UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO

CENTRO TECNOLÓGICO

COLEGIADO DO CURSO DE CIÊNCIA DA COMPUTAÇÃO

Murilo Borghardt Scalser

SNOPI : UM SISTEMA DE INTERFACE ADAPTATIVA BASEADA EM ONTOLOGIA

VITÓRIA

2022

Murilo Borghardt Scalser

SNOPI : UM SISTEMA DE INTERFACE ADAPTATIVA BASEADA EM ONTOLOGIA

Monografia apresentada ao Curso de Ciência da Computação do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do Grau de Bacharel em Ciência da Computação.

Universidade Federal do Espírito Santo - UFES
Centro Tecnológico
Colegiado do Curso de Ciência da Computação

Orientador: Monalessa Perini Barcellos
Coorientador: Alexandre Adler Cunha de Freitas

VITÓRIA
2022

Murilo Borghardt Scalser

SNOPI : Um Sistema De Interface Adaptativa Baseada Em Ontologia / Murilo Borghardt Scalser. – Vitória, ES, 2022 - 62 p. : 30 cm.

Orientadora: Monalessa Perini Barcellos

Monografia (PG) – Universidade Federal do Espírito Santo – UFES Centro Tecnológico Colegiado do Curso de Ciência da Computação, 2022.

1. Interface Adaptativa. 2. Ontologia. I. Barcellos, Monalessa Perini. II. Universidade Federal do Espírito Santo. IV. SNOPI : Um Sistema De Interface Adaptativa Baseada Em Ontologia

CDU 02:141:005.7

Murilo Borghardt Scalser

SNOPI : UM SISTEMA DE INTERFACE ADAPTATIVA BASEADA EM ONTOLOGIA

Monografia apresentada ao Curso de Ciência da Computação do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do Grau de Bacharel em Ciência da Computação.

Trabalho aprovado. Vitória, ES, 24 de março de 2022:

Profa. Monalessa Perini Barcellos, D. Sc.

Profa. Patrícia Dockhorn Costa, Ph. D.

Murillo Vasconcelos H. B. Castro, M. Sc.

Vitória, ES

2022

Aos meus pais, Tânia e Paulo, por nunca terem medido esforços
para me proporcionar um ensino de qualidade e que
estiveram sempre ao meu lado, a vida inteira.

AGRADECIMENTOS

Primeiramente agradeço aos meus pais, Tânia Borghardt e Paulo Scalser que, apesar de todas as dificuldades, me ajudaram e sempre apoiaram a realização do meu sonho. Também agradeço ao meu irmão, Guilherme, que sempre me incentivou durante todos os anos da faculdade.

Agradeço aos amigos, Ana Luiza Barros, Carolina Lins, Raissa Olegário e Thamyris dos Santos, que sempre estiveram ao meu lado, pela amizade incondicional e pelo apoio demonstrado ao longo de todo o período de tempo em que me dediquei a este trabalho. Sem vocês seria muito mais difícil terminar esta caminhada.

Não tenho palavras para agradecer as várias amizades que fiz durante minha graduação. Mas em especial agradeço a Carolina Manso por ser minha melhor dupla de trabalho, melhor dupla de estudos, melhor dupla para jogos, melhor amiga pra curtir e compartilhar os momentos de alegria e tristeza, melhor amiga pra minha vida que a graduação me proporcionou.

Agradeço a todos os meus professores do curso de Ciência da Computação da Universidade Federal do Espírito Santo (UFES) pela excelência da qualidade técnica de cada um. Agradeço também ao grupo NEMO (Núcleo de Estudos em Modelagem Conceitual e Ontologias) pela oportunidade de fazer parte e pelos ensinamentos aprendidos que me auxiliaram durante minha jornada na UFES e irão me auxiliar na minha próxima jornada.

A todos que direta ou indiretamente fizeram parte de minha formação e que contribuíram, de alguma forma, para a realização deste trabalho, o meu muito obrigado.

Por fim, agradeço ao meu coorientador, Alexandre, por todo apoio e dedicação ao longo deste trabalho. À minha orientadora Monalessa, agradeço aos conselhos, pelas orientações, pela oportunidade e empenho dedicado a esta pesquisa.

RESUMO

Desde a criação e democratização de computadores pessoais e *smartphones*, novas formas de manusear dispositivos foram surgindo. Com isso, também surgiu um novo ramo de estudos, sendo este conhecido atualmente como Interação Humano-Computador (IHC). Um problema relacionado a IHC conhecido e, muito discutido atualmente, é a baixa usabilidade quando diferentes tipos de usuários utilizam o mesmo sistema. Uma solução possível para este problema são os sistemas adaptativos, pois eles conseguem alterar aspectos de sua estrutura ou funcionalidades e, com isso, acomodar e atender melhor às necessidades de diferentes usuários e suas mudanças ao longo do tempo. Porém, o campo da IHC envolve diferentes áreas de conhecimento, com diferentes especialistas envolvidos, fazendo com que conceitos muitas vezes não possuam um significado consensual, criando, assim, conflitos semânticos no entendimento e na modelagem das características do usuário, entre outros. Diante disto, este trabalho propõe o uso de ontologias no desenvolvimento de sistemas adaptativos. Ontologias têm sido usadas com sucesso em vários domínios para capturar e organizar conhecimento, visando lidar com interoperabilidade e problemas relacionados ao conhecimento. No Núcleo de Estudos em Modelagem Conceitual e Ontologias (NEMO), no qual este trabalho foi realizado, encontra-se em desenvolvimento HCI-ON (*Human-Computer Interaction Ontology Network*), uma rede de ontologias que trata aspectos relacionados a IHC. Nessa rede, no contexto da pesquisa de doutorado do coorientador deste trabalho, estão sendo desenvolvidas ontologias que tratam aspectos relevantes para o desenvolvimento de interfaces adaptativas (e.g., caracterização do usuário). Neste trabalho, um extrato de HCI-ON foi utilizado no desenvolvimento de SNOPI (*Social Network with Ontology-based Adaptive Interface*), uma rede social com interface adaptativa baseada em ontologia. O extrato de HCI-ON foi utilizado na modelagem conceitual de SNOPI e para construção da ontoSNOPI (ontologia operacional de SNOPI). Uma versão inicial de SNOPI foi projetada e desenvolvida, incluindo algumas adaptações de interface de acordo com o perfil do usuário.

Palavras-chave: Interface Adaptativa, Ontologia, Sistema Adaptativo, Rede Social.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS

API	Application Programming Interface
CRUD	Create, Read, Update, Delete
HCI-ON	Human-Computer Interaction Ontology Network
OSDAUI	Ontologies to Support the Development of Adaptive User Interface
HTML	HyperText Markup Language
IHC	Interação Humano-Computador
ISO	International Organization for Standardization
LGPL	GNU Lesser General Public License
MVC	Model-View-Controller
MVVM	Model-View-ViewModel
NEMO	Núcleo de Estudos em Modelagem Conceitual e Ontologias
OWL	Web Ontology Language
W3C	World Wide Web Consortium
OWL-API	Web Ontology Language - Application Programming Interface
SCSS	Sassy Cascading Style Sheets
SNOPI	Social Network with Ontology-based adaPtive Interface
SQL	Standard Query Language
UCO	User Characterization Ontology
UFO	Unified Foundational Ontology
UML	Unified Modeling Language
URL	Uniform Resource Locator

SUMÁRIO

Capítulo 1

Introdução

Este capítulo apresenta uma breve introdução ao tema do trabalho, seus objetivos, histórico do desenvolvimento e a organização deste documento.

1.1 Introdução

Sistemas interativos são sistemas focados na usabilidade para o usuário e seu estudo é realizado principalmente na área de pesquisa de Interação Humano-Computador (IHC). O conceito de usabilidade em sistemas interativos está em constante evolução e geralmente inclui também qualidades como bem-estar, eficácia coletiva, fluxos e outros (CARROLL, 2012). O interesse por sistemas interativos vem aumentando, pois usabilidade é um ponto crucial para o usuário devido à sua ligação direta com efetividade (capacidade de executar a tarefa de forma correta e completa), eficiência (consumo de recursos - e.g., tempo, dinheiro, força de trabalho, memória - para ter eficácia) e satisfação (nível de conforto que o usuário sente ao utilizar a interface) (ISO, 2002).

Sistemas interativos podem ser, também, sistemas adaptativos. De acordo com Benyon (1987), sistemas adaptativos são sistemas que podem alterar aspectos de sua estrutura ou funcionalidades a fim de acomodar as necessidades de diferentes usuários e suas mudanças ao longo do tempo. Com isso, a identificação e classificação das características do usuário se torna necessária. Para apoiar essa classificação são utilizadas informações relacionadas a características físicas, perfis de uso, problemas de saúde, entre outras. Dentre as adaptações que podem ser realizadas em um sistema adaptativo estão as adaptações na interface do usuário. Este trabalho está interessado particularmente em sistemas interativos adaptativos, cujas adaptações são realizadas automaticamente na interface do usuário, ou seja, cuja interface do usuário é uma interface adaptativa.

As adaptações de interface do usuário, como também outras adaptações do sistema, podem ser feitas com base em conhecimento estruturado em ontologias. Uma ontologia é uma especificação formal e explícita de uma conceituação compartilhada (GRUBER, 1995). Ontologias têm sido usadas com sucesso em vários domínios para capturar e organizar

conhecimento, visando lidar com interoperabilidade e problemas relacionados ao conhecimento (COSTA, 2021). No contexto de IHC, ontologias têm sido aplicadas principalmente para representação de conhecimento, apoio ao design e avaliação de IHC, adaptação de interfaces e anotação semântica, entre outros (COSTA *et al.*, 2020).

Levando-se em consideração os benefícios providos por um sistema com interface adaptativa, como, por exemplo, uma melhor interação com o usuário de acordo com suas características, e o uso bem sucedido de ontologias no domínio de IHC (COSTA *et al.*, 2020) e também no domínio de Engenharia de Software (RUY *et al.*, 2016), que trata do desenvolvimento de sistemas de software em geral (entre eles, sistemas interativos e adaptativos), encontra-se em desenvolvimento no NEMO (Núcleo de Estudos em Modelagem Conceitual e Ontologias), grupo de pesquisa no qual este trabalho está sendo realizado, uma pesquisa de Doutorado (cujo autor é o coorientador deste trabalho) que investiga o uso de ontologias no desenvolvimento de interfaces adaptativas. No contexto dessa pesquisa, foi planejada a realização de um estudo exploratório sobre como ontologias podem ser usadas para auxiliar no desenvolvimento de um sistema de interface adaptativa. O sistema em questão foi desenvolvido no contexto deste trabalho. Este trabalho também está relacionado a outra pesquisa de Doutorado (COSTA, 2021), que propõe uma rede de ontologias para o domínio de IHC (HCI-ON - *HCI Ontology Network*), que será usada em soluções de problemas relacionados à semântica e/ou representação de conhecimento. As ontologias usadas neste trabalho para desenvolver o sistema proposto são ontologias da rede de ontologias proposta por COSTA *et al.* (2021) e que têm sido desenvolvidas no contexto da pesquisa de doutorado do coorientador deste trabalho. Dessa forma, o desenvolvimento do sistema proposto neste trabalho é relevante, pois, além dos benefícios providos pelo sistema em si, será possível colaborar com pesquisas de Doutorado em andamento no NEMO, visando solucionar problemas semânticos e de representação de conhecimento no desenvolvimento de sistemas interativos, bem como contribuir para melhorar a usabilidade desses sistemas.

