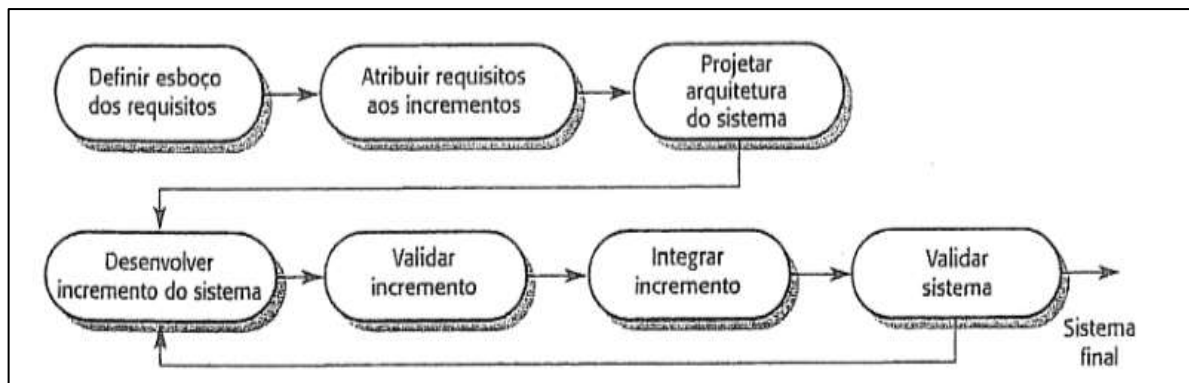


Figura 2. 3 Processo Incremental

Fonte: Vasconcelos, Rouiller, Machado, Medeiros(2006)

Segundo Vasconcelos, Rouiller, Machado, Medeiros(2006),as principais vantagens do modelo incremental são:

- A funcionalidade do sistema estará disponível mais cedo, pois ela é entregue a partir dos incrementos;
- Incrementos iniciais agem como um protótipo para ajudar a elicitar requisitos para incrementos finais;

- Diminuem-se os riscos de falhas no projeto como um todo;
- Os serviços de prioridade mais alta do sistema tendem a receber mais testes.

2.6 Qualidade de Software

Qualidade de software é a conformidade dos requisitos funcionais e de desempenho explicitamente declarados, a padrões de desenvolvimento claramente documentados e a características implícitas que são esperadas de todo software profissionalmente desenvolvido (Pressman, 1995)

A avaliação da qualidade de software pode ser realizada em dois momentos: durante a geração do software e após este estar pronto para o uso, chamando esses dois momentos, respectivamente, de processo e produto.

No processo procura-se avaliar de que forma o software está sendo desenvolvido, identificando práticas que possam conduzir a problemas na qualidade do produto e desenvolvendo e/ou utilizando métodos e ferramentas que evitem esses problemas. Já no produto concluído, procura-se avaliar a sua qualidade a fim de identificar deficiências e limitações em sua aplicabilidade como um produto final.

E de acordo com Shiba (1997), deve-se considerar alguns aspectos para se obter a qualidade do produto:

- **Funcionalidade:** identifica os procedimentos de funcionamento de um produto;
- **Confiabilidade:** o produto não deve apresentar problemas junto aos clientes, caso contrário, o fornecedor deverá resolvê-los;
- **Usabilidade:** deve-se testar o máximo possível o produto e constatar o resultado como satisfatório;
- **Eficiência:** comprovação, pelo cliente, de sua satisfação com o produto;
- **Manutenibilidade:** garantia de correções dos problemas.
- **Portabilidade:** o produto muda de ambiente e a operação ocorre da mesma forma satisfatória.

2.7 Arquitectura de Software

Uma arquitetura de software envolve a descrição de elementos arquiteturais dos quais os sistemas serão construídos, interações entre esses elementos, padrões que guiam suas composições e restrições sobre estes padrões segundo. (PFLEEGER, 1998).

A arquitetura de um software consiste na definição de seus componentes, as propriedades externamente visíveis destes elementos e os relacionamentos entre eles, enfatizando a separação dos interesses. (BASS, 2003).

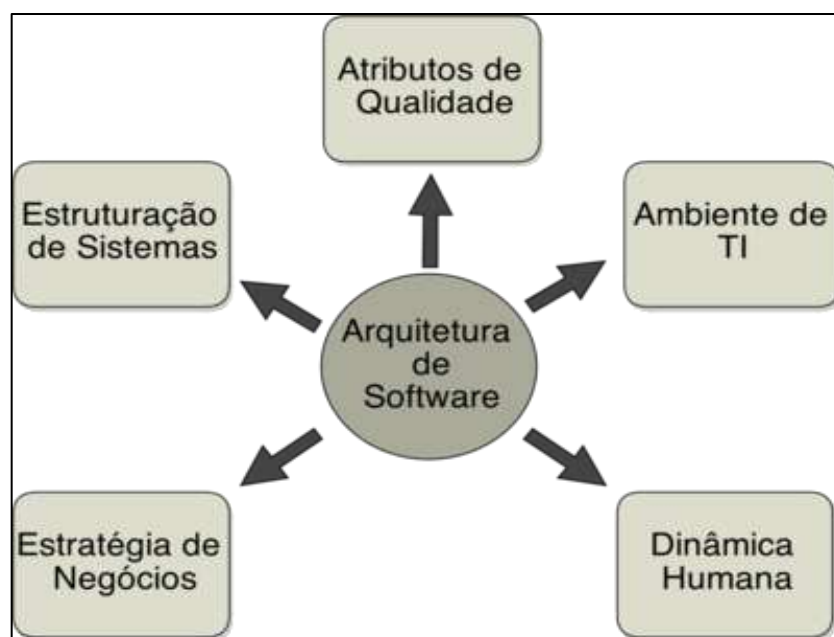


Figura 2. 4 Arquitetura de Software

Fonte: lobotech.com.br

2.7.1 Linguagem de Programação

Na programação de computadores, uma linguagem de programação serve como meio de comunicação entre o indivíduo que deseja resolver um determinado problema e o computador.

Uma Linguagem de Programação é uma linguagem destinada a ser usada por uma pessoa para expressar um processo através do qual um computador pode resolver um problema. Três categorias principais de linguagem de programação têm sido desenvolvidas: linguagem de máquina, linguagens assembly e linguagens de alto nível. Onde apenas os primeiros computadores somente podiam ser programados usando linguagem de máquina. Linguagem

essa que usa sequências de zeros e uns (bits) que representam instruções precisas para computação e acessos de dados.

2.7.2 Linguagem C#

Neste capítulo falaremos de uma das linguagens usadas no desenvolvimento do nosso trabalho, que é a linguagem C#.

O C# é uma linguagem orientada a objetos com a qual podemos criar classes que podem ser utilizadas por outras linguagens como, por exemplo, o Visual Basic. Uma característica importante é que ainda é possível utilizar os componentes COM, facilitando assim uma rápida migração para um ambiente de desenvolvimento de alto nível sem precisar reescrever todas as aplicações que você possui.

Segundo Bittuti(), o C# é uma linguagem da Microsoft, apresentada juntamente com o Framework .NET. O C# foi construído com base nos conceitos de Orientação a objectos.

Para (Ferguson, Patterson, Beres, Boutquin, Gupta) 2003, o C# proporciona vários operadores e que lhe permitem escrever expressões matemáticas e de bits. Muitos (Mas nem todos) destes operadores podem ser redefinidos. Permitindo-lhe mudar a forma em que trabalham estes operadores.

C# admite uma larga lista de expressões que lhe permitem definir varias rotas de execução dentro do código. As instruções de fluxo de controlo que usam palavras chave como *if*, *switch*, *while*, *for*, *break* e *continue* permitem ao código ramificar-se por caminhos diferentes. Dependendo dos valores de suas variáveis. As classes podem conter códigos e dados. Cada membro de uma classe tem algo chamado âmbito de acessibilidade. Que define a visibilidade do membro com respeito a outros objectos. C# admite os âmbitos de acessibilidade *public*, *protected*, *internal* e *private* (Ferguson, Patterson, Beres, Boutquin, Gupta, 2003)

2.7.3 Programação em camadas

Para este ponto é importante frisar que usaremos a programação em 3 camadas. Pois muitas são as vantagens em usar este modelo, em relação a facilidade de gerenciamento e actualização das aplicações.

A ideia básica do modelo em 3 camadas é retirar as Regras do Negócio, da aplicação Cliente e centraliza-las em um determinado ponto, o qual é chamado de Servidor de Aplicações. O acesso

à base de dado é feito através de regras contidas no Servidor de Aplicações. Ao centralizar as Regras de Negócio em um único ponto, fica mais fácil a actualização das mesmas (Battisti, 2000).

Para Battisti (2000), todo o acesso do cliente, aos dados do servidor de base de dados, é feito de acordo com as regras contidas no Servidor de Aplicações. O cliente não tem acesso aos dados do servidor de Base de dados, sem antes passar pelo servidor de aplicações. Com isso, as três (3) camadas são as seguintes:

- **Camada de Apresentação:** Continua no programa instalado. Alterações na Interface do programa ainda irão gerar a necessidade de actualizar a aplicação em todas as estações de trabalho, onde a aplicação estiver sendo utilizada. Porém cabe ressaltar que alterações na interface são menos frequentes nas regras de negócio;
- **Camada de Lógica de Negócio:** São regras do negócio, as quais determinam de que maneira os dados serão utilizados e manipulados. Desta maneira, quando uma regra do negócio for alterada, basta actualizá-la no Servidor de Aplicações. Após a actualização, todos os usuários passarão a ter acesso à nova versão, sem que seja necessário reinstalar o programa em cada um dos computadores da rede;
- **Camada de Acesso a Dados:** Nesta camada temos o servidor de Base de Dados, no qual reside toda a informação necessária para o funcionamento da aplicação. Cabe reforçar que os dados são acessados somente através do Servidor de Aplicação, e não directamente pela aplicação cliente.

2.8 Linguagens e Tecnologias Web

Linguagens Web, é um ponto do desenvolvimento web que abrange todas as linguagens de programação web, as mais usadas hoje são:

HTML - HyperText Markup Language: é a linguagem usada para descrever e definir o conteúdo de uma página Web em um formato estruturado. A linguagem de programação HTML libera permissão a máquinas diferentes de ler documentos da Internet, por meio do protocolo HTTP, dando acesso a documentos de um único endereço na rede (chamado protocolo HTTP, dando acesso a documentos de um único endereço na rede (chamado URL). (KIOSKEA.NET, 2008).

Java script: o Java script tem a possibilidade de criar pequenos detalhes (programas) em uma página web ou em programas orientados a objetos.

É uma linguagem de script que quando agregado nos tag's Html, permite também modernizar (incrementar) a apresentação e interatividade de páginas Web.

Características Java Script segundo SANTOS (2006):

- Permite acessar os objetos do browser;
- Código integrado na página HTML;
- Código interpretado pelo browser no momento da execução;
- Códigos de programação simples, mas para aplicações limitadas;
- Confidencialidade do código é nulo (Código é visível por ser client side),
- Criar efeitos especiais nas paginas e definir interatividades com o usuário.

2.8.1 ASP.NET

Falando um pouco do ASP.NET, esta plataforma introduz uma camada de abstração que nos permite trabalhar com valores obtidos através de pedidos http no lado servidor. Uma das vantagens decorrentes da utilização desta plataforma reside no facto de esta conseguir transformar eventos cliente (gerados do browser) em eventos servidor que podem ser tratados através da adição de código escrito numa das linguagens da plataforma .NET (tudo isto de forma quase transparente para o programador). (Abreu, 2013).

Bem, segundo Abreu (2013) a geração de eventos servidor é conseguida através da utilização de formulários e controlos servidor (elementos anotados com o atributo runat = “server”), que geram o código HTML da página carregada no browser e que são responsáveis por iniciar um novo pedido a partir do cliente para a própria pagina no lado servidor. E estas operações são designadas por POSTBACKS. A plataforma garante ainda que os estados dos controlos no lado servidor são mantidos (e actualizados, quando necessário) entre pedidos.

2.8.2 CSS

Então surgiu o CSS, que é uma outra linguagem, separada do HTML, como objetivo único de cuidar da estilização da página. A vantagem é que o CSS é bem mais robusto que o HTML para estilização, como veremos. Mas, principalmente, escrever formatação visual misturado com

conteúdo de texto no HTML se mostrou algo bem impraticável. O CSS resolve isso separando as coisas; regras de estilo não aparecem mais no HTML, apenas no CSS.