1.2 Objetivos

Este trabalho tem como objetivo geral desenvolver um sistema de interface adaptativa baseada em ontologias de *HCI-ON*. O sistema deve se adaptar automaticamente para o

usuário, apoiando suas necessidades de uso de acordo com suas características. Em síntese, o sistema deve alterar a sua interface para melhorar sua usabilidade de acordo com informações acerca do usuário.

Esse objetivo geral pode ser detalhado nos seguintes objetivos específicos:

I. Investigar na literatura sobre o uso de ontologias no desenvolvimento de interfaces adaptativas;

II. Especificar o sistema a ser desenvolvido, estabelecendo seus requisitos, bem como caracterizando quais aspectos e funcionalidades serão adaptados pelo sistema para apoiar o usuário;

III. Projetar a solução a ser desenvolvida;

IV. Implementar e testar o sistema com interface adaptativa.

1.3 Histórico de Desenvolvimento do Trabalho

Este trabalho foi conduzido de acordo com as seguintes atividades:

I. Revisão da Literatura: O trabalho foi iniciado com uma revisão bibliográfica sobre interfaces adaptativas em IHC, ontologias e redes ontologias. Foram lidos materiais (livros, teses, dissertações e artigos científicos) no contexto do assunto. Em paralelo, está sendo realizado um mapeamento sistemático da literatura[1] sobre o desenvolvimento de interfaces adaptativas em IHC utilizando ontologias, em conjunto com Alexandre Adler Cunha de Freitas e Simone Dornelas Costa, alunos autores das pesquisas de Doutorado com as quais este trabalho está relacionado.

II. Desenvolvimento de Ontologias: Consistiu no desenvolvimento das ontologias necessárias para o desenvolvimento do sistema. Esta etapa foi realizada em conjunto com os alunos de Doutorado Alexandre Adler Cunha de Freitas e Simone Dornelas Costa .

[1] Um mapeamento sistemático da literatura é um estudo secundário estudo destinado a dar uma visão geral de uma área de pesquisa através da classificação e contagem de contribuições em relação às categorias dessa classificação. Faz um amplo estudo da literatura sobre um tema específico e tem como objetivo identificar evidências sobre esse tópico (PETERSEN *et al.*, 2015).

III. **Estudo de Tecnologias**: Nesta etapa ocorreu o estudo das tecnologias necessárias para o desenvolvimento do sistema e a seleção das tecnologias mais adequadas para o propósito deste trabalho. Foram consideradas tecnologias para o desenvolvimento *front-end* , *back-end* e da ontologia operacional.

IV. **Levantamento e Análise de Requisitos:** Consistiu na definição do domínio de aplicação e propósito do sistema, identificação e análise dos requisitos do sistema, incluindo a elaboração de diagramas da UML (*Unified Modeling Language)*, tais como, diagrama de classes. Após isso, utilizando-se o método prototipação, foram desenvolvidos *mockups* do sistema.

V. **Design, Implementação e Testes:** Nesta etapa, foi definido o projeto de arquitetura da ferramenta e o projeto de seus componentes. O sistema foi implementado e foram realizados testes para a verificar a consistência do sistema.

VI. **Escrita da Monografia:** Consistiu na escrita desta monografia.

1.4 Organização do Texto

Este primeiro capítulo contém a introdução do trabalho, assim como, seus objetivos e suas etapas de desenvolvimento. Além deste capítulo, a monografia possui outros três capítulos, sendo eles:

1. **Capítulo 2 – Fundamentação Teórica:** Apresenta uma breve fundamentação teórica sobre IHC, interfaces adaptativas e ontologias. Também apresenta as tecnologias que foram utilizadas no desenvolvimento do sistema.
2. **Capítulo 3 – SNOPI - Requisitos, Modelagem Conceitual e ontoSNOPI:** Apresenta os requisitos do sistema, a descrição de seu contexto e propósito de uso, seus casos de uso, modelos de classe e a ontologia operacional desenvolvida.
3. **Capítulo 4 – SNOPI: Projeto de Sistema e Implementação:** Apresenta a arquitetura sistema, detalhes de componentes da arquitetura. Também descreve as funcionalidades implementadas do sistema, apresenta algumas delas e descreve o fluxo de utilização.

4. **Capítulo 5 – Considerações finais:** Apresenta as considerações finais do trabalho, incluindo algumas dificuldades encontradas, limitações do sistema desenvolvido e experiências adquiridas. Além disso, são sugeridos alguns trabalhos futuros.

Capítulo 2

Fundamentação Teórica

Neste capítulo são apresentados os principais fundamentos teóricos relacionados a este trabalho. Na Seção 2.1 são tratados aspectos sobre Interação Humano-Computador. A Seção 2.2 trata de Interfaces Adaptativas. Na Seção 2.3 são apresentados os tópicos mais relevantes sobre ontologias, redes de ontologias, HCI-ON e seu fragmento relevante para este trabalho. Na Seção 2.4 são apresentadas as tecnologias utilizadas neste trabalho. Por fim, a Seção 2.5 apresenta as considerações finais do capítulo.

2.1 Interação Humano-Computador

Até o final dos anos 1970, os únicos usuários que interagiam com computadores eram profissionais da área de tecnologia ou entusiastas da computação. Isso mudou de forma disruptiva com o surgimento do computador pessoal no final dos anos 1970 (CAROLL, 2012). Essa nova forma de usar o computador exigiu um novo ramo de estudos e foi assim que nasceu o conceito *interação humano-computador* (IHC). A IHC é uma área de pesquisa e prática, que surgiu inicialmente como uma área de especialidade da Ciência da Computação, abrangendo as ciências cognitivas e a engenharia de fatores humanos. Mas, o conceito IHC tem se expandido de forma rápida e constante por mais de três décadas, atraindo profissionais de muitas outras disciplinas e incorporando diversos conceitos e abordagens (CAROLL, 2012).

Sobre a conceituação referente ao domínio de IHC, pesquisadores pontuam fatores diferentes nas definições técnica ou científica, o que leva à existência de vários termos e conceitos e, consequentemente, à falta de uma conceituação comum na comunidade. Além disso, à medida que a área de IHC continua a evoluir, novos termos são propostos e novos significados são atribuídos aos termos existentes (COSTA, 2021).

Um conceito ligado diretamente a IHC é o de sistemas interativos, que, de acordo com a ISO 9241-210 (2011), é a combinação de hardware, software e/ou serviços que recebe entrada e comunica a saída para os usuários. Com a evolução dos computadores, rapidamente se reconheceu que os sistemas interativos eram a chave para progredir além das conquistas iniciais (CAROLL, 2012). Mas, existe um fator entre a comunicação do usuário e o computador, que deve ser considerado criteriosamente, caso contrário, pode interferir na interação com o sistema. Este fator nada mais é que o próprio usuário, que possui

características e necessidades próprias, que podem requerer funcionalidades ou interfaces diferentes para utilizar o mesmo sistema. Para tratar essa questão e minimizar os problemas causados por sistemas que não respondem adequadamente às necessidades de diferentes usuários, surgem as interfaces adaptáveis e adaptativas.

2.2 Interfaces Adaptativas

O sistema com interface adaptativa é um tipo particular de sistema adaptativo (BENYON, 1988). Um sistema é adaptativo se o mesmo for capaz de mudar características e funcionalidades automaticamente, de acordo com as necessidades dos usuários (OPPERMANN, 1994). É possível focar a adaptação do sistema visando especificamente à interface de usuário, resultando em uma interface adaptativa.

Uma interface de usuário adaptativa é um artefato de software que melhora sua capacidade de interagir com um usuário, construindo um modelo do usuário com base na experiência parcial do mesmo (LANGLEY, 1999). E, segundo Schneider-Hufschmidt e Malinowski (1993), uma interface de usuário adaptativa é uma interface que se adapta (*i.e.*, muda seu *layout* ou elementos) às necessidades do usuário ou contexto e pode ser alterada de forma semelhante para cada usuário.

O termo interface adaptável, que também se refere a interfaces que sofrem adaptações, pode ser confundido com interface adaptativa. No entanto, são conceitos distintos. De acordo com Jameson (2008), uma interface é *adaptável* quando um usuário pode explícita e deliberadamente adaptar a interface para as suas necessidades. Por outro lado, uma interface é *adaptativa* quando ela se adequa sozinha às necessidades do usuário. A semelhança entre os termos está no fato de que ambos tratam adaptações de interface. A principal diferença está na maneira como são implementadas as adaptações. Exemplificando, uma interface adaptável provê uma opção para alteração na interface pelo usuário, já uma interface adaptativa proporciona modificações automaticamente de acordo com as características do usuário. Este trabalho tem como foco as interfaces adaptativas.

A necessidade por interfaces adaptativas surge quando sistemas interativos se tornam cada vez mais abrangentes e eficazes, permitindo que diferentes tipos de usuários possam ter acesso a partir de algum dispositivo de computação, sendo ele um computador, *smartphone*,

tablet, *notebook*, entre outros. Nesse ponto, sistemas foram desenvolvidos levando em consideração a grande maioria dos usuários, para melhor usabilidade da interface. No entanto, este modo de desenvolvimento acarreta em dois problemas: o primeiro é a exclusão de alguns tipos de usuários e o segundo é que as características de um usuário se modificam com o passar do tempo. Com isso, para atender a todos os usuários, poderia-se adaptar manualmente a interface, desenvolver outra interface para certos tipos de usuário ou criar uma interface adaptativa. Desenvolver uma interface adaptativa é uma boa estratégia para diminuir o impacto desses problemas (OPPERMANN, 1994), pois, ao considerar as diferentes características dos diferentes usuários, a própria interface irá se moldar para atender as necessidades do usuário, tornando o sistema mais acessível e melhorando sua usabilidade.

Contudo, desenvolver uma interface adaptativa possui alguns desafios (OPPERMANN, 1994), sejam eles na coleta de informações do usuário, na mensuração de características do usuário ou na organização do conhecimento mediante as informações coletadas. Além disso, há questões relacionadas à proteção e privacidade dos dados do usuário, à adaptabilidade do usuário a uma nova interface e à dificuldade em adaptar o modelo conceitual da interface para conseguir atingir os objetivos desejados.