Uma folha de estilo CSS (folha de estilo em cascata) não é XHTML, mas sim, um conjunto informações sobre a formatação e exibição dos componentes do layout de uma página WEB. Ela é um código separado que altera as características da página. O CSS deve ser utilizado para liberar do XHTML ou do HTML o peso da responsabilidade da apresentação.

3. METODOLOGIA

Actualmente nas empresas é necessário que se tenha algum nível de processo visando como objectivo a qualidade no desenvolvimento de software.

3.1 Metodologia de Investigação Científica (Metodologia de Pesquisa)

O tipo de pesquisa realizada no presente trabalho foi descritiva e exploratória em relação aos objetivos, visto que, segundo Gil (1996), proporciona uma proximidade com a questão.

No sentido de, construir hipóteses, a metodologia inclui entrevistas com pessoas directamente envolvidas, sobre a forma como fazem a monografia e as experiências que tiveram no processo.

Os procedimentos de coleta dos dados supracitados, foi através de pesquisa bibliográfica e documental, com abordagem quantitativa e qualitativa, com o intuito de relacionar os dados para a interpretação.

Ao longo da construção da pesquisa, o primeiro procedimento realizado diz respeito à obtenção de nome do autor, curso, tema, resumo e ano de defesa das monografias. Em seguida, definiram-se as categorias de Engenharia Informática, para melhor evidenciar o perfil dos TFC e monografias.

3.2 Descrição do Campo de estudo

A Universidade Técnica de Angola (UTANGA) é uma universidade angolana com sede no bairro do Capolo II, Quilamba Quiaxi, na província de Luanda.

Fundada aos 7 de maio de 2007, tendo como actual reitor(a), Dr. Ilídio Pascoal Simão.

Instituições orgânicas

As unidades orgânicas ofertam os seguintes cursos:

Faculdade de Engenharias Em nível de graduação, em 2017, ministrava os cursos de: Arquitectura e Urbanismo, Engenharia de Telecomunicações e Electrónica, Engenharia de Geologia e Minas, Engenharia Civil Engenharia do Ambiente, Engenharia Informática, Engenharia de Minas.

Faculdade de Letras e Ciências Sociais Em nível de graduação, em 2017, ministrava os cursos de: Relações Internacionais Psicologia, Língua e Literatura Inglesa, Direito. Faculdade de Gestão e Ciências Económicas Em nível de graduação, em 2017, ministrava os cursos de: Gestão Contabilidade e Finanças.

3.3 Delimitação do Estudo

A pesquisa e elaboração de trabalho científico é transversal a diversas áreas de conhecimento, tendo em conta a abrangência do assunto, delimitou-se o escopo desta investigação. Assim sendo, temos como nosso objecto de estudo a UTANGA.

3.4 Processo de Desenvolvimento

Para esse sistema seguiu-se a seguinte fase de desenvolvimento:

- **Levantamento de requisitos:** é a fase inicial do processo de desenvolvimento de um projeto, onde se realiza a coleta de informações essenciais relacionadas ao projeto em questão. Essas informações são obtidas por meio de entrevistas conduzidas com partes interessadas relevantes no projeto, conhecidas como stakeholders. Essas conversas visam compreender as necessidades, expectativas e requisitos específicos relacionados ao desenvolvimento do software. O objetivo é obter uma compreensão abrangente das exigências do projeto antes de prosseguir para as etapas subsequentes do desenvolvimento.
- **Análise de requisitos:** A análise de requisitos marca o ponto de partida técnico no desenvolvimento de software, concentrando-se na definição dos serviços que o sistema deve oferecer, na forma como interage com outros elementos e nas restrições que governam sua operação. No cerne dessa atividade, está a determinação do que o sistema deve realizar, delineando suas funcionalidades, em vez de detalhar como essas funcionalidades serão implementadas. Em suma, a análise de requisitos estabelece as bases, delineando o "o quê" que o sistema precisa fazer, sem entrar nos detalhes operacionais do "como".
- **Projecto:** Nesta etapa, estamos focados em definir o funcionamento do sistema de acordo com os requisitos previamente identificados, considerando as tecnologias disponíveis. Isso implica em criar um plano detalhado de como o sistema atenderá às

demandas específicas, levando em conta as capacidades e recursos tecnológicos disponíveis. Em resumo, durante o projeto, estamos elaborando uma estrutura que traduz os requisitos em um plano concreto de como o sistema será implementado, considerando as ferramentas e tecnologias disponíveis.

- **Implementação:** Nesta etapa, estamos efetivamente traduzindo o projeto em ação, realizando a codificação do sistema. Isso é feito por meio do uso de uma ou mais linguagens de programação, onde os desenvolvedores escrevem o código-fonte que transforma o plano de projeto em um software funcional. Em resumo, durante a implementação, as linhas de código estão sendo criadas para dar vida ao sistema de acordo com as especificações delineadas nas fases anteriores.
- **Testes:** Nesta fase, concentramo-nos na verificação do sistema construído, utilizando testes que foram desenvolvidos com base nas especificações delineadas na fase de projeto. O objetivo principal é identificar e relatar quaisquer erros ou falhas no software. O resultado chave desta fase é o relatório de testes, que documenta as informações sobre os problemas detectados durante os testes, contribuindo para a avaliação da qualidade e confiabilidade do sistema. Em resumo, os testes são essenciais para garantir que o software atenda aos requisitos estabelecidos e funcione de maneira eficaz.
- **Implantação:** Nesta fase, o sistema entra efetivamente em operação. Isso envolve empacotar, distribuir e instalar o software no ambiente do usuário. Além disso, durante essa fase, são criados manuais do sistema, os dados necessários são carregados, os arquivos são importados e os usuários recebem treinamento para utilizar o sistema de maneira eficaz. Em resumo, a implantação é o processo de tornar o sistema disponível e funcional no ambiente em que será utilizado, abrangendo desde a instalação técnica até a preparação e capacitação dos usuários.

3.5 Análise de Requisitos

A análise de requisitos é fundamental para o desenvolvimento de sistemas, pois trata de descobrir o que o cliente quer com o sistema. Esta mesma análise está associada ao processo de descoberta das operações que o sistema deve realizar e quais são as restrições que existirão no mesmo. Esta análise recebe o nome de Requisitos Funcionais e Requisitos não Funcionais.

3.5.1 Requisitos Funcionais

Um requisito funcional representa algo que o sistema deve fazer, ou seja, uma função esperada do sistema que agregue valor a seus usuários, exemplos típicos incluem a emissão de relatórios e a realização e manutenção de cadastros, Xexéo, (2007).

Depois de entendido os conceitos de requisitos funcionais, eis abaixo alguns requisitos funcionais do sistema:

| Ref | Nome | Descrição | Prioridade |
|------------|-------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------|
| RF1 | Submeter a proposta de TFC | Função que permite ao Estudante submeter, antes do desenvolvimento, um tema como proposta de Trabalho de Fim de Curso que estará disponível para a visualização docente. A proposta de trabalho é composta de título, descrição, indicação do nome do possível orientador. Após a submissão de uma proposta de trabalho, o status dele é definido como “Proposta”. | Importante |
| RF2 | Listar propostas de TFC | Função que permite ao coordenador ter acesso às propostas de Trabalhos de Fim de Curso submetidas. As propostas serão apresentadas com os orientadores indicados e, poderão possuir as opções de aprovar, reprovar. | Importante |
| RF3 | Aprovar Tema Proposto | Função que permite ao coordenador Aprovar um Tema Proposto para Trabalho de Fim de Curso. Quando um trabalho for aprovado, o status dele é definido como “Aprovada”. | Importante |
| RF4 | Meus Orientandos | Função que permite ao orientador ter acesso às propostas de Trabalhos de Fim de Curso submetidas pelos Estudantes. As propostas serão mostradas ao orientador indicado. | Importante |
| RF5 | Aceitar proposta | Função que permite o orientador confirmar a orientação da proposta enviada previamente pelo estudante. Quando isso acontece, o docente ganha permissão de emitir o termo de aceite e o status do trabalho é definido como “Trabalho em desenvolvimento”. | Importante |
| RF6 | Listar trabalhos defendidos | Função que permite ao professor, estudante, coordenador visualizar a lista de Trabalhos defendidos. | Essencial |
| RF7 | Visualizar Tema | Função que permite ao estudante ver as informações relacionadas ao seu tema. | Importante |
| RF8 | Submeter a versão finalizada do TFC | Função que permite ao Estudante submeter a versão finalizado do projeto. A submissão é | Importante |

| | | | |
|-------------|--------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------|------------|
| | | composta de tema, resumo e dos arquivos do trabalho concluído. | |
| RF9 | Listar tarefas | Função que permite ao professor/estudante visualizar a lista de tarefas. | Essencial |
| RF10 | Cadastrar tarefa | Função que permite ao professor criar tarefa, que pode ser (investigar sobre uma matéria, implementar uma funcionalidade). | Essencial |
| RF11 | Concluir tarefa | Função que permite ao estudante concluir a tarefa atribuída pelo o seu orientador, ao concluir é exigido anexar o arquivo com a resposta da tarefa. | Essencial |
| RF12 | Visualizar detalhes da tarefa | Função que permite ao estudante/professor visualizar os detalhes da tarefa | Essencial |
| RF13 | Agendar encontro | Função que permite ao orientador agendar encontro com os seus orientandos. | Essencial |
| RF14 | Finalizar encontro | Função que permite ao professor finalizar um encontro. | Essencial |
| RF15 | Listar encontro | Função que permite ao professor/estudante listar os encontros agendados ou que já ocorreu. | Desejável |
| RF16 | Finalizar TFC | Função que permite ao orientador alterar o estado do TFC para “Finalizado”, após ser alterado estará disponível para enviar a versão do TFC. | Importante |
| RF17 | Cadastrar área de investigação | Função que permite ao coordenador cadastrar uma área de investigação . | Importante |
| RF18 | Editar área de investigação | Função que permite ao coordenador editar uma área de conhecimento. | Desejável |
| RF19 | Ocultar área de investigação | Função que permite ao coordenador ocultar uma área de investigação . | Desejável |
| RF20 | Cadastrar banca | Função que permite ao coordenador cadastrar qual será o dia e horário das bancas, além de selecionar quem fará parte da banca. Após o cadastro da banca. | Importante |
| RF21 | Atualizar banca | Função que permite ao orientador actualizar qual será o dia e horário das bancas, além de atualizar quem participará da banca | Desejável |
| RF22 | Visualizar banca | Função que permite ao estudante/docente visualizar quais são os docentes que compõe uma banca | Importante |
| RF23 | Lançar Nota | Função que permite ao Coordenador lançar a nota final do Estudante, após ser lançada, o estado do trabalho é definido como “Trabalho Defendido”. | Importante |
| RF24 | Pesquisar usuário | Função que permite ao gestor pesquisar usuário | Importante |

Tabela 3. 1 Requisitos Funcionais

3.5.2 Requisitos não funcionais

Os Requisitos não funcionais estão relacionados ao uso da aplicação em termos de segurança, usabilidade, desempenho, disponibilidade e portabilidade. Para o nosso sistema extraiu-se os seguintes requisitos não funcionais:

| | |
|-----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Segurança | <ul style="list-style-type: none">• A integridade e confidencialidade das informações são asseguradas através de mecanismos de controlo de acesso de usuários não autorizados, através de senha e definição de acesso para cada usuário, de modo que cada um pode ter disponível somente a atividade relacionada a ele; |
| | <ul style="list-style-type: none">• O sistema garante que a exclusão de informações emita uma opção de aviso antes de executar a ação; |
| | <ul style="list-style-type: none">• A mensagem de erro exibida aos usuários é genérica, sem dar detalhes das informações, para não comprometer a segurança e a integridade dos dados; |
| | <ul style="list-style-type: none">• O canal de comunicação com o servidor de banco de dados deve ser seguro; |
| Usabilidade | <ul style="list-style-type: none">• O sistema deve ser intuitivo e fácil de navegar, qualquer usuário pode usá-lo sem ter conhecimentos avançado de informática. |
| Portabilidade | <ul style="list-style-type: none">• O sistema deve rodar em qualquer navegador; |
| Confiabilidade | <ul style="list-style-type: none">• O sistema deve ter um plano de contingência que permita a recuperação de dados;• O sistema não deve permitir a duplicidade de dados;• O sistema valida a coleta de dados para evitar entradas inapropriadas. |

Tabela 3. 2 Requisitos Não Funcionais

3.5.3 Regras de negócio do nosso sistema

As regras de negócio definem como uma empresa ou instituição opera seus negócios. Essas regras podem incluir políticas, procedimentos, restrições, requisitos, objetivos, entre outros. Em termos simples, as regras de negócio são as orientações que uma empresa segue para garantir que suas operações sejam consistentes, eficientes e alinhadas com seus objetivos e valores. As

regras de negócio podem abranger várias áreas da empresa, como finanças, recursos humanos, marketing, vendas, produção, entre outras.

| Código | Descrição |
|---------------|---------------------------------------------------------------------------------------------------|
| RN -01 | O coordenador é o responsável aprovar uma proposta de tema para o projecto de conclusão de curso; |
| RN -02 | Só é possível atribuir tarefas a uma proposta aprovada; |
| RN -03 | O estudante só pode enviar a versão finalizada do TFC após o Orientador marcar como Finalizado; |
| RN -04 | O coordenador é o responsável por criar bancas. |

Tabela 3. 3 Regras de Negócio

3.6 Diagramas UML do nosso Sistema

Neste tópico falaremos sobre os diagramas UML usados para a elaboração do sistema.