2.3 Ontologias

Uma ontologia é uma especificação formal e explícita de uma conceituação compartilhada (STUDER, 1998). Ontologias podem ser organizadas em três camadas arquiteturais (SCHERP, 2011) sendo elas: (i) *ontologias de fundamentação*, que possuem um escopo mais geral. São reutilizáveis em diferentes cenários de modelagem e visam modelar conceitos básicos, gerais e também relações que compõem o nosso mundo (como objetos e eventos); (ii) *ontologias de núcleo*, que são um refinamento de ontologias de fundamentação e, com isso, fornecem uma definição mais detalhada, porém abstrata, do conhecimento estruturado em um cenário de modelagem (por exemplo, serviço, organização). Mas, ainda assim, são genéricas e abrangem um conjunto de (sub)domínios; e por último, (iii) *ontologias de domínio,* onde encontra-se a representação do conhecimento que é específico para um determinado domínio (por exemplo, serviço de software, universidade).

Uma outra importante distinção é entre ontologias como modelos conceituais, chamadas de ontologias de referência e ontologias como artefatos computacionais, chamadas de ontologias operacionais (GUIZZARDI, 2007). Uma *ontologia de referência* é construída com o único objetivo, de fazer a melhor descrição possível do domínio na realidade para um certo nível de granularidade e ponto de vista. É uma ontologia que independe de uma solução específica, o que é diferente de uma *ontologia operacional*, que tem como foco garantir a realização de funções computacionais.

Para domínios grandes e complexos, a organização do conhecimento como uma única ontologia torna a manipulação (e.g., criação, uso, reuso e manutenção) algo custoso. Mas, levar ao outro extremo e representar cada subdomínio isoladamente também é uma tarefa muito custosa, não só pelo seu tamanho, mas também pelos inúmeros problemas relacionados à integração e fusão de ontologias (RUY *et al.,* 2016). Nesse contexto, ontologias podem ser organizadas em redes de ontologias. Uma *rede de ontologias* é uma coleção de ontologias relacionadas entre si por meio de relações como dependência e alinhamento.

O domínio de IHC é muito amplo e, com isso, muitas ontologias têm sido criadas e usadas de maneira isolada, mesmo quando possuem conceitos compartilhados entre elas (COSTA, 2021). Isso dificulta obter uma conceituação abrangente e consistente acerca do domínio, bem como o reúso dessa conceituação como um todo ou de extratos dela. A partir dessa problemática, COSTA *et al.* (2020) propuseram a *Human-Computer Interaction Ontology Network* (HCI-ON), uma rede de ontologias que contém ontologias que tratam aspectos relacionados a IHC. Em síntese, HCI-ON tem como objetivo: (i) estabelecer uma estrutura baseada em camadas para suportar ontologias que abordam diferentes aspectos de IHC; (ii) fornecer um suporte eficaz para integrar e alinhar ontologias de domínio, promovendo, assim, o crescimento do conhecimento representado na rede; e (iii) apoiar soluções para resolver problemas relacionados a conhecimento e de interoperabilidade semântica em IHC (COSTA, 2021).

Neste trabalho, será utilizado um extrato de HCI-ON contendo conceitos de algumas ontologias que ainda encontram-se em desenvolvimento[2]: *User Characterization Ontology* (UCO), que envolve conceitos relacionados à caracterização de usuário, *UI Types and Elements Ontology*

[2] Embora o autor deste trabalho esteja participando ativamente do desenvolvimento das ontologias apresentadas, optou-se por apresentá-las no capítulo de fundamentação teórica, pois as referidas ontologias são contribuição da pesquisa de doutorado do coorientador deste trabalho.

(UIT&EO), que aborda componentes de interface, *Adaptive Interface Ontology (AIO)*, que trata das de adaptações em componentes de interface, *Device Context Ontology (DCO)*, que aborda aspectos do contexto em que se encontra o dispositivo no qual o sistema adaptativo é executado, *User Profile Ontology (UPO)*, que aborda aspectos, de preferências do usuário. A Figura 1 apresenta uma visão geral (parcial) de HCI-ON, destacando-se com linhas roxas as ontologias diretamente utilizadas neste trabalho. As Ontologias AIO, DCO, UPO e UCO não estão presentes na Figura 1, pois ainda estão em desenvolvimento e seu desenvolvimento não havia sido iniciado quando a figura foi criada por Costa (2021).

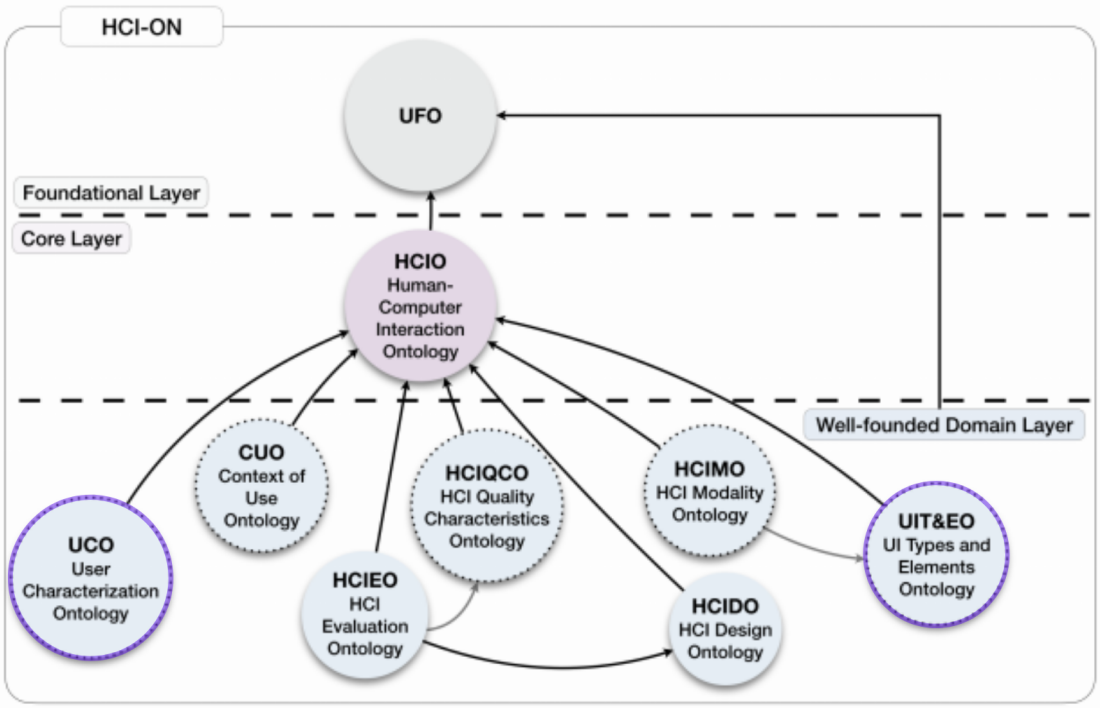

Figura 1 - Visão geral (parcial) de HCI-ON. Fonte: Adaptado de COSTA (2021)

Todas as ontologias de HCI-ON são fundamentadas em UFO (*Unified Foundational Ontology*)[3] (GUIZZARDI, 2005), que fica na *Foundational Layer* de HCI-ON. Na *Core Layer*, está a *Human-Computer Interaction Ontology* (HCIO) (Costa et al., 2022), que inclui conceitos centrais de IHC, como *User*, *Interactive Computer System* e *Human-Computer Interaction,* que são reutilizados pelas diversas ontologias de domínio localizadas na *Well-founded Domain Layer*. Dada a forte

[3] UFO foi desenvolvida reunindo continuamente teorias dos campos da ontologia formal, ciência cognitiva, linguística e lógica filosófica na filosofia. Inclui uma série de pequenas teorias que cobrem os conceitos básicos de modelagem conceitual, incluindo tipos de entidade e tipos de relação (GUIZZARDI et al., 2021).

relação entre Engenharia de Software e IHC, embora não mostrado na Figura 1, HCI-ON reutiliza alguns conceitos de ontologias da *Software Engineering Ontology Network* (SEON) (RUY *et al.*, 2016), tais como *Hardware Equipment* e *Computer System,* conceitos presentes na *System and Software Ontology (*SysSwO) de SEON.

As Figuras 2 e 3 apresentam fragmentos de HCI-ON relevantes para este trabalho. Nas figuras,linhas pretas tracejadas separam conceitos de diferentes camadas e ontologias (os nomes de cada ontologia e respectiva camada são indicados à direita). Diferentes cores são utilizadas para indicar conceitos de diferentes ontologias: conceitos de UFO aparecem em cinza, conceitos de SysSwO em verde, conceitos de HCIO em amarelo, conceitos de UPO em rosa claro, conceitos de AIO em roxo, conceitos de UCO em verde escuro, conceitos de DCO em rosa escuro e conceitos de UIT&EO[4] em azul. Os conceitos com bordas destacadas em azul foram diretamente utilizados no desenvolvimento do sistema proposto neste trabalho, tendo sido utilizados para a implementação de uma ontologia operacional e na modelagem conceitual do sistema.

[4] UIT&EO foi utilizada durante o desenvolvimento para auxiliar a mapear os componentes que sofrem com as modificações de adaptações da interface, não sendo utilizados diretamente para a modelagem conceitual e nem para a construção da ontologia operacional.

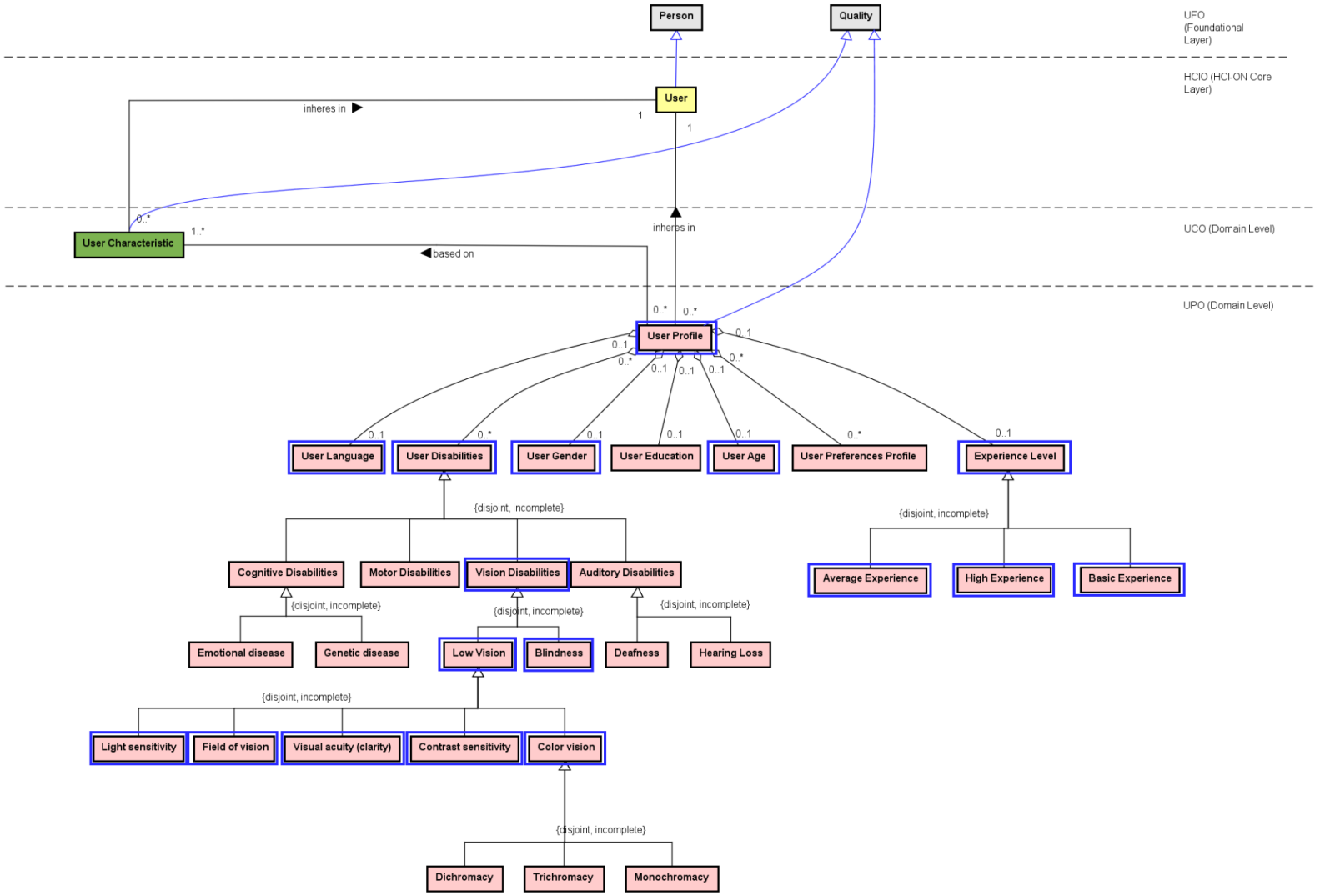

Figura 2 - Fragmento de HCI-ON incluindo conceitos sobre perfil de usuário relevantes a este trabalho. Fonte: Adaptado de FREITAS (2022, em desenvolvimento)

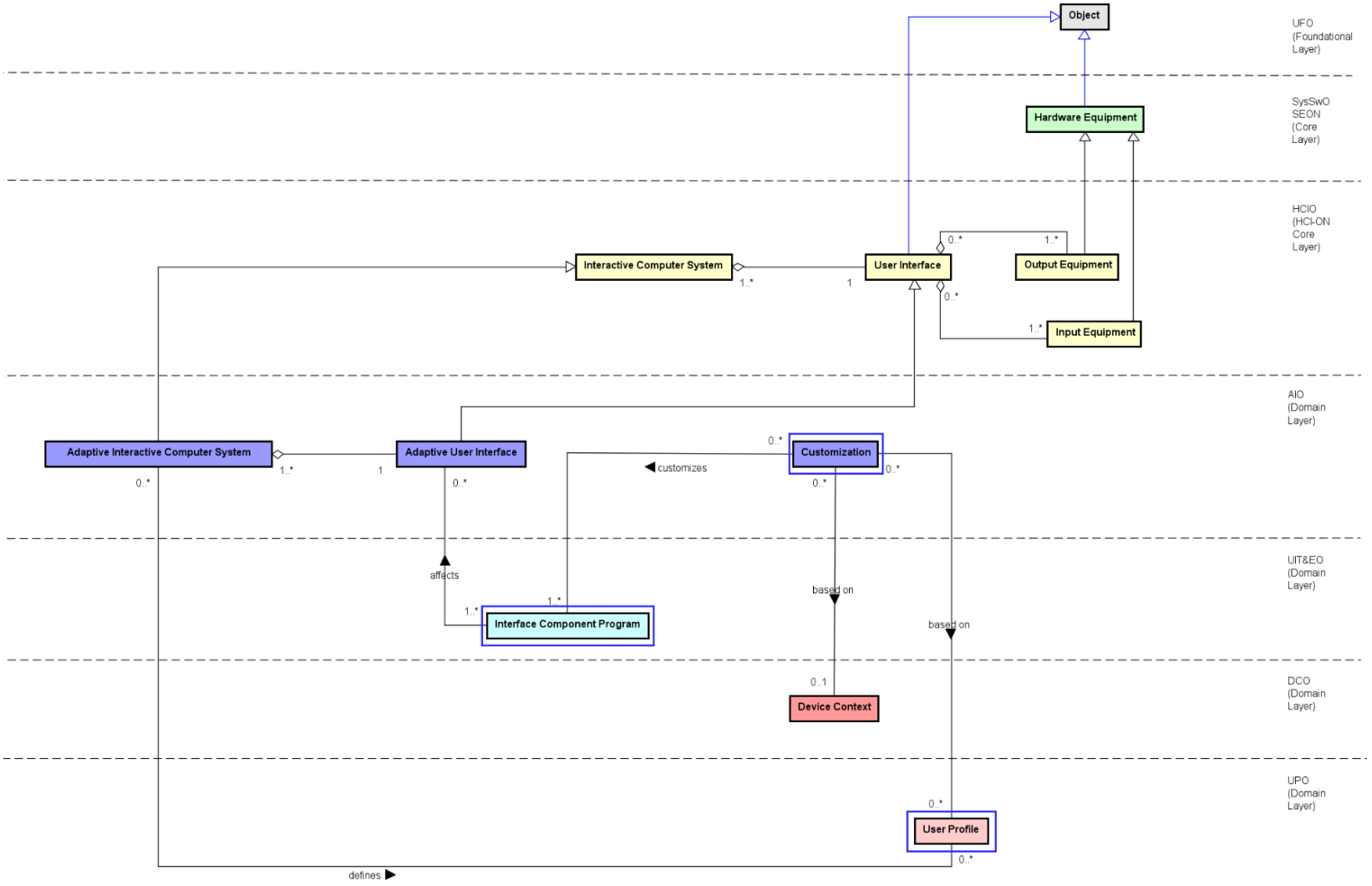

Figura 3 - Fragmento de HCI-ON incluindo conceitos relevantes relacionados a adaptação de interfaces. Fonte: Adaptado de FREITAS (2022, em desenvolvimento)

2.4 Tecnologias Utilizadas neste Trabalho

Nesta seção são apresentadas descrições das tecnologias utilizadas utilizadas neste trabalho. O sistema foi desenvolvida como uma aplicação baseada na Web, mais conhecido como *WebApp* (PRESSMAN; LOWE, 2009), se baseando no modelo cliente-servidor, onde chama-se de *front-end* a parte de interesse do cliente e *back-end* a parte de interesse do servidor. Em muitas aplicações *WebApps* é comum a prática do uso de *frameworks* para auxílio do desenvolvimento. Em Engenharia de Software , um *framework* é uma estrutura que fornece códigos reutilizáveis de forma abstrata, que podem auxiliar nos mais diversos projetos (RIEHLE, 2000). O objetivo de utilizar um *framework* é diminuir a complexidade do desenvolvimento de uma aplicação *Web* retirando o custo de tarefas comuns a vários sistemas (por exemplo: controle de autenticação). Neste trabalho, foram utilizados *frameworks* em ambos os lados da aplicação, tanto na parte do cliente quanto na parte do servidor. Os *frameworks* utilizados são apresentados nas seções a seguir. Este trabalho também conta com o uso da ferramenta *JHipster*[5], que é uma plataforma para desenvolvimento de aplicações *Web*, onde encapsula o uso de ferramentas, criando um ambiente com menor grau de complexidade para desenvolvimento de uma aplicação (por exemplo: realizando a pré-configuração para o uso das ferramentas utilizadas escolhidos, identificando e aplicando os modelos dos *frameworks* selecionados). O sistema desenvolvido também conta com o uso de ferramentas para a manipulação de ontologias e a realização de *reasoning,* que serão apresentados nas seções seguintes.

2.4.1 *Back-end: Framework Spring*

O *back-end* do sistema foi desenvolvido baseado no padrão de arquitetura *Spring Boot*, que é um módulo do *Framework Spring*. O *Spring Boot*[6] também se baseia no modelo *Model-View-Controller* (GAMMA, HELM, JOHNSON, VLISSIDES, 1994), comumente conhecido apenas por MVC. O padrão MVC é constituído por três camadas, sendo elas: a camada de Modelo (*Model*), responsável pela lógica de negócio da aplicação; a camada de Visão

[5] *https://www.jhipster.tech*

[6] *https://docs.spring.io/spring-boot/docs/current/reference/html/*

(*View*), que é a camada de interação com o usuário; e a camada de Controle (*Controller*), que faz a comunicação entre as camadas citadas anteriores.

Entretanto, a arquitetura *Spring Boot* possui uma estrutura um pouco mais complexa que o MVC e é constituído por quatro camadas, sendo que três são correspondentes ao MVC, e contendo uma quarta camada voltada para lógica de negócios, que lida com a lógica de negócios dos objetos para depois haver a persistência no banco de dados.

O *framework open source Spring* utiliza como linguagem de programação JAVA[7], sendo popular em aplicações voltadas para *Web*. A escolha deste *framework* foi devido ao mesmo possuir uma documentação vasta e de fácil compreensão, diminuindo, assim, a curva de aprendizado para a sua utilização. Além disso, a aplicação do padrão de arquitetura *Spring Boot*, baseada no MVC, atua de forma clara no *framework* auxiliando no desenvolvimento do código.

A implementação do *back-end* com o *Spring* neste trabalho utilizou as seguintes linguagens:

- **JAVA[8]:** Linguagem de programação orientada a objetos de código aberto, sendo uma das linguagens mais utilizadas para desenvolvimento de aplicações *Web* e *Mobile*.
- **PostgreSQL[9]:** Linguagem de gerenciamento de banco de dados objeto relacional, que utiliza e estende a linguagem de consulta estruturada, o SQL.

No Capítulo 3 será detalhada a implementação do padrão utilizado a partir da utilização do *Spring Boot*.

2.4.2 *Front-end: Framework Angular*

O *front-end* da aplicação foi desenvolvido a partir do padrão MVVM, também conhecido por *Model-View-ViewModel* (SMITH, 2009). Este padrão foi baseado no padrão MVC e tem como principal diferença a camada *ViewModel*, pois utiliza o recurso de *data binding*.

---

[7] *https://www.java.com/*

[8] *https://www.oracle.com/br/java/*

[9] *https://www.postgresql.org/*

Este recurso, compreende em atualizar automaticamente as informações da camada de *View* quando houver alterações na camada *Model*.