3.6.1 Diagrama de Casos de Uso

O Diagrama de Casos de Uso tem o objectivo de auxiliar a comunicação entre os analistas e o cliente. Um diagrama de Caso de Uso descreve um cenário que mostra as funcionalidades do sistema do ponto de vista do usuário. O cliente deve ver no diagrama de Casos de Uso as principais funcionalidades de seu sistema.

O diagrama de Caso de Uso é representado por:

- Actores;
- Casos de Uso.

O relacionamento entre estes elementos. pode se dar por:

- Associações entre actores e Casos de Uso;
- Generalizações entre actores;
- Generalizações, extends e includes entre os casos de uso.

Para esse sistema, como mostra a figura 6, foi criado o seguinte diagrama abaixo ilustrado, composto ele por três (3) actores, catorze (16) casos de uso, dezoito (21) relacionamentos.

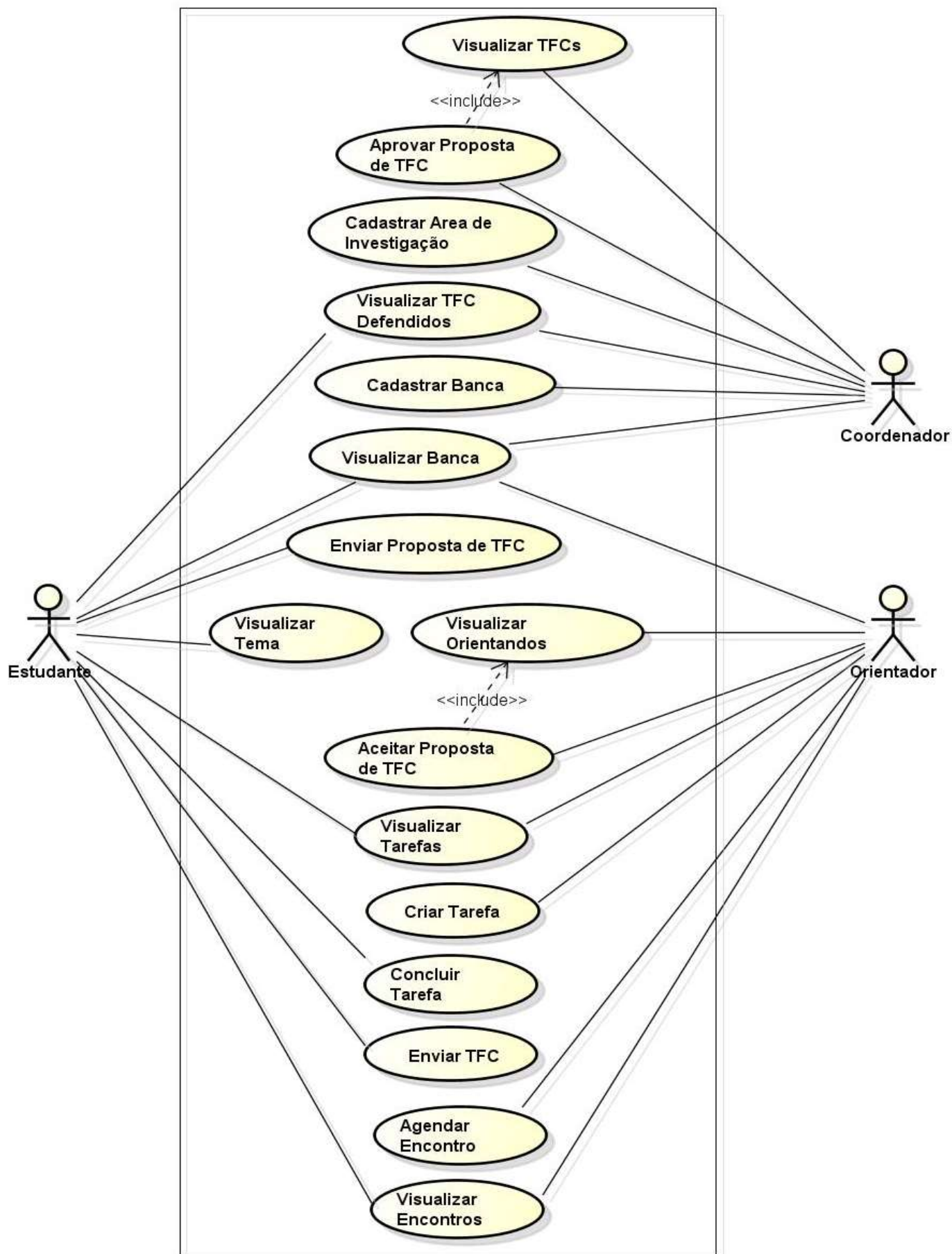


Figura 3.1 Diagrama de Caso de Uso do projecto.

3.6.2 Descrição dos principais casos de uso do nosso sistema

Na tabela 3.4 apresentamos a descrição do caso de uso de como TFC é visualizado.

| CU1 | Visualizar TFC |
|--------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Actores | <ul style="list-style-type: none">• Coordenador; |
| Pré-Condição | <ul style="list-style-type: none">• Estar logado; |
| Pós- Condição | <ul style="list-style-type: none">• Os temas deverão ser visualizados; |
| Fluxo Principal | <ul style="list-style-type: none">• Solicitar página de Proposta de TFC;• Pesquisar tema;• Processar e validar dados da pesquisa;• Apresentar tema pesquisado ao utilizador. |
| Fluxo Alternativo | Erro encontrado no ponto 3: Tema não encontrado; <ul style="list-style-type: none">• Informar o utilizador sobre o erro e voltar ao ponto 2. |

Tabela 3.4 Descrição do Caso de Uso Visualizar TFC

Na tabela 3.5 apresentamos a descrição do caso de uso de como uma Proposta de TFC é aprovado.

| CU2 | Aprovar Proposta de TFC |
|--------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Actores | <ul style="list-style-type: none">• Coordenador; |
| Pré-Condição | <ul style="list-style-type: none">• Estar logado; |
| Pós- Condição | <ul style="list-style-type: none">• O tema deverá ser aprovado; |
| Fluxo Principal | <ul style="list-style-type: none">• Solicitar página de Propostas de TFC;• Selecionar o Tema;• Aprovar Proposta;• Processar e validar dados;• Informar o usuário sobre a operação. |
| Fluxo Alternativo | Erro encontrado no ponto 3: Tema não Aprovado; <ul style="list-style-type: none">• Informar o utilizador sobre o erro e voltar ao ponto 2. |

Tabela 3.5 Descrição do Caso de Uso Aprovar Proposta de TFC

Na tabela 3.6 apresentamos a descrição do caso de uso cadastrar área de investigação.

| CU3 Cadastrar Área de Investigação | |
|------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Actor | Coordenador |
| Pré-Condição | <ul style="list-style-type: none"> • O Usuário deve estar logado; |
| Pós-condição | <ul style="list-style-type: none"> • A área deve ser cadastrada; |
| Fluxo Principal | <ul style="list-style-type: none"> • Selecionar Cadastrar Área de Investigação; • Inserir dados; • Validar dados; • Guardar dados, • Informar o usuário sobre a operação. |
| Fluxo Alternativo | Erro no ponto 3: Dados inválidos; <ul style="list-style-type: none"> • Informar o utilizador sobre o erro e retomar ao ponto 2. |

Tabela 3.6 Descrição do Caso de Uso Cadastrar Área de Investigação

Na tabela 3.7 apresentamos a descrição do caso de uso Visualizar TFC Defendidos.

| CU4 Visualizar TFC Defendidos | |
|-------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Actores | Estudante/Coordenadores; |
| Pré-Condição | <ul style="list-style-type: none"> • Estar logado; |
| Pós- Condição | <ul style="list-style-type: none"> • Os TFCs deverão ser visualizados; |
| Fluxo Principal | <ul style="list-style-type: none"> • Solicitar página de TFC defendidos; • Pesquisar tema; • Processar e validar dados da pesquisa; • Apresentar tema pesquisado ao utilizador. |
| Fluxo Alternativo | Erro encontrado no ponto 3: Tema não encontrado; <ul style="list-style-type: none"> • Informar o utilizador sobre o erro e voltar ao ponto 2. |

Tabela 3.7 Descrição do Caso de Uso Visualizar TFC Defendidos

Na tabela 3.8 apresentamos a descrição do caso de uso cadastrar banca.

| CU5 Cadastrar Banca | |
|--------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Actor | Coordenador |
| Pré-Condição | <ul style="list-style-type: none"> • O Usuário deve estar logado; |
| Pós-condição | <ul style="list-style-type: none"> • A banca deve ser cadastrada; |
| Fluxo Principal | <ul style="list-style-type: none"> • Selecionar Cadastrar Banca; • Inserir dados; • Validar dados; • Guardar dados, • Informar o usuário sobre a operação. |
| Fluxo Alternativo | Erro no ponto 3: Dados inválidos; <ul style="list-style-type: none"> • Informar o utilizador sobre o erro e retomar ao ponto 2. |

Tabela 3.8 Descrição do Caso de Uso Cadastrar Banca

Na tabela 3.9 apresentamos a descrição do caso de uso Visualizar Banca.

| CU6 Visualizar Banca | |
|--------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Actores | Estudante/Coordenadores/Professor; |
| Pré-Condição | <ul style="list-style-type: none"> • Estar logado; |
| Pós- Condição | <ul style="list-style-type: none"> • As Bancas deverão ser visualizadas; |
| Fluxo Principal | <ul style="list-style-type: none"> • Solicitar página de Bancas; • Pesquisar Bancas; • Processar e validar dados da pesquisa; • Apresentar tema pesquisado ao utilizador. |
| Fluxo Alternativo | Erro encontrado no ponto 3: Tema não encontrado; <ul style="list-style-type: none"> • Informar o utilizador sobre o erro e voltar ao ponto 2. |

Tabela 3.9 Descrição do Caso de Uso Visualizar Banca

Na tabela 3.10 apresentamos a descrição do caso de uso Enviar Proposta de TFC.

| CU7 | Enviar Proposta de TFC |
|--------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Actores | <ul style="list-style-type: none"> • Estudante; |
| Pré-Condição | <ul style="list-style-type: none"> • Estar logado; |
| Pós- Condição | <ul style="list-style-type: none"> • O tema deve ser enviado; |
| Fluxo Principal | <ul style="list-style-type: none"> • Selecionar Enviar Proposta de TFC; • Inserir dados; • Validar dados; • Guardar dados; • Informar o usuário sobre a operação. |
| Fluxo Alternativo | Erro encontrado no ponto 3: Dados inválidos; <ul style="list-style-type: none"> • Informar o utilizador sobre o erro e voltar ao ponto 2. |

Tabela 3. 10 Descrição do Caso de Uso Enviar Proposta de TFC

Na tabela 3.11 apresentamos a descrição do caso de uso Meus Orientandos.