O *framework Angular*[10] orienta-se pelos padrões tanto do MVC quanto do MVVM e foi utilizado para o desenvolvimento deste trabalho. O *Angular* foi desenvolvido e estruturado com elementos que o tornam componentizado, o que contribui para diminuição do custo de desenvolvimento inicial e torna o desenvolvimento incremental. Também neste aspecto da componentização, o *framework* auxilia na utilização e manutenção do código implementado, mesmo com o aumento da complexidade da aplicação. Um ponto importante para a escolha do *framework* é a sua documentação completa, muito didática. Além disso, o *framework* possui uma comunidade muito ativa. Devido ao forte acolhimento da comunidade pelo *framework*, os desenvolvedores estão sempre atualizando e implementando melhorias para auxiliar no desenvolvimento de aplicações, o que faz com que o *Angular* possua muitas versões estáveis. Neste trabalho utilizamos a versão estável e com compatibilidade com a ferramenta *JHipster*, que no momento do desenvolvimento do sistema SNOPI era a versão 10.0.0 do Angular.

A implementação do *front-end* com o *Angular* neste trabalho utilizou as seguintes linguagens:

- ***HyperText Markup Language* (HTML)**: Linguagem de marcação interpretada utilizada para a construção e organização de elementos da página *Web*.
- ***TypeScript***: Linguagem que estende *JavaScript* e tem como características ser interpretada e estruturada, fortemente tipada, multiparadigmas e assíncrona. Utilizada no *framework* para comunicar componentes e aplicar *scripts*.
- ***Sassy Cascading Style Sheets* (SCSS[11])**: Linguagem que estende a sintaxe CSS (*Cascading Style Sheets* - linguagem de estilização), usada para aplicar estilos, com recursos programáveis, nos componentes do HTML, a fim de torná-los mais agradáveis e modernos para os usuários.

[10] *https://angular.io/*

[11] *https://sass-lang.com/*

No Capítulo 3 será detalhada a implementação do padrão MVVM a partir da utilização do *Angular*.

2.4.3 Ontologia: Manipulação e *Reasoning*

Para realizar a manipulação da ontologia foram utilizadas ferramentas específicas, visando diminuir a complexidade envolvida no processo. Dentre as ferramentas, o *Protégé*[12] foi utilizado para construção da ontologia operacional. O *Protégé* é uma ferramenta de código aberto para manipulação de ontologias, desenvolvido e mantido pela Universidade de *Stanford*.

Utilizando o *Protégé*, após a construção da ontologia operacional ocorre a exportação do arquivo OWL (*Web Ontology Language*) e o mesmo pode ser consumido pelo *back-end* do sistema. O OWL, de acordo com a W3C[13], consiste em uma linguagem da *Web* semântica projetada para representar conhecimento rico e complexo sobre coisas, grupos de coisas e relações entre coisas. Neste trabalho, utilizamos o OWL 2, que representa uma extensão do OWL. A partir deste ponto, utilizamos duas API para inserir instâncias na ontologia e realizar inferências (*reasoning*) sobre essas instâncias, sendo essas APIs, o OWL-API e o *HermiT Reasoner*, respectivamente.

O OWL-API[14] tem como objetivo ser uma API voltada para diminuir a complexidade do desenvolvimento com a linguagem OWL no ambiente JAVA. Essa API possui foco em OWL 2, possui código aberto e está disponível sob as licenças LGPL e Apache.

Para realizar inferências dentro do próprio ambiente de desenvolvimento foi utilizada a API *HermiT Reasoner*[15]. O *HermiT Reasoner* é um raciocinador para ontologias escritas usando a OWL. Dado um arquivo OWL, o *HermiT* pode determinar se a ontologia está consistente ou não, consegue identificar relações entre classes e entre outras funcionalidades. Também vale pontuar que a API está em conformidade com a OWL 2 e é implementada com base na

[12] *https://protege.stanford.edu/*

[13] *https://www.w3.org/*

[14] *https://github.com/owlcs/owlapi*

[15] *http://www.hermit-reasoner.com/*

linguagem de OWL DL[16], que é uma sublinguagem expressiva utilizada na implementação de OWL.

No Capítulo 3 será detalhada a manipulação da ontologia a partir da utilização das tecnologias mencionadas.

2.5 Considerações Finais do Capítulo

Este capítulo apresentou fundamentos teóricos relevantes para o desenvolvimento deste trabalho. Foram abordados tópicos relacionados a IHC, interfaces adaptativas, ontologias e foi apresentado o extrato de HCI-ON relevante para o trabalho. Também foram apresentadas as tecnologias e o porquê de elas terem sido utilizadas neste trabalho.

[16] *https://www.w3.org/TR/owl-guide/*

Capítulo 3

SNOPI - Requisitos, Modelagem Conceitual e ontoSNOPI

Este capítulo apresenta os principais resultados produzidos ao longo do desenvolvimento do sistema proposto. Na Seção 3.1 são apresentados os resultados produzidos durante a especificação e análise de requisitos do sistema. A Seção 3.2 apresenta informações relacionadas à ontologia utilizada. Por fim, na Seção 3.3, são apresentadas as considerações finais do capítulo.

3.1 Requisitos e Modelagem Conceitual

O levantamento de requisitos envolve buscar, junto aos usuários, clientes e outros interessados, seus sistemas e documentos, todas as informações possíveis sobre as funções que o sistema deve executar (requisitos funcionais) e as restrições sob as quais ele deve operar (requisitos não funcionais) (FALBO, 2017). Nesta seção, são apresentados os requisitos identificados, os casos de usos e o diagrama de classes do sistema.

3.1.1 Visão Geral e Propósito do Sistema

O mundo cada vez mais se torna digital e com isso novas formas de se relacionar com dispositivos computacionais devem ser pensadas para melhorar a interação dos usuários. Conforme discutido no Capítulo 1, neste trabalho é proposto um sistema com interface adaptativa baseada em ontologias. O desenvolvimento do sistema serviu como um estudo exploratório sobre o uso de ontologias no desenvolvimento de sistemas de interfaces adaptativas, para ajudar a minimizar problemas causados por sistemas que não respondem adequadamente às necessidades de diferentes usuários.

O sistema desenvolvido é uma rede social voltada para a área acadêmica e tem como objetivo prover ao usuário funcionalidades para ele compartilhar postagens, seguir outros usuários para acompanhar suas publicações e também organizar/publicar *links* úteis (por exemplo: *links* de trabalhos, artigos publicados, conferências, entre outros). A decisão pelo desenvolvimento deste sistema se deu ao se notar que, com a pandemia, o acesso a eventos científicos (congressos, conferências, workshops, simpósios, etc.) foi facilitado, uma vez que os

eventos passaram a ser realizados na modalidade online e com inscrições com valores reduzidos ou até mesmo gratuitas, permitindo que mais pessoas possam participar. Com isso, notou-se que informações sobre eventos científicos passíveis de serem assistidas passaram a ser divulgadas com maior frequência entre os membros do grupo de pesquisa no qual este trabalho foi conduzido, havendo, inclusive, um incentivo por parte dos pesquisadores seniores para que os demais membros do grupo participassem de tais eventos. Essas informações ficavam espalhadas em e-mails, mensagens no WhatsApp e em outros canais de comunicação e eram facilmente esquecidas. Além disso, geralmente era requerido algum esforço para serem recuperadas (e.g., a pessoa teria que lembrar em que canal recebeu a informação para recuperá-la ou teria que procurá-la em diversos canais até encontrá-la). Percebeu-se, então, que seria interessante ter um único canal no qual poderiam ser centralizadas informações sobre os eventos e, também, outras informações de interesse acadêmico, tais como artigos e outras publicações. Levando-se em consideração o crescente uso de redes sociais pela sociedade atual e visando estimular a interação entre os diferentes usuários, optou-se por desenvolver o sistema no formato de uma rede social.

O sistema conta com uma interface adaptativa baseada em ontologias, a qual apresenta a possibilidade de adaptações, sendo elas: modo escuro, modo de alto contraste, modo de experiência baixa/média/alta, modo de modificação de tamanho de fontes, cores adaptadas para tratamento de daltonismo e sistema responsivo para a maioria dos tamanhos de telas. Considerando que o desenvolvimento deste trabalho foi realizado em um intervalo de tempo esperado para um trabalho de conclusão de curso de graduação, não seria possível implementar muitas variações de adaptações de interface. Sendo assim, optou-se por um conjunto limitado, mas que seja capaz de demonstrar minimamente os resultados que podem ser obtidos com o uso de ontologias para apoiar a implementação de uma interface adaptativa. Dessa forma, este trabalho fornece resultados (o sistema em si e o conhecimento adquirido e lições aprendidas ao longo do processo de desenvolvimento) que o permitem cumprir o seu propósito de servir como um estudo exploratório sobre o uso de ontologias no desenvolvimento de interfaces adaptativas. Vale destacar que nesta monografia o foco é apresentar o sistema desenvolvido e

colaborar com o estudo exploratório, que será tratado no contexto da pesquisa de doutorado do coorientador deste trabalho.

3.1.2 Requisitos

Visando atender ao propósito do sistema descrito anteriormente, foram definidos os requisitos apresentados na Tabela 1.

Tabela 1 - Requisitos funcionais identificados para SNOPI

ID	Requisito	Dependência
RF01	O sistema deve permitir ao usuário postar e deletar mensagens.	RF07
RF02	O sistema deve permitir ao usuário seguir outros usuários.	RF01
RF03	O sistema deve permitir armazenar informações sobre as características do usuário (considerando as informações relacionadas ao conceito *Profile* apresentado na ontologia).	
RF04	O sistema deve permitir ao usuário curtir mensagens (postagens).	RF01, RF02
RF05	O sistema deve permitir ao usuário comentar mensagens.	RF01, RF02
RF06	O sistema deve permitir ao usuário compartilhar mensagens.	RF01, RF02
RF07	O sistema deve permitir o cadastro de usuários.	
RF08	O sistema deve permitir ao usuário modificar suas preferências de interface e alterá-las quando desejar.	RF03
RF09	O sistema deve permitir ao usuário buscar por mensagens e por outros usuários dentro do sistema.	RF01, RF07
RF10	O sistema deve fornecer um meio para gerenciar outros usuários do sistema.	RF07

Também foram definidos três requisitos não-funcionais:

- Responsividade: o sistema deve se adaptar a dispositivos com diferentes tamanhos de tela (*mobile* e *desktop*).
- Adaptabilidade: a interface do sistema deve se adaptar de acordo com características e preferências do usuário.
- Portabilidade: o sistema deve rodar nos principais navegadores (Chrome, Opera, Firefox, Microsoft Edge).

- Controle de Acesso: o sistema deve controlar o acesso dos usuários considerando-se dois perfis distintos (usuário e administrador).

3.1.3 Casos de Uso

Os casos de uso tem como propósito captar e definir as funcionalidades que um sistema deve prover para seus usuários (atores) (FALBO, 2017). Como o sistema consiste em um protótipo de uma rede social para aplicação da interface adaptativa, temos como principais atores:

- **Usuário:** ator responsável por utilizar as funcionalidades do sistema de modo livre..
- **Administrador:** ator responsável por gerir os usuários do sistema e o sistema de modo geral.