| CU8 | Meus Orientandos |
|--------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Actores | Orientador; |
| Pré-Condição | <ul style="list-style-type: none"> • Estar logado; |
| Pós- Condição | <ul style="list-style-type: none"> • Os Orientandos e seus tema deverão ser visualizados; |
| Fluxo Principal | <ul style="list-style-type: none"> • Solicitar página Meus Orientandos; • Pesquisar Nome do Orientando ou Tema; • Processar e validar dados da pesquisa; • Apresentar o resultado ao utilizador. |
| Fluxo Alternativo | Erro encontrado no ponto 3: Tema não encontrado; <ul style="list-style-type: none"> • Informar o utilizador sobre o erro e voltar ao ponto 2. |

Tabela 3. 11 Descrição do Caso de Uso Meus Orientandos

Na tabela 3.12 apresentamos a descrição do caso de uso Aceitar Proposta de TFC.

| CU9 Aceitar Proposta de TFC | |
|-----------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Actores | Orientador; |
| Pré-Condição | <ul style="list-style-type: none"> • Estar logado; |
| Pós- Condição | <ul style="list-style-type: none"> • A Proposta deve Ser Aceite; |
| Fluxo Principal | <ul style="list-style-type: none"> • Solicitar pagina Meus Orientandos; • Selecionar o Tema; • Selecionar Aceitar Proposta; • Processar e validar dados; • Aceitar Proposta; • Informar o usuário sobre a operação. |
| Fluxo Alternativo | Erro encontrado no ponto 4: Tema não actualizado; <ul style="list-style-type: none"> • Informar o utilizador sobre o erro e voltar ao ponto 2. |

Tabela 3. 12 Descrição do Caso de Uso Aceitar Proposta de TFC

Na tabela 3.13 apresentamos a descrição do caso de uso Visualizar Tema.

| CU10 Visualizar Tema | |
|--------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Actores | Estudante/Orientador; |
| Pré-Condição | <ul style="list-style-type: none"> • Estar logado; |
| Pós- Condição | <ul style="list-style-type: none"> • O Tema deverá ser visualizado; |
| Fluxo Principal | <ul style="list-style-type: none"> • Solicitar página Visualizar Tema; • Selecionar Tema; • Processar e validar dados da pesquisa; • Apresentar o tema ao utilizador. |
| Fluxo Alternativo | Nenhum Tema encontrado; <ul style="list-style-type: none"> • Informar ao utilizador. |

Tabela 3.13 Descrição do Caso de Uso Visualizar Tema.

Na tabela 3.14 apresentamos a descrição do caso de uso Visualizar Tarefas.

| CU11 Visualizar Tarefas | |
|--------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Actores | Estudante/Orientador; |
| Pré-Condição | <ul style="list-style-type: none"> • Estar logado; |
| Pós- Condição | <ul style="list-style-type: none"> • As tarefas deverão ser visualizados; |
| Fluxo Principal | <ul style="list-style-type: none"> • Solicitar página de tarefas; • Pesquisar tarefa; • Processar e validar dados da pesquisa; • Apresentar tema pesquisado ao utilizador. |
| Fluxo Alternativo | Erro encontrado no ponto 3: Tema não encontrado; <ul style="list-style-type: none"> • Informar o utilizador sobre o erro e voltar ao ponto 2. |

Tabela 3.14 Descrição do Caso de Uso Visualizar Tarefas

Na tabela 3.15 apresentamos a descrição do caso de uso Criar Tarefa.

| CU12 Criar Tarefa | |
|--------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Actor | Orientador |
| Pré-Condição | <ul style="list-style-type: none"> • O Usuário deve estar logado; |
| Pós-condição | <ul style="list-style-type: none"> • A Tarefa deverá ser criada; |
| Fluxo Principal | <ul style="list-style-type: none"> • Solicitar pagina Meus Orientandos; • Selecionar Estudante/Tema; • Solicitar pagina de Tarefas do Estudante; • Selecionar Criar Tarefa; • Inserir dados; • Validar dados; • Guardar dados; • Informar o usuário sobre a operação. |
| Fluxo Alternativo | Erro no ponto 6: Dados inválidos; <ul style="list-style-type: none"> • Informar o utilizador sobre o erro e retomar ao ponto 5. |

Tabela 3.15 Descrição do Caso de Uso Criar Tarefa

Na tabela 3.16 apresentamos a descrição do caso de uso Criar Tarefa.

| CU13 Concluir Tarefa | |
|--------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Actor | Estudante |
| Pré-Condição | <ul style="list-style-type: none"> O Usuário deve estar logado; |
| Pós-condição | <ul style="list-style-type: none"> A Tarefa deverá ser concluída; |
| Fluxo Principal | <ul style="list-style-type: none"> Solicitar pagina Actividades; Selecionar a Tarefa; Selecionar Concluir Tarefa; Validar dados; Guardar dados; Informar o usuário sobre a operação. |
| Fluxo Alternativo | Erro no ponto 4: Dados inválidos; <ul style="list-style-type: none"> Informar o utilizador sobre o erro e retomar ao ponto 3. |

Tabela 3.16 Descrição do Caso de Uso Concluir Tarefa

Na tabela 3.17 apresentamos a descrição do caso de uso Enviar TFC.

| CU14 Enviar TFC | |
|--------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Actor | Estudante |
| Pré-Condição | <ul style="list-style-type: none"> O Usuário deve estar logado; |
| Pós-condição | <ul style="list-style-type: none"> O TFC deverá ser enviado; |
| Fluxo Principal | <ul style="list-style-type: none"> Solicitar pagina Enviar TFC; Anexar o Ficheiro contendo TFC. Inserir dados; Validar dados; Guardar dados; Informar o usuário sobre a operação. |
| Fluxo Alternativo | Erro no ponto 4: Dados inválidos; <ul style="list-style-type: none"> Informar o utilizador sobre o erro e retomar ao ponto 3. |

Tabela 3.17 Descrição do Caso de Uso Enviar TFC

Na tabela 3.18 apresentamos a descrição do caso de uso Agendar Encontro.

| CU15 Agendar Encontro | |
|--------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Actor | Orientador |
| Pré-Condção | <ul style="list-style-type: none"> • O Usuário deve estar logado; |
| Pós-condção | <ul style="list-style-type: none"> • O Encontro deverá ser Agendado; |
| Fluxo Principal | <ul style="list-style-type: none"> • Solicitar pagina Encontros; • Selecionar Agendar Encontro; • Selecionar Estudante; • Inserir dados; • Validar dados; • Guardar dados; • Informar o usuário sobre a operação. |
| Fluxo Alternativo | Erro no ponto 4: Dados inválidos; <ul style="list-style-type: none"> • Informar o utilizador sobre o erro e retomar ao ponto 3. |

Tabela 3.18 Descrição do Caso de Uso Agendar Encontro

Na tabela 3.19 apresentamos a descrição do caso de uso Visualizar Encontros.

| CU16 Visualizar Encontros | |
|---------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Actor | Estudante/Orientador |
| Pré-Condção | <ul style="list-style-type: none"> • O Usuário deve estar logado; |
| Pós-condção | <ul style="list-style-type: none"> • O Encontro deverá ser Visualizado; |
| Fluxo Principal | <ul style="list-style-type: none"> • Solicitar pagina Encontros; • Processar e validar dados da solicitado; • Apresentar os encontros pesquisado ao utilizador. • Informar o usuário sobre a operação. |
| Fluxo Alternativo | Nenhum Encontro Cadastrado; <ul style="list-style-type: none"> • Informar o utilizador. |

Tabela 3.19 Descrição do Caso de Uso Visualizar Encontros

3.6.3 Diagrama de Actividades

Em linguagem UML é um diagrama de actividade que representa os fluxos do trabalho passo-à-passo do negócio e operacionais dos componentes em um sistema. Ele mostra o fluxo de controlo geral. Para este sistema fez-se os seguintes diagramas de Actividades representados pelos principais casos de uso do diagrama de casos de uso.

A figura 3.2 é referente a um diagrama aonde representamos o fluxo de actividades necessários para a realização do caso de uso Cadastrar Área de Investigação.

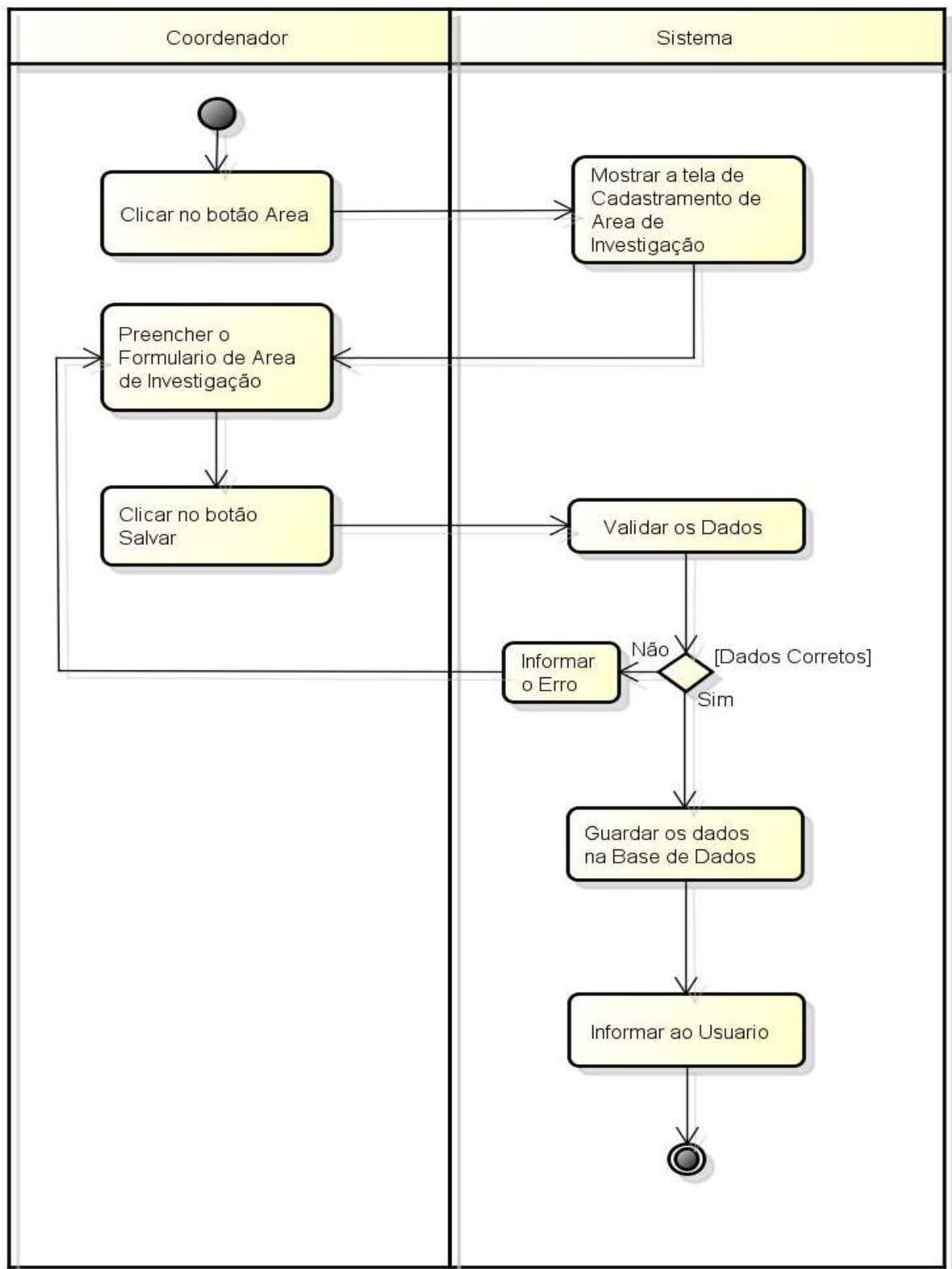


Figura 3.2 Diagrama de Actividade Área de Investigação.

A figura 3.3 é referente a um diagrama aonde representamos o fluxo de actividades necessários para a realização do caso de uso Enviar Proposta de TFC.

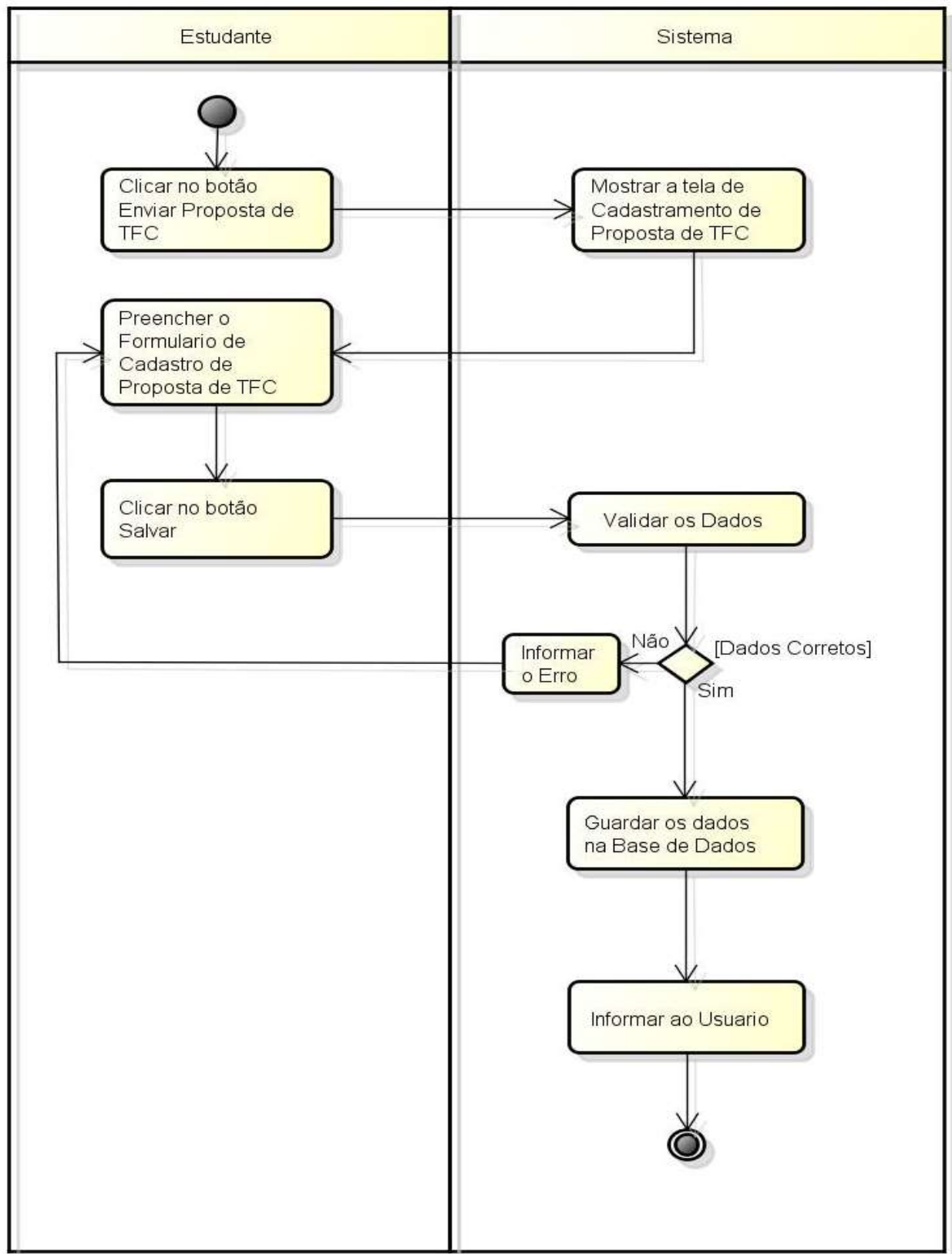


Figura 3.3 Diagrama de Actividade Enviar Proposta de TFC.

3.6.4 Diagrama de Sequência

Os diagramas de sequência permitem representar o comportamento dinâmico do sistema, nomeadamente as interacções entre objectos. Os diagramas de sequência são uma das formas preferenciais de especificar os casos de uso. O diagrama consiste essencialmente na representação de sequência de chamada de métodos (troca de mensagens) entre os objectos das classes que suportam a aplicação.

Desta forma temos abaixo ilustrado alguns digramas de sequência que representam processos relevantes no nosso sistema.

Na figura 3.4 é representado a sequência de processos para que se realize o caso de uso Enviar Proposta de TFC.

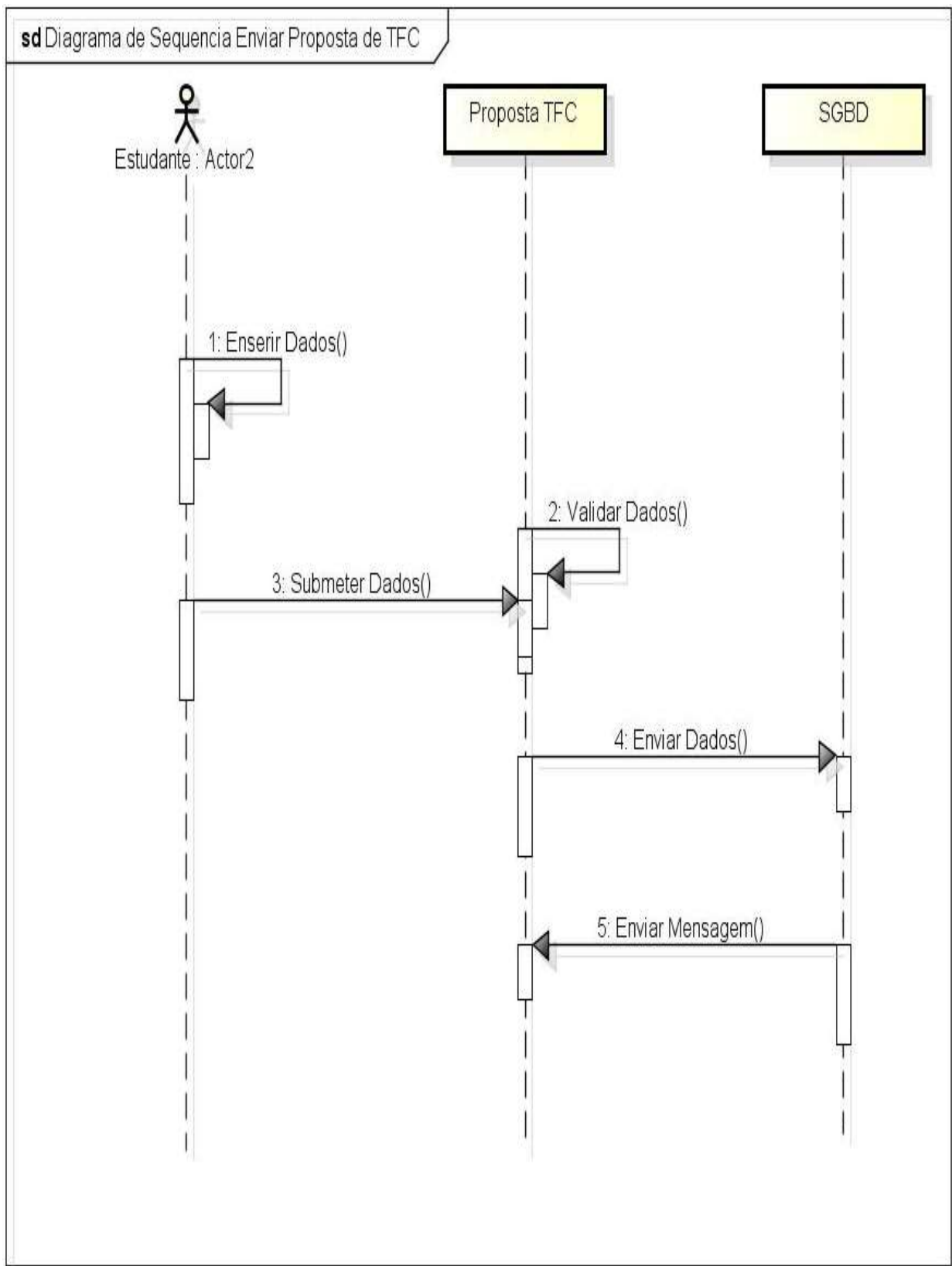


Figura 3.4 Diagrama de Sequência Enviar Proposta de TFC.

Na figura 3.5 é representado a sequência de processos para que se realize o caso de uso Visualizar um Tema.

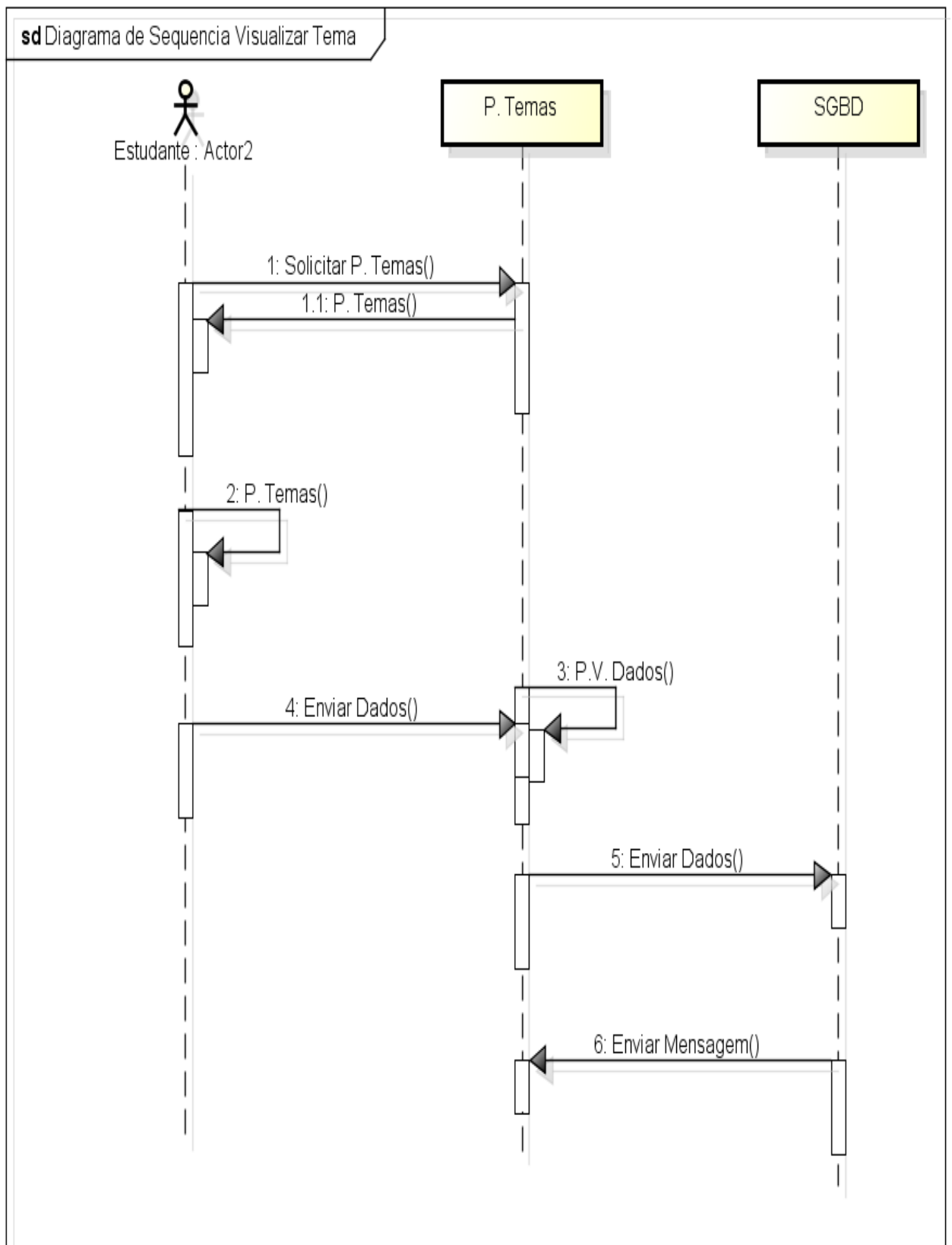


Figura 3.5 Diagrama de Sequência Visualizar Tema.

3.6.5 Diagrama de Classe

Os diagramas de classe nos permitem denotar o conteúdo estático e os relacionamentos de classes. Em um diagrama de classe, também podemos mostrar se uma classe herda de outra ou se contém uma referência para outra. Em resumo, podemos retratar todas as dependências de código-fonte entre as classes.