A Figura 4 apresenta os casos de usos do sistema. Para cada caso de uso são indicados os requisitos relacionados.

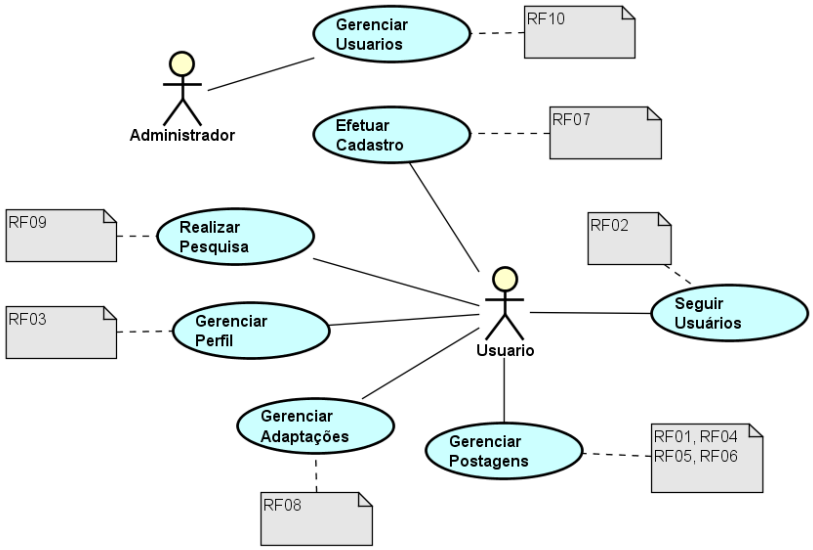

Figura 4 - Diagrama de casos de uso de SNOPI

O caso de uso **Gerenciar Postagens** é realizado pelo **Usuário** do sistema e tem como função gerir as postagens (por exemplo, compartilhar postagens, criar novas postagens, deletar postagens de autoria do usuário, curtir e comentar outras postagens). O caso de uso **Efetuar Cadastro** é realizado pelo ator usuário e tem como função possibilitar ao ator criar contas gratuitas para utilização do sistema e lida diretamente com o sistema de autorização de

cada usuário, mantendo a integridade do sistema. O caso de uso **Realizar Pesquisa** promove a conexão de um usuário com outros, permitindo que ele realize buscas por postagens e outros usuários. O Caso de uso **Seguir Usuários** realiza o mapeamento da função que permite um usuário seguir outros usuários presentes no sistema.

Os casos de usos **Gerenciar Perfil** e **Gerenciar Adaptações** são necessários para coletar informações que serão utilizadas como base para as adaptações de interface. No caso de uso **Gerenciar Perfil** o usuário fornece informações sobre suas características (e.g., data de nascimento, nível de experiência com computadores). No caso de uso **Gerência de Adaptações** o usuário registra suas preferências acerca da interface e pode, inclusive, reverter adaptações que tenham sido propostas automaticamente pelo sistema.. No sistema proposto, o **Administrador** tem como função principal **Gerenciar Usuários**, que consiste em gerir informações e contas de outros usuários do sistema (criar, consultar, editar e apagar).

3.1.4 Diagramas de Classes

As classes de um modelo representam o modelo inicial do sistema. O paradigma de desenvolvimento orientado a objetos tem como atividade crucial a modelagem de conceitual estrutural, na qual ocorre a identificação das classes do sistema (FALBO, 2017). Considerando-se que um extrato de HCI-ON (apresentado no Capítulo 2) foi utilizado, em tempo de desenvolvimento, como fonte de conhecimento para o desenvolvimento do sistema e, em tempo de execução, como forma de auxiliar na identificação das adaptações a serem feitas na interface de acordo com as características e preferências do usuário (o que será discutido na próxima seção), algumas classes são representação de conceitos presentes na ontologia [17](o que auxilia na persistência dos dados, que será tratada mais adiante). A Figura 5 apresenta o diagrama de classes de SNOPI, destacando-se em azul as classes que foram derivadas de conceitos da ontologia.

[17] Neste capítulo, a ontologia diz respeito a um extrato de HCI-ON utilizado no desenvolvimento do sistema.

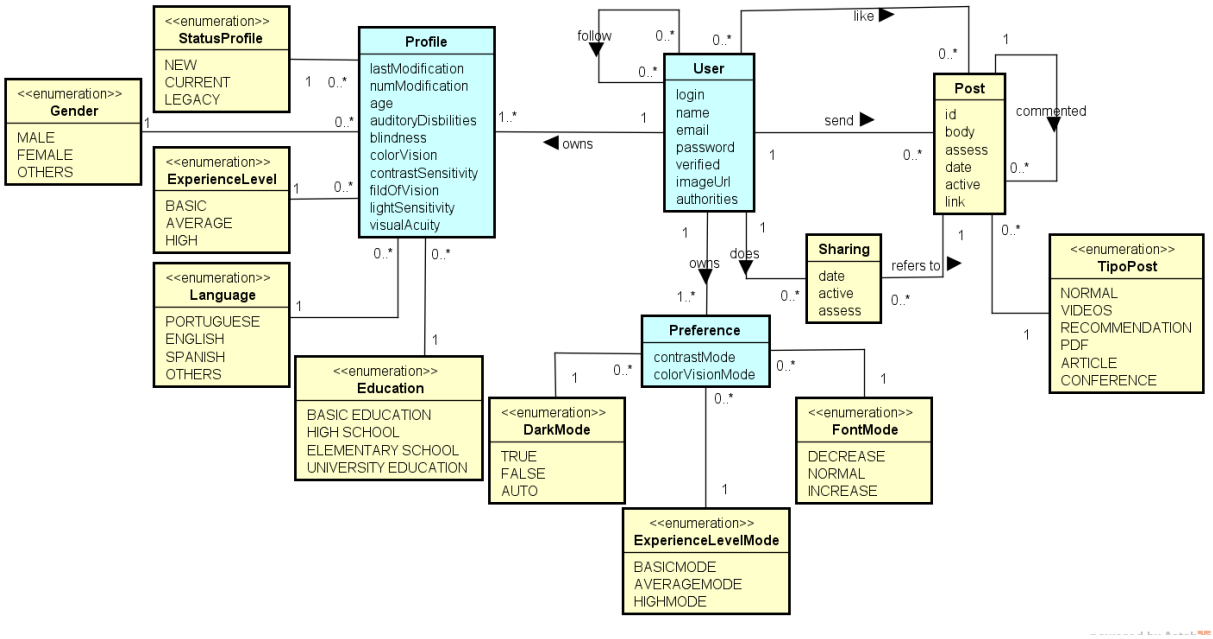

Figura 5 - Diagrama de Classes de SNOPI[18]

Considerando-se que o sistema proposto é uma rede social, o modelo de classes inclui conceitos básicos para as ações principais que uma rede social suporta, sendo elas: criar postagens, curtir postagens e comentar postagens.

A classe **Profile**, baseada no conceito User Profile da ontologia. Nela são armazenadas informações sobre as características dos usuários que, por sua vez, são registrados na classe **User**. Na classe **Profile** também são armazenados metadados úteis ao gerenciamento e melhoria das adaptações da interface. Por exemplo, o atributo *numModificacao* armazena a quantidade de modificações (i.e., inferências) feitas pelo sistema para um usuário para definir sua interface. Muitas modificações pode indicar uma insatisfação do usuário com as sugestões de adaptação de interface feitas automaticamente pelo sistema. A classe associativa **Sharing** tem como finalidade armazenar as informações importantes sobre os compartilhamentos de *posts*.

A classe **Preference** armazena as preferências do usuário em relação à interface do sistema e define como deve ser a interface para esse usuário. Inicialmente, as preferências são

[18] Embora o modelo preveja a possibilidade de um User possuir vários Profiles e várias Preferences, no contexto deste trabalho foi considerado apenas um Profile e uma Preference para cada User.

definidas automaticamente pelo sistema com base em informações contidas em **Profile**. A classe **Preference** foi definida com base no conceito **Customization** da ontologia, o qual se refere às customizações (i.e., adaptações) que devem ser feitas na interface do sistema para um dado usuário, considerando as inferências feitas usando a ontologia operacional.

A classe **Post** armazena as mensagens postadas na rede social e as relações entre diferentes mensagens, que materializam as interações entre usuários do sistema.

3.2 Ontologia Operacional

Uma ontologia operacional, é uma versão de implementação da ontologia de referência que é entendível por máquina e é projetada com o foco em garantir algumas propriedades computacionais desejáveis (GUIZZARDI, 2007). Em outras palavras, enquanto a ontologia de referência foca na semântica no nível conceitual, a ontologia operacional foca em aplicar a semântica no nível de implementação.

3.2.1 Construção da OntoSNOPI

No sistema proposto neste trabalho, informações sobre o perfil dos usuários são utilizadas para definir as adaptações necessárias na interface. Isso é feito em tempo de execução, por meio de inferências sobre instâncias de conceitos da ontologia. Umas das formas de tornar possível as inferências foi o desenvolvimento de uma ontologia operacional. A ontologia operacional foi criada a partir da ontologia de referência equivalente ao extrato de HCI-ON selecionado para ser usado neste trabalho. Durante a criação da ontologia operacional, foram necessários alguns recortes e adaptações na ontologia de referência, visando obter uma ontologia concisa e com escopo reduzido, adequado para as funcionalidades previstas para SNOPI.

No desenvolvimento da ontologia operacional, inicialmente, foram criadas as classes mais gerais e, em seguida, as classes específicas. Como mostra a Figura 6, foram elaboradas duas classes com conceitos mais gerais: Customization e User_Profile, sendo elas subclasses da classe nativa owl:Thing. A classe Customization possui subclasses que representam as adaptações que são realizadas na interface do sistema (Average_Experience_Mode,

Basic_Experience_Mode, Dark_Mode, Desktop_Mode, Font_Decrease, Font_Increase, High_Experience_Mode, Light_Mode, Mobile_Mode). A classe User_Profile, por sua vez, apresenta subclasses sobre características, dificuldades e informações gerais sobre os usuários (User_Age, User_Disability, Auditory_Disabiliti, Vision_Disability, Blindness, Low_Vision, Color_Vision, Contrast_Sensitivity, Fild_Of_Vision, Light_Sensitivity, Visual_Acuity, User_Education, User_Experience_Level, Average_Experience, Basic_Experience, High_Experience, User_Gender, User_Language, Use_ Preferences_Profile).

Utilizando-se as classes definidas na ontologia operacional é possível representar as informações necessárias para identificar as adaptações a serem feitas. Por exemplo, as classes *User_Experience_Level* representa o nível de experiência que o usuário possui em relação a tecnologia e a classe *User_Disabilitiy* captura se o usuário possui alguma deficiência. A Figura 6 ilustra a hierarquia de classes definida.