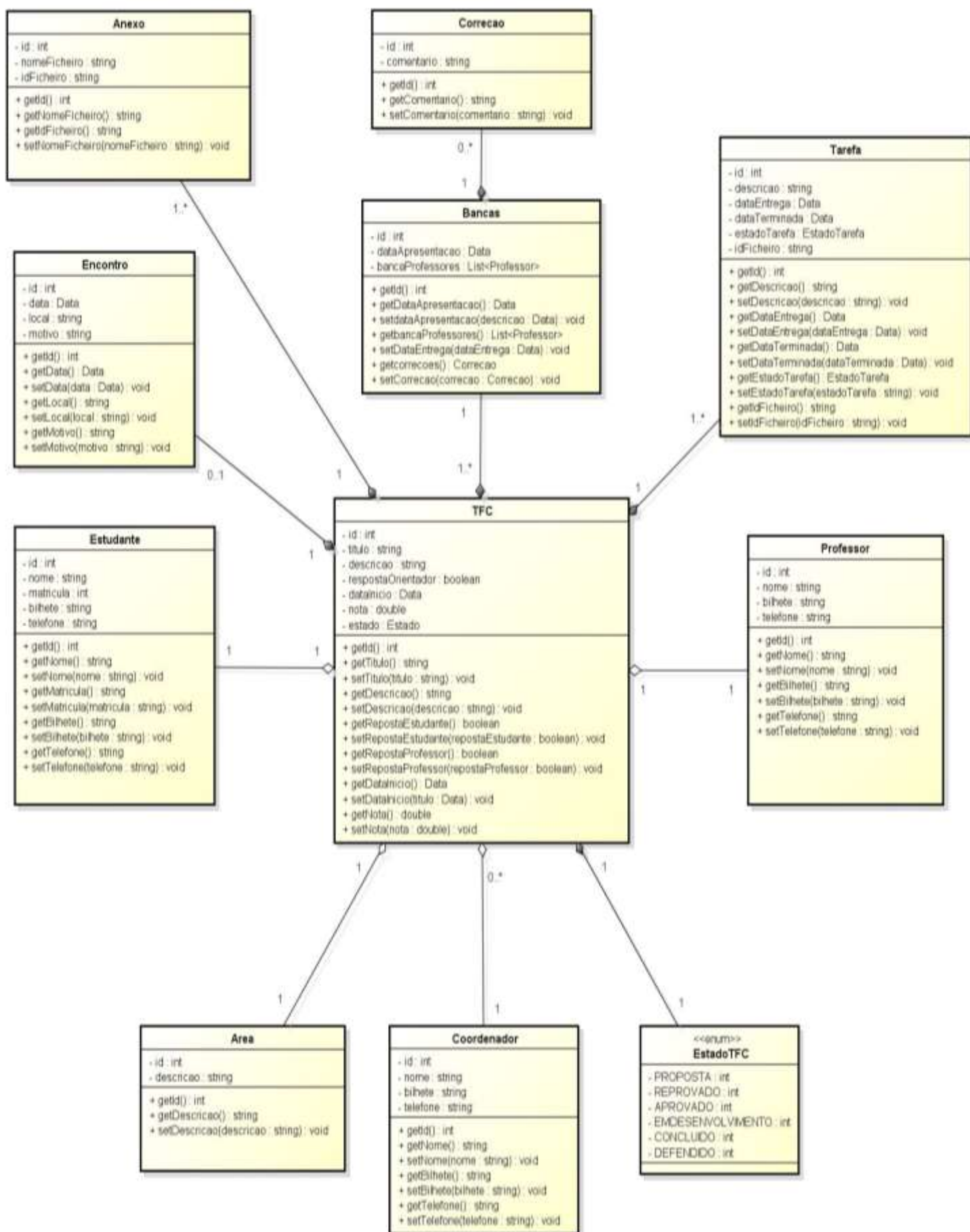


Figura 3.6 Diagrama de Classe

3.5.7. Diagrama de Instalação

A figura 3.7 apresenta o diagrama de como o sistema poderá ser instalado. A base de dados será alojada num servidor de base de dados com o PostGreSQL 15 e a aplicação deverá estar instalado num servidor web IIS a comunicação entre o servidor de base de dados e do servidor web deverá ser feito através do protocolo TCP/IP pois os mesmos estão alojados na mesma rede. O cliente poderá ter acesso a aplicação através de browser com requisições http.

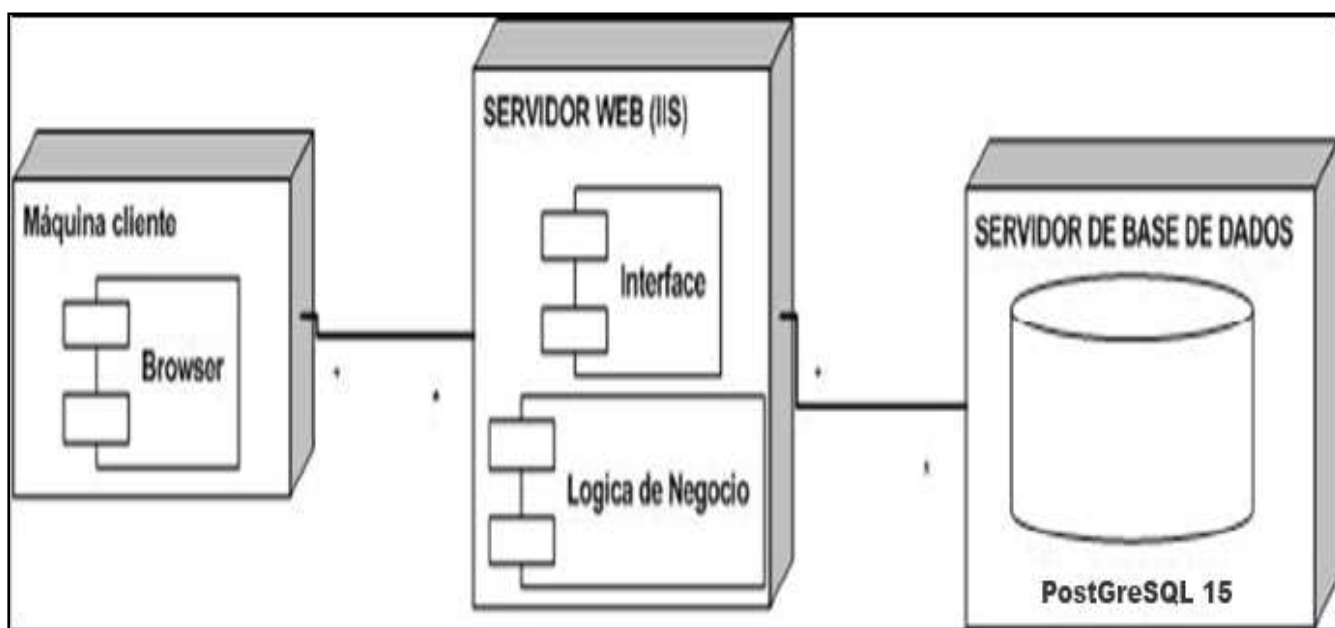


Figura 3.7 Diagrama de instalação do sistema.

3.5.8. Modelo conceitual

Denominamos entidade estes elementos. Atribuímos a cada entidade definidos atributos pertinentes ao sistema. Desta forma, podemos representar conceitualmente como entidades aqueles elementos no qual gostaríamos de armazenar dados que por sua vez, através do relacionamento representaremos o tipo de relação existente entre as entidades, logo a seguir na figura 3.8.

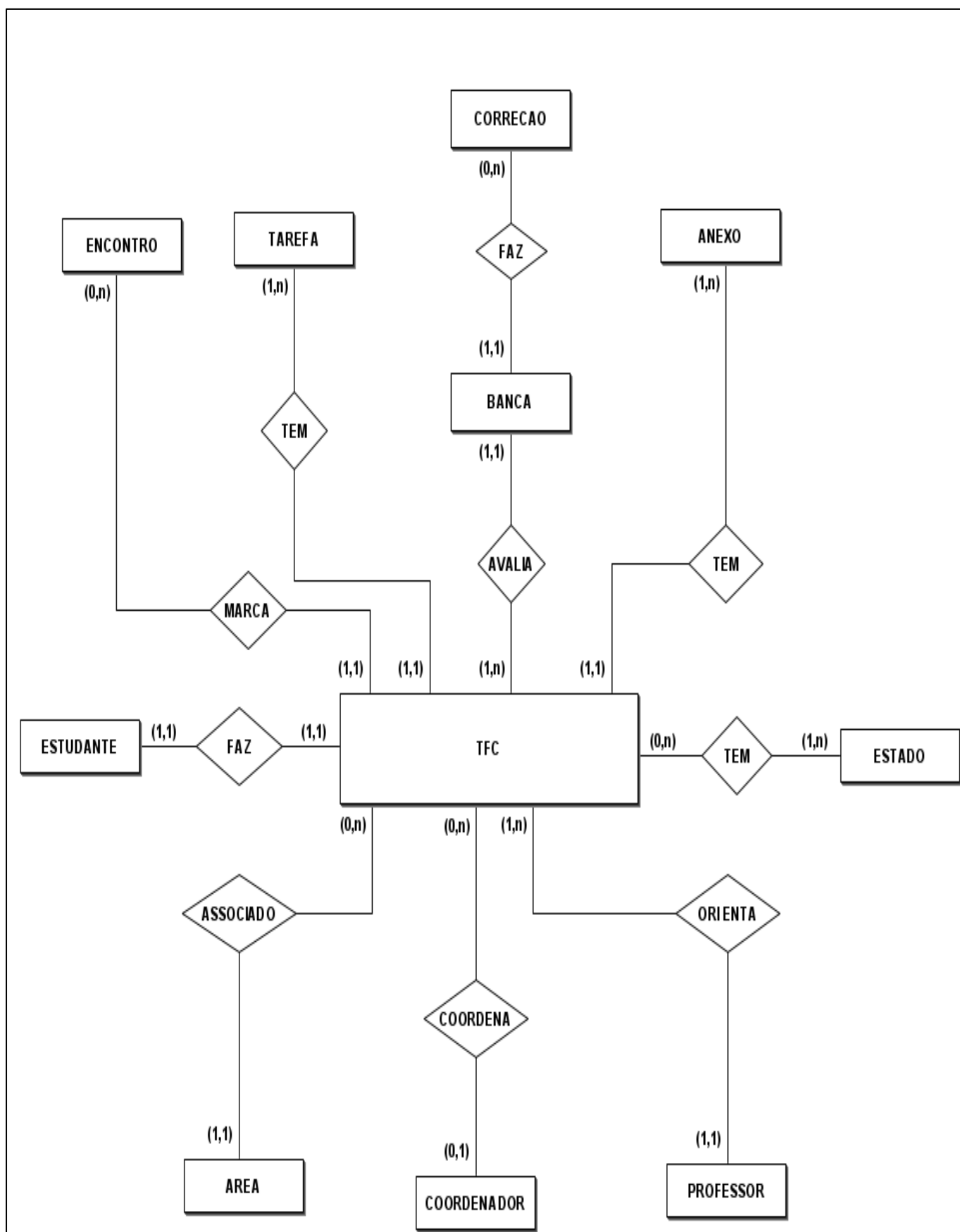


Figura 3.8 Modelo Entidade Relacionamento

3.5.9. Modelo lógico

Na figura abaixo apresentamos o diagrama lógico que compreende uma descrição das estruturas que serão armazenadas no banco e que resulta numa representação gráfica dos dados de uma maneira lógica, inclusive nomeando os componentes e ações que exercem uns sobre os outros

A relação entre a tabela Area e a tabela TFC: A relação entre a tabela Area e a tabela TFC é de 1..n (lê-se um para muitos) sendo que a coluna IdArea da tabela Area, é uma chave estrangeira (ForeignKey - FK) na tabela TFC. Partindo do pressuposto que uma área de conhecimento pode ser atribuído a várias propostas de TFC.

A relação entre a tabela TFC e tabela Tarefa: A relação entre estas tabelas é de 1..n, sendo que a coluna IdTFC é chave estrangeira (ForeignKey- FK) na tabela Tarefa. Pois um TFC pode ter várias tarefas.

A relação entre a tabela Banca e tabela TFC: A relação elas é de 1..n, sendo que a coluna IdBanca da tabela Banca é chave estrangeira (ForeignKey – FK) na tabela TFC. Pois uma banca avalia vários TFCs.

A relação entre a tabela TFC e tabela Encontro: Estas têm uma relação de 1..n, sendo que o IdTFC da tabela Estudante é chave estrangeira (ForeignKey – FK) na tabela Encontro. Pois em um TFC pode se marcar vários encontros.