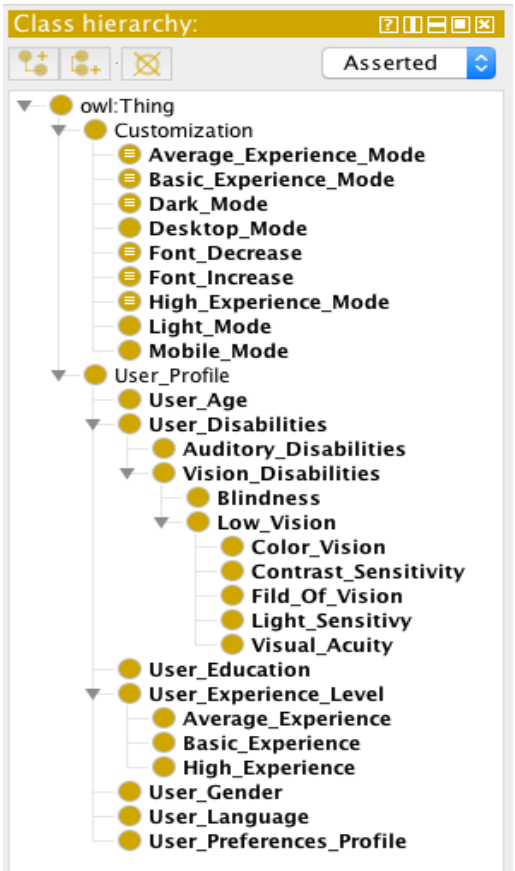

Figura 6 - Ontologia Operacional - hierarquia de classes definida

Após a criação da ontologia no Protégé, foi gerado o arquivo OWL correspondente. Em uma ontologia operacional o conhecimento é representado no formato de tripla *Subject -> Predicate -> Object*. *Subject* e *Object* representam classes da ontologia e Predicate representa a relação que liga a *Subject* e *Object*. Dessa forma, as instâncias dos conceitos da ontologia (i.e., os dados) são armazenadas no formato de um grafo de conhecimento.

Na Tabela 2 são exibidas as classes que foram criadas e suas definições.

Tabela 2 - Classes da ontologia operacional

Classe	Descrição
Customization	Indica possíveis modos de adaptação da interface
User_Profile	Representa o perfil de um usuário.
Subclasse *Customization*	
Average_Experience_Mode	Indica uma adaptação de interface para usuários que possuem experiência média em dispositivos computacionais.
Basic_Experience_Mode	Indica uma adaptação de interface para usuários que possuem experiência baixa em dispositivos computacionais.
Dark_Mode	Indica uma adaptação de interface para o modo escuro.
Desktop_Mode	Indica uma adaptação de interface para ambiente *desktop*.
Font_Decrease	Indica uma adaptação de interface em que há o aumento da fonte dos caracteres para melhorar a legibilidade.
Font_Increase	Indica uma adaptação de interface em que há a diminuição de fonte dos caracteres, para melhorar a legibilidade.
High_Experience_Mode	Indica uma adaptação de interface para usuários que possuem experiência alta em dispositivos computacionais.
Light_Mode	Indica uma adaptação de interface para o modo claro.
Mobile_Mode	Indica uma adaptação de interface para usuário em ambiente *mobile*.
Subclasse *User_Profile*	
User_Age	Representa a idade do usuário.
User_Disabilitiy	Indica uma deficiência do usuário.
Auditory_Disability	Indica deficiência auditiva do usuário.
Vision_Disability	Indica deficiência de visão do usuário.

Blindness	Indica cegueira do usuário.
Low_Vision	Indica algum grau de cegueira do usuário, não sendo total a cegueira.

Tabela 2 - Classes da ontologia operacional (continuação)

Classe	Descrição
Vision_Disability	Indica deficiência de visão do usuário.
Blindness	Indica cegueira do usuário.
Low_Vision	Indica algum grau de cegueira do usuário, não sendo total a cegueira.
Color_Vision	Indica a presença de algum grau de daltonismo.
Contrast_Sensitivity	Indica a presença de algum grau de sensibilidade ao contraste.
Fild_Of_Vision	Indica a presença de algum grau de perda de campo de visão.
Light_Sensitivity	Indica a presença de algum grau de sensibilidade à luminosidade.
Visual_Acuity	Indica a presença de algum grau de alguma doença ocular.
User_Education	Representa o grau de escolaridade do usuário.
User_Experience_Level	Representa o grau de experiência do usuário com dispositivos computacionais.
Average_Experience	Representa experiência média do usuário com dispositivos computacionais.
Basic_Experience	Representa experiência baixa do usuário com dispositivos computacionais.
High_Experience	Representa experiência alta do usuário com dispositivos computacionais.
User_Gender	Representa o gênero do usuário.
User_Language	Representa o idioma do usuário.
Use_ PreferencesProfile	Representa as preferências do perfil do usuário.

Na Tabela 3 são apresentadas as propriedades definidas na ontologia operacional.

Tabela 3 - Propriedades definidas na ontologia operacional

Propriedade	Descrição	Tipo	Domínio	Imagem
has_Disability	Indica a presença de alguma deficiência do usuário.	*Object Property*	*User_Disabilities*	*Customization*

has_Experience_ Level	Indica o grau de experiência com dispositivos computacionais do usuário.	*Object Property*	*User_Experien ce_Level*	*Customization*
has_Age	Referente a idade do usuário	*DataType Porperty*	*User_Profile*	*xsd:integer*

Tabela 3 - Propriedades definidas na ontologia operacional (continuação)

Propriedade	Descrição	Tipo	Domínio	Imagem
has_Genre	Referente ao gênero do usuário.	*DataType Porperty*	*User_Profile*	*xsd:string*
has_Language	Referente a linguagem do usuário.	*DataType Porperty*	*User_Profile*	*xsd:string*
has_Name	Referente ao nome do usuário.	*DataType Porperty*	*User_Profile*	*xsd:string*

Após a criação das classes e propriedades, temos como resultado a estrutura final da ontologia ilustrada na Figura 7.

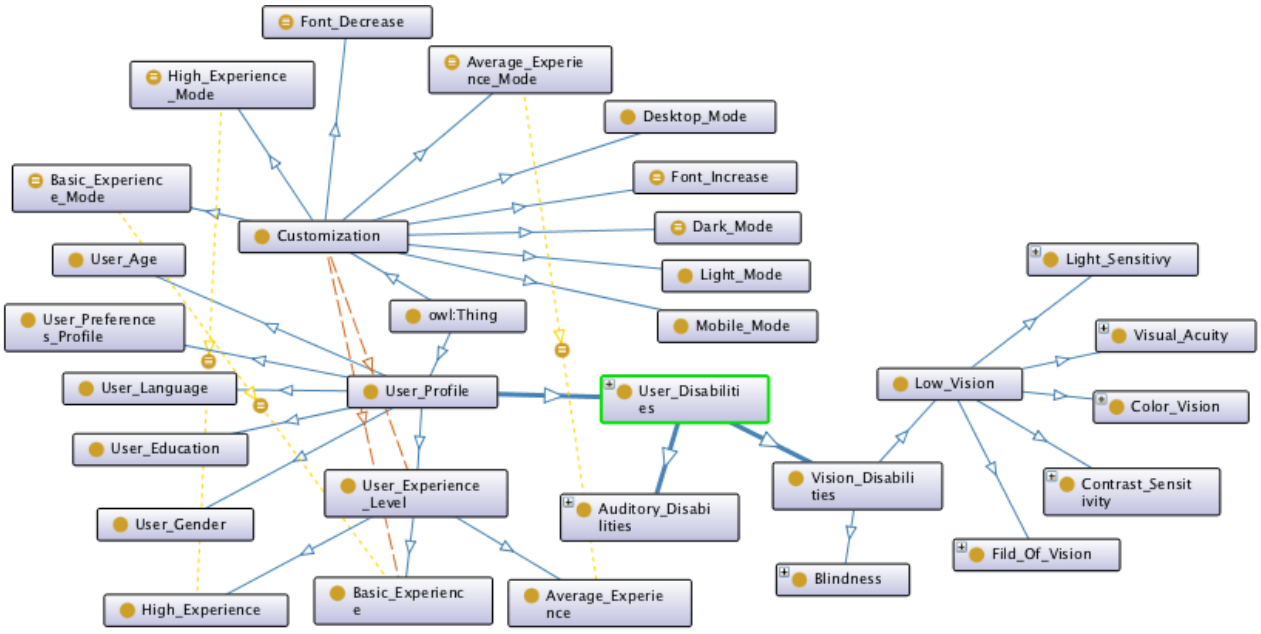

Figura 7 - Representação gráfica da ontologia operacional

Para definir quais adaptações de interface (indicadas na ontologia como subclasses de *Customization*) devem ser feitas considerando as características do usuário (indicadas na ontologia como subclasses de *User_Profile*), foram definidos alguns axiomas. A Figura 8

apresenta, como exemplo, o axioma que indica as condições necessárias para que a adaptação que coloca a interface em modo escuro seja selecionada.

Para que um indivíduo seja atribuído à classe Dark_Mode, é necessário que o indivíduo tenha alguma das seguintes predisposições (has_Disabilities): Daltonismo (Color_Vision) ou Sensibilidade a luminosidade (Light_Sensitivy).

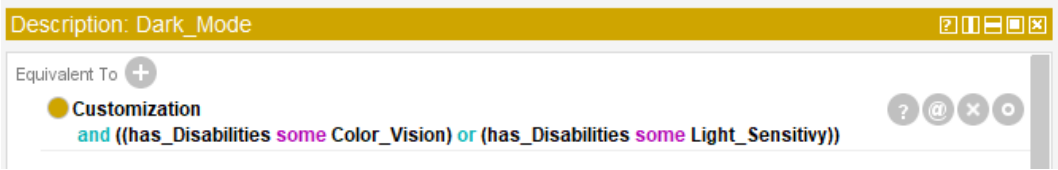

Figura 8 - Axiomas da Classe Dark_Mode

Para a criação dos axiomas que estabelecem as condições para que se aplique cada adaptação de interface, foram consideradas as recomendações da W3C Accessibility[19], foram observados usuários que necessitam de algum tipo especial de acessibilidade e foram consultadas possíveis soluções no Web Content Accessibility Guidelines (WCAG)[20]. Para avaliar a consistência da ontologia, foram realizados alguns testes de inferência no Protégé. Este mecanismo visa encontrar as conexões explícitas e implícitas entre as classes, identificando possíveis inconsistências. Como exemplo, na Figura 9, indicamos um possível indivíduo '01', que tem 62 anos e sensibilidade a luminosidade para testes de inferência. Após a inferência, foi possível identificar, como mostra a Figura 10, que as adaptações de interface para esse indivíduo devem ser "Dark_Mode" e "Font_Increase". O resultado da inferência está consistente com as diretrizes utilizadas como base para a elaboração dos axiomas. Dessa forma, o teste de inferência não demonstrou inconsistências.