A relação entre a tabela Professor e a tabela TFC: elas têm a relação de 1..n, sendo que o IdProfessor da tabela Professor é chave estrangeira (ForeignKey) na tabela TFC. Pois um professor pode orientar vários TFCs.

A relação entre a tabela Professor e a tabela Banca: elas estão relacionadas de n..n, ou seja muitos para muitos gerando uma nova tabela, sendo que o IdProfessor da tabela Professor e o IdBanca são chaves estrangeiras (ForeignKey) na tabela BancaProfessores. Sendo que um professor pode pertencer várias bancas e uma banca pode ter vários professores.

Na figura 3.9 veremos o diagrama Lógico para o nosso sistema:

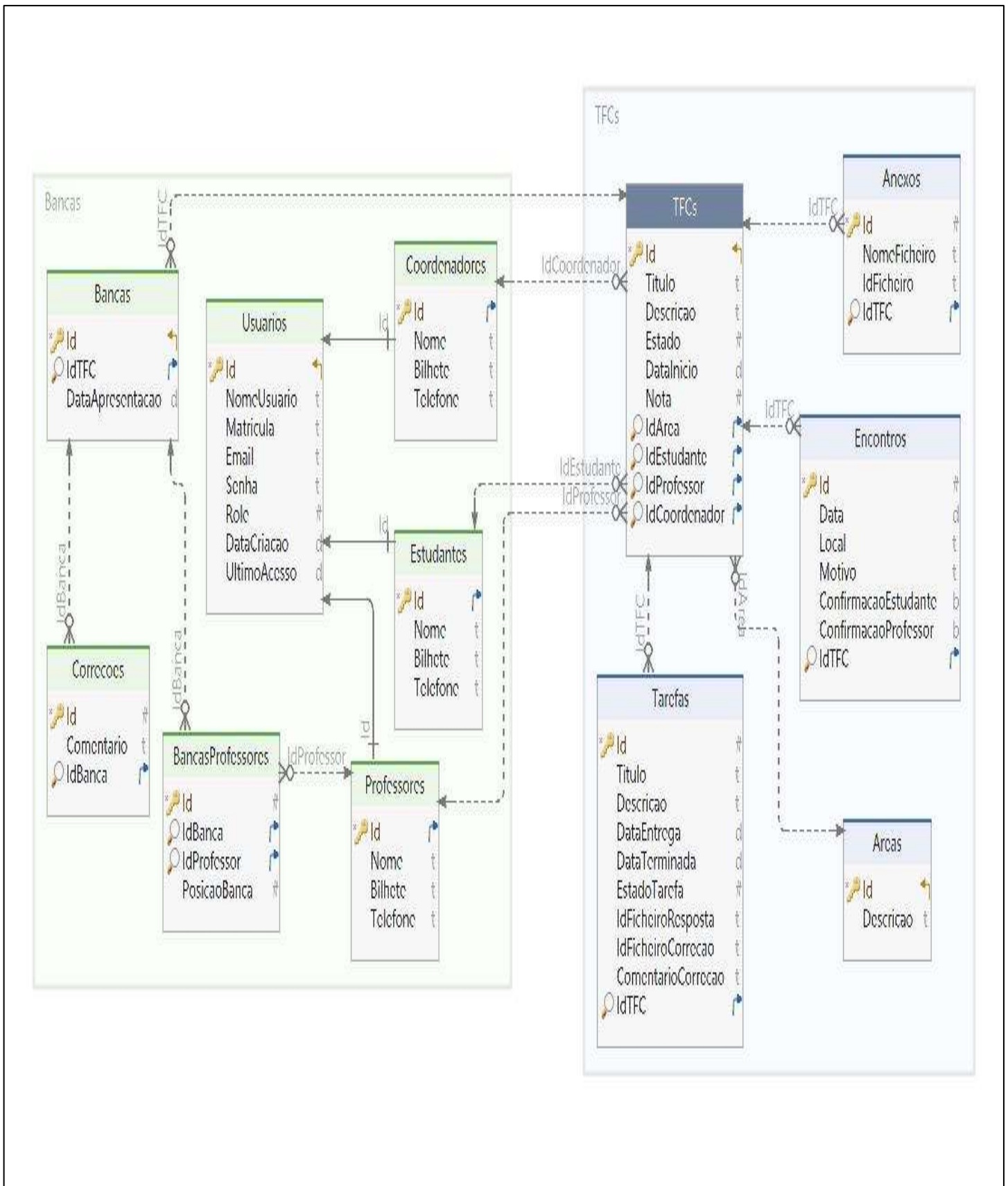


Figura 3.9 Diagrama Lógico

3.6. Ferramentas utilizadas

Para a modulação deste projecto usamos as seguintes ferramentas Microsoft como: ASP.NET CORE, C#, PostgreSQL, RabbitMQ, Angular.

- **O Visual Studio** - é um ambiente de desenvolvimento integrado (IDE) amplamente utilizado para criar aplicativos de software para uma variedade de plataformas, incluindo Windows, Android, iOS, Linux e serviços em nuvem. Ele oferece uma gama de ferramentas e recursos para facilitar o desenvolvimento, depuração e implantação de software, juntamente com suporte para várias linguagens de programação, como C#, Visual Basic, C++, Python, JavaScript e muito mais.

O Visual Studio inclui um conjunto abrangente de recursos, como editores de código avançados, depuradores poderosos, designers de interface gráfica, ferramentas de teste, integração com sistemas de controle de versão, suporte a desenvolvimento web e móvel, e uma ampla gama de extensões disponíveis para personalização e aprimoramento do ambiente de desenvolvimento.

- **PostgreSQL** - é um sistema de gerenciamento de banco de dados relacional de código aberto extremamente poderoso e robusto. Ele oferece recursos avançados de armazenamento, manipulação e consulta de dados, tornando-o uma escolha popular para uma variedade de aplicativos, desde pequenas aplicações de uso interno até grandes sistemas corporativos.

O PostgreSQL é conhecido por sua conformidade com os padrões SQL ANSI e sua capacidade de suportar recursos avançados, como transações ACID (Atomicidade, Consistência, Isolamento e Durabilidade), procedimentos armazenados, gatilhos, índices, replicação e muitos outros. Além disso, ele é altamente extensível, permitindo aos usuários criar e integrar suas próprias extensões e módulos.

- **ASP.NET Core** - é um framework de desenvolvimento web de código aberto, multiplataforma e de alto desempenho, desenvolvido pela Microsoft. Ele permite a criação de aplicativos web modernos e escaláveis usando a linguagem de programação C# e o modelo de programação MVC (Model-View-Controller) ou o modelo de programação Razor Pages.

ASP.NET Core oferece uma série de recursos poderosos, incluindo roteamento flexível, injeção de dependência integrada, middleware configurável, suporte para APIs RESTful, autenticação e autorização robustas, integração com serviços em nuvem, e

muito mais. Ele é projetado para ser leve, modular e altamente extensível, facilitando o desenvolvimento e a implantação de aplicativos web em diversas plataformas, incluindo Windows, Linux e macOS.

- **C#** - é uma linguagem de programação orientada a objetos, desenvolvida pela Microsoft como parte da plataforma .NET. Ela foi projetada para ser simples, moderna, segura e de alto desempenho, tornando-a adequada para uma ampla gama de aplicativos, desde aplicativos de desktop até aplicativos web e móveis.

C# oferece recursos poderosos, como tipagem estática, coleta de lixo automática, herança, polimorfismo, delegados, eventos, expressões lambda, e muito mais. Ela também é altamente integrada com a plataforma .NET, permitindo acesso fácil a uma ampla gama de bibliotecas e estruturas de software para desenvolvimento rápido e eficiente de aplicativos.

- **RabbitMQ** - é um sistema de mensageria de código aberto amplamente utilizado para a troca de mensagens entre aplicativos distribuídos. Ele implementa o protocolo Advanced Message Queuing Protocol (AMQP), que é um padrão aberto para sistemas de mensageria assíncrona.

RabbitMQ permite que aplicativos enviem mensagens entre si de forma confiável, mesmo quando estão em execução em diferentes plataformas e linguagens de programação. Ele oferece recursos avançados, como filas de mensagens, trocas, roteamento de mensagens, confirmações de entrega, persistência de mensagens e muito mais, tornando-o uma escolha popular para a implementação de arquiteturas orientadas a mensagens em ambientes distribuídos.

- **Angular** - é um framework de desenvolvimento de aplicativos web de código aberto mantido pelo Google. Ele é amplamente utilizado para criar aplicativos web single-page (SPAs) dinâmicos e escaláveis. Angular utiliza a linguagem TypeScript para o desenvolvimento, o que traz recursos adicionais de orientação a objetos e tipagem estática ao JavaScript, facilitando a manutenção e o desenvolvimento de aplicativos mais robustos.

Angular oferece uma arquitetura baseada em componentes, onde os aplicativos são construídos a partir de componentes reutilizáveis que encapsulam lógica, dados e apresentação. Ele também inclui uma ampla gama de recursos, como roteamento, gerenciamento de formulários, injeção de dependência, comunicação com servidores por meio de HTTP, e muito mais.

4. RESULTADOS

4.1 Descrição do projecto

Como já mencionamos, este projecto consiste em desenvolver um sistema de gestão de Trabalho de Conclusão de Curso para Universidade Técnica de Angola, e vai portanto permitir que os estudantes possam consultar temas para a monografia, solicitarem aprovação de tema que queiram defender ou seja propor tema para defesa, bem como escolher o seu orientador e poder interagir com ele no processo de elaboração bem como trocar arquivos sem sair da sua residência, a partir de um computador, com um browser, e que tenha acesso a internet.

Procuramos seguir todos requisitos que são exigidos pelo sistema de gestão. No nosso projecto procuramos integrar elementos programáticos de alto nível, no que tange ao designer procuramos seguir os padrões de usabilidade, de modo a tornar a experiência do usuário mais fácil.

O principal foco do sistema é permitir o feedback entre os estudantes e os professores da UTANGA. Categorizamos os usuários por perfis, de modo a fazer a separação de privilégio.

4.2 Interface do Sistema

4.2.1 Página Inicial do nosso sistema

A figura 18 mostra a página inicial do sistema, pagina esta que estará disponibilizada à todo usuário. Onde ele poderá visualizar temas, ver as actividades por fazer, editar seu perfil.



Figura 4. 1 Página inicial

4.2.2 Página de Cadastro de Proposta

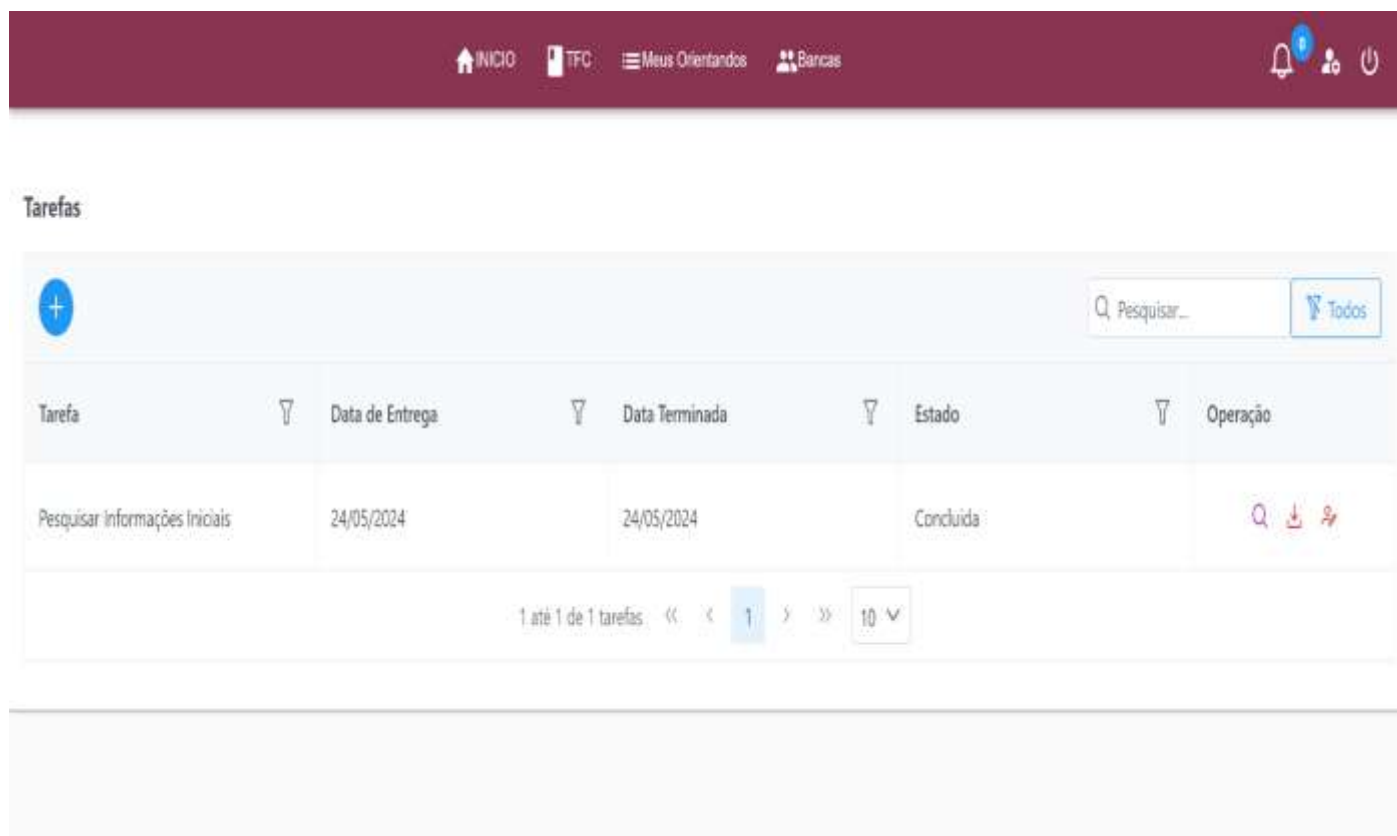
A figura 4.2 mostra-nos como os Cadastro de Proposta são feito, função essa que é executada por um estudante ou professor.



Figura 4. 2 Cadastro de Proposta

4.2.3 Página de Atividades

A figura 4.3 mostra como as tarefas são visualizadas quais os procedimentos a seguir para que um usuário possa visualizar as mesmas, função essa em que pode ser executado pelo o usuário autenticado.






| Tarefa | Data de Entrega | Data Terminada | Estado | Operação |
|--------------------------------|-----------------|----------------|-----------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Pesquisar Informações Iniciais | 24/05/2024 | 24/05/2024 | Concluída |    |

Figura 4. 3 Visualizar Tarefas

4.2.4 Página de Login

A figura 4.4 mostra o login para acessar o sistema, que apenas utilizadores cadastrados poderão executar essa funcionalidade.



Figura 4. 4 Login

4.2.5 Página de Cadastro de Área Investigação

A figura 4.5 mostra o cadastro de área de investigação, que é associado a uma proposta que já foi acima mostrado.

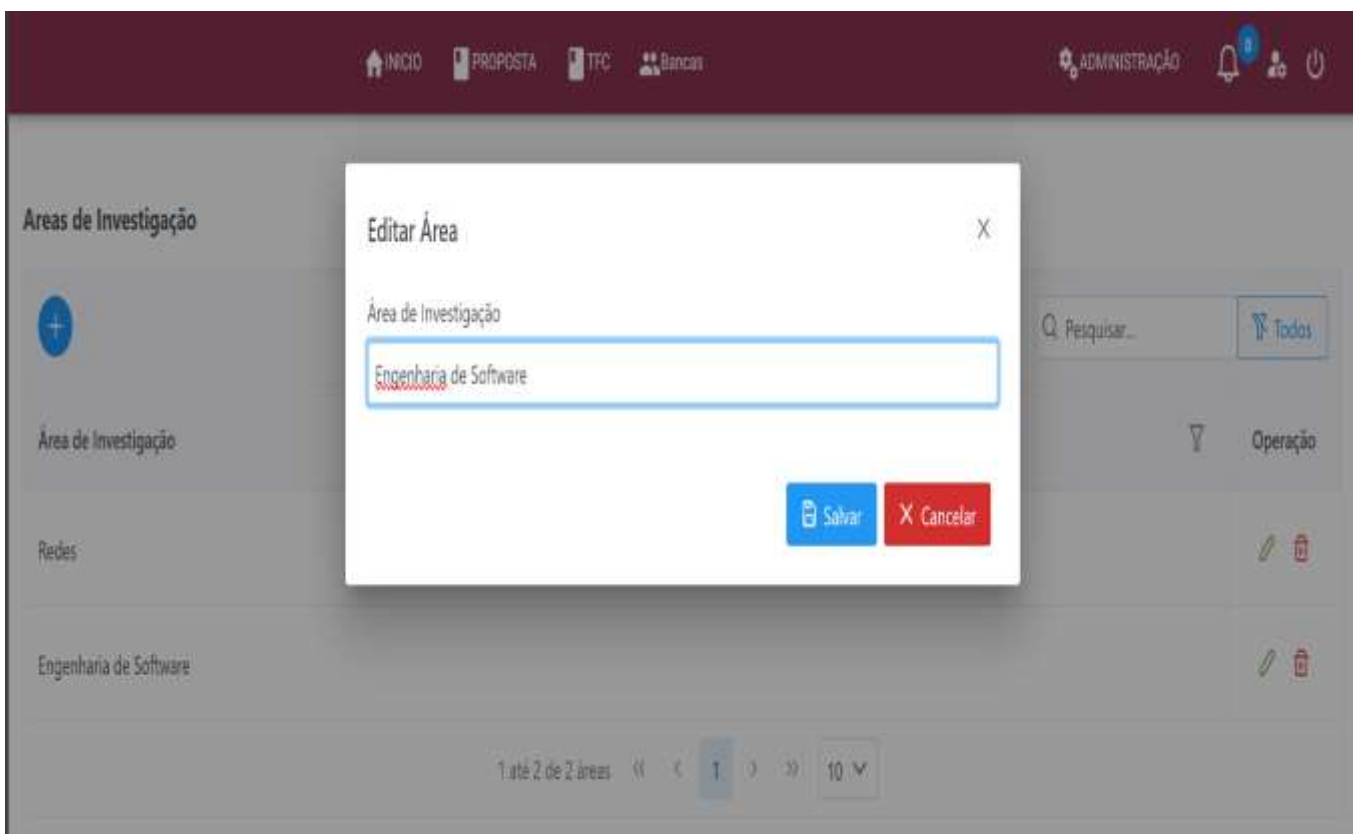


Figura 4. 5 Cadastro Área de Investigação

5 CONCLUSÃO

Neste trabalho, desenvolvemos um Sistema de Gestão de Trabalho de Fim de Curso (TFC) que visa otimizar e facilitar o gerenciamento de TFCs na UTANGA. Ao longo deste projeto, abordamos os desafios enfrentados pelos estudantes, orientadores e coordenadores na gestão eficiente dos TFCs, e propusemos soluções para superar esses obstáculos.

Uma das principais características do nosso sistema é a sua interface intuitiva e amigável, projetada para atender às necessidades dos diversos usuários envolvidos no processo de elaboração e acompanhamento dos TFCs. A integração de funcionalidades como gerenciamento de prazos, acompanhamento do progresso, comunicação entre os envolvidos e armazenamento seguro de documentos torna o processo mais eficiente e transparente.

É importante ressaltar que, embora nosso sistema represente um avanço significativo na gestão de TFCs, ainda há espaço para melhorias e aprimoramentos futuros. Sugestões como a incorporação de funcionalidades adicionais, a expansão para outras plataformas e a integração com sistemas acadêmicos já existentes podem enriquecer ainda mais a experiência dos usuários e maximizar os benefícios do sistema.

Em suma, o Sistema de Gestão de TFC desenvolvido neste trabalho oferece uma solução eficaz e inovadora para os desafios enfrentados pela UTANGA na gestão de trabalhos de fim de curso. Esperamos que este sistema contribua para a melhoria contínua dos processos acadêmicos e facilite a jornada dos estudantes rumo à conclusão de seus cursos.

6 RECOMENDAÇÕES

Recomendamos que em próximas versões sejam adicionadas mais funcionalidades ao sistema como:

- Implementação de alertas de notificações usando a RabbitMQ ou SignalR para troca de mensagens em tempo real.
- Criação de uma secção para publicações de noticias relacionadas aos Trabalhos de Fim de Curso.

7 REFERÊNCIAS BIBLIOGRÁFICAS

1. Abreu, Luís – ASP.NET 4.5 Curso Completo, “Editora FCA”, 2013;
2. Battisti, Julio – ASP.NET, 2003;
3. Damas, L. D. (2005). SQL Edição 13. FCA.
4. LARMAN, Craig “UML y Patrones”, Prentice Hall Iberoamericana, 1999.
5. PRESSMAN, R. S. (2003). Engenharia De Software Edição 2003. São Paulo (BR).
6. MELO, Ana Cristina. **Desenvolvendo Aplicações com UML 2.0 – do conceitual à implementação** – 2a. Rio de Janeiro: Brasport, 2004.
7. Nunes, M., O'Neill, H. – Fundamental de UML, “Editora FCA”, 2003;
8. <http://www.automacaoerotica.blogspot.com>. acessado aos 10.06.2017..
9. <http://www.dsc.ufcg.edu.br>. acessado aos 12.06.2017.
10. <http://www.oficinaintegrada.com>. acessado aos 12.06.2017.
11. <http://www.omecanico.com.br/oficina-organizada>. acessado aos 03.06.2017.
12. Edmundo Fernandes. 2014. O impacto dos computadores na sociedade. Tecnologia em Geral. [Online] Tecnologia em Geral, 2014. [Citação: 17 de 05 de 2018.] <http://tecnologiaemgeral.blogspot.com/2014/10/o-impacto-dos-computadores-na-sociedade.html>;
13. BeCode. 2016. Manifesto Ágil e as TOP 3 Metodologias Ágeis de Desenvolvimento! BeCode. [Online] BeCode, 2016. [Citação: 28 de 05 de 2018.] <https://becode.com.br/manifesto-agil-e-top-metodologias-ageis/>.
14. <http://blog.takao.com.br/softwares-de-gestao-que-podem-ajudar-sua-oficina/> : acessado aos 21/11/2017;
15. <http://professores.dcc.ufla.br/~bruno/aulas/livro-normal.pdf>: Acessado aos 28/12/2017.

16. <http://walderson.com/site/wp-content/uploads/2015/07/DocumentodeRegrasdeNegocio-Exemplo.pdf>: acessado aos 28/12/2017;
17. Holme, Lars; Johnson, Patrick; Bruney, Mike. "Professional Visual Studio 2019." Wrox, 2019;
18. Sempf, Bill; Jacobson, Chuck. "Visual Studio 2019 For Dummies." For Dummies, 2019;
19. Krosing, Hannu; Ties Mertens, Jim; O'Hara, Dan; Treat, Bruce. "PostgreSQL 9.0 High Performance." Packt Publishing, 2010;
20. Hsu, Regina O.; Rebello, Gregory L. "Practical PostgreSQL." O'Reilly Media, 2002;
21. Freeman, Adam. "Pro ASP.NET Core MVC 2." Apress, 2017;
22. Dewey, Dustin. "ASP.NET Core in Action." Manning Publications, 2018;
23. Troelsen, Andrew; Japikse, Philip. "Pro C# 7: With .NET and .NET Core." Apress, 2017;
24. Videla, Alvaro; Williams, Jason. "RabbitMQ in Action." Manning Publications, 2012;
25. Ponge, Alexis Richardson; Alvaro, Videla. "RabbitMQ in Depth." Manning Publications, 2017;

26. Freeman, Adam. "Pro Angular 9: Build Powerful and Dynamic Web Apps." Apress, 2020;
27. Pavithra, S.; Kumar, A. Krishna. "Mastering Angular: A deep dive into concepts, modules, tools, and libraries to build modern web applications." Packt Publishing, 2017.
28. https://fit.faccat.br/~guto/artigos/Artigo_Paradigmas_de_Programacao.pdf:
acessado aos 28/12/2017;
29. msdn.microsoft.com/pt-br/library/dd409377.aspx. acessado aos 15.06.2017.
30. Sistema de Gestão, disponível em ><https://www.guiaempreendedor.com/clico-responde-o-que-e-sistema-de-gestao/>> acessado aos 06/12/2017;
31. Sistemas de Gestão de Serviços, Disponível em <https://www.erpflex.com.br/sistema-de-gestao-servicos> acessado aos 26/12/2017;