[19] *https://www.w3.org/standards/webdesign/accessibility*
[20] *https://www.w3.org/TR/WCAG/*

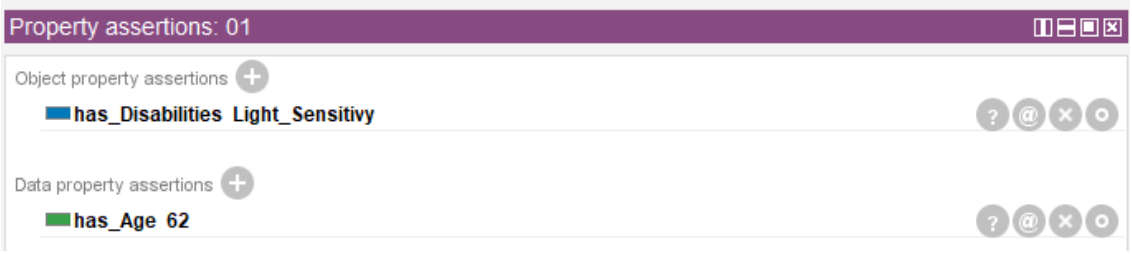

Figura 9 - Propriedades do indivíduo 01

Figura 10 - Inferência sobre o indivíduo 01

3.2.2 Manipulação e *Reasoning* da OntoSNOPI

Como mencionado na seção anterior, este trabalho realiza inferências em tempo de execução para identificar as adaptações de interface adequadas para cada usuário. Para isso, o primeiro passo consiste no carregamento do arquivo OWL de ontoSNOPI para o *back-end* do sistema, pois é no contexto do servidor que ocorre o *reasoning*. Após realizar o carregamento, é necessário criar uma instância da ontologia no sistema e, para isso, foi utilizada a OWL API, criando assim um *OWLOntology*, que é um objeto para representar a ontologia no ambiente JAVA com a OWL API.

A partir da criação da ontologia operacional no sistema, a ontologia deve receber as informações dos usuários e realizar inferências para identificar as adaptações de interface a serem aplicadas (mais detalhes na **Seção 3.5**). Quando um novo usuário realiza o *login* no sistema, ou quando um usuário altera seus dados na página de configurações do seu perfil, o sistema carrega suas informações e as persiste no *back-end* (Figura 11) criando, assim, um indivíduo na ontoSNOPI para representar o usuário.

```
IRI ontologyIRI = IRI.create(
  "http://www.semanticweb.org/interface-adaptativa"
);
OWLDataFactory factory = manager.getOWLDataFactory();

//Carregando Individuo para inferencia
this.loadingIndividual(factory, ontologyIRI, ontology, manager, idUser);

// Reasoner
ReasonerFactory reasonerFactory = new ReasonerFactory();
Configuration configuration = new Configuration();
configuration.throwInconsistentOntologyException = false;
OWLReasoner reasoner = reasonerFactory.createReasoner(
  ontology,
  configuration
);
```

Figura 11 - Fragmento do código utilizado para manipulação de ontologias

Após a existência do indivíduo em uma instância da ontoSNOPI, utiliza-se o Hermit Reasoner para realizar a inferência em tempo de execução. Com base no resultado da inferência, as adaptações são feitas e os dados referentes às adaptações identificadas são persistidos no sistema (para que sempre que o usuário utilizar o sistema a interface esteja adaptada para ele). Uma nova adaptação na interface só é realizada se o usuário solicitar alterando configurações do seu perfil.

3.3 Considerações Finais do Capítulo

Este capítulo apresentou SNOPI, uma rede social desenvolvida para apoiar a disseminação de informações sobre eventos científicos. Foram apresentados os principais resultados produzidos durante a Especificação e Análise de Requisitos, a saber: o propósito do sistema, requisitos, modelos de caso de uso e modelos de classe Também foi apresentada ontoSNOPI, a ontologia operacional desenvolvida para apoiar a identificação das adaptações de interface a serem feitas de acordo com características do usuário.

Capítulo 4

SNOPI: Projeto de Sistema e Implementação

Este capítulo apresenta os principais resultados produzidos ao longo do desenvolvimento do sistema proposto. Na Seção 4.1 são apresentados resultados produzidos durante a fase de projeto do sistema. Na Seção 4.2 são apresentadas algumas telas do sistema. Por fim, na Seção 4.3, são apresentadas as considerações finais do capítulo.

4.1 Projeto de Sistema

O objetivo da fase de projeto de sistema é produzir uma solução para o problema identificado e modelado durante o levantamento e análise de requisitos, incorporando a tecnologia aos requisitos e projetando o que será construído na implementação (BARCELLOS, 2018). A aplicação SNOPI foi desenvolvida utilizando as tecnologias abordadas na Seção 2.5. A seguir é apresentada a arquitetura de SNOPI.

A aplicação foi desenvolvida seguindo um paradigma cliente-servidor (Figura 12) e, portanto, foi dividida em dois subsistemas, sendo que um é executado no lado do cliente (*front-end*) e o outro no lado do servidor (*back-end*).

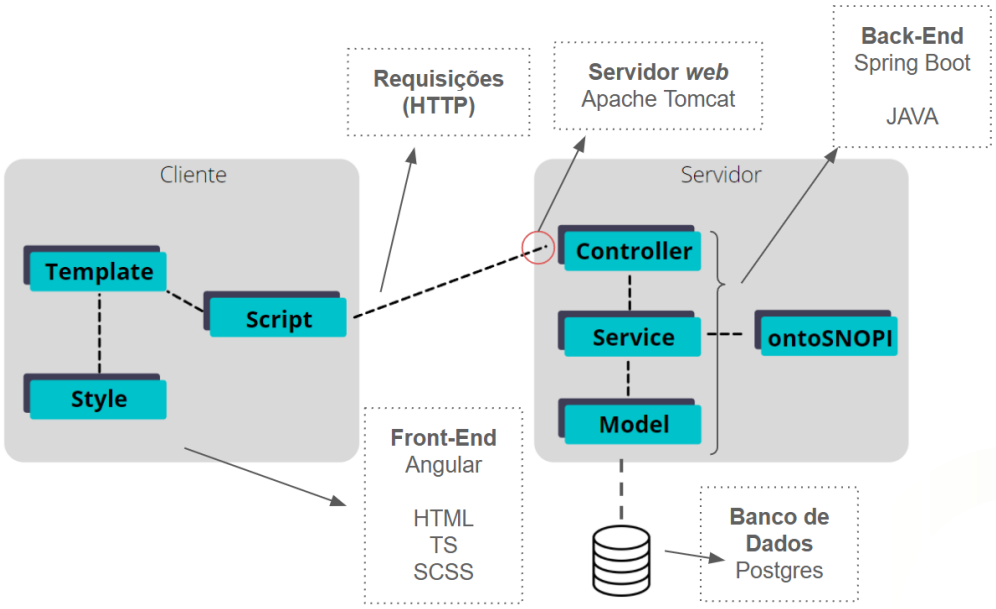

Figura 12 - Diagrama da Arquitetura de Software

O *back-end*, que compreende o servidor, foi desenvolvido utilizando o *Framework Spring*, que tem como base o padrão *Spring Boot*. O mesmo é composto pela camada de controle, camada de lógica de negócio e de gerência de dados, que representam respectivamente a camada de *Controller*, *Service* e *Model* do padrão apresentado. A camada de controle, no padrão *Spring Boot*, tem como responsabilidade principal cuidar das requisições realizadas pela camada de *View*, autorizando e aplicando uma prévia da lógica de negócio apenas para validar a requisição. A camada de lógica de negócio tem como responsabilidade lidar com as regras de negócio impostas à aplicação e foi desenvolvida seguindo o padrão de Modelo de Domínio. A camada de gerência de dados foi desenvolvida com base na camada de lógica de negócio, para criar a persistência dos dados necessários.

O *front-end*, que compreende o cliente, foi desenvolvido utilizando o *framework* Angular, que se baseia no padrão MVVM. O *front-end* é composto pela camada de interface com o usuário, que tem como função lidar com os componentes responsáveis pela exibição do conteúdo e está diretamente relacionada com a camada de *View e View Model* do padrão MVVM. O *front-end* também é composto pela camada de modelo e serviço, que está diretamente associada à camada *Model* do padrão MVVM.

Nas próximas subseções serão apresentadas informações adicionais sobre cada camada da arquitetura de SNOPI.

4.1.1 Camada Lógica de Negócio

Foi utilizado o padrão de Modelo de Domínio para o desenvolvimento da camada lógica de negócio. No padrão Modelo de Domínio, as responsabilidades são distribuídas nos objetos de domínio do problema e a lógica de aplicação é pulverizada nesses objetos, sendo que cada objeto tem uma parte de lógica que é relevante a ele (BARCELLOS, 2018).

4.1.2 Camada de Interface com o Usuário

Sistemas de informação são desenvolvidos para serem utilizados por pessoas. Assim, um aspecto fundamental no projeto é a interface com o usuário. O projeto da interface com o usuário estabelece uma forma de comunicação entre as pessoas, o sistema e como o sistema

apresentará as informações a ele (BARCELLOS, 2018). Neste trabalho, a interface foi de grande importância, pois, ocorrem adaptações na interface para melhorar a experiência do usuário com a aplicação. Para isso, essa camada envolve dois outros componentes, o de Interação Humana e o de Controle de Interação.

Na aplicação, o *front-end* foi implementado utilizando o *framework Angular*. O *framework* segue uma arquitetura específica que se baseia com o padrão Modular, que cria um ambiente modularizado por entidades de visualização, visando diminuir a complexidade de desenvolvimento. No *Angular*, estes são elementos básicos para seu funcionamento: *Component, Template, Service* e *Module*. Sendo que cada um possui uma função específica, sendo elas: *Component* (fazendo parte do Componente de Interação Humana (CIH)) - define uma classe que contém dados e a lógica de negócio. Está associado a modelos HTML; *Template* (fazendo parte do CIH) - combina HTML com marcação *Angular*, que pode modificar elementos HTML antes de serem exibidos; *Service* (fazendo parte do Componente de Controle de Interação (CCI)) - declara uma exibição específica e pode ser compartilhado entre componentes; *Module* (fazendo parte do CCI) - declaram um contexto de compilação para um conjunto de componentes dedicado a um escopo da aplicação.

4.1.2.1 Componente de Interação Humana

O Componente de Interação Humana corresponde à interface da aplicação, que é constituída por elementos gráficos ou textuais, para facilitar a comunicação com o usuário (FALBO, 2018). O desenvolvimento da interface começou com a prototipação de algumas telas para começar a estudar os elementos necessários e suas variações. A interface é um dos principais componentes do sistema, pois está conectada diretamente com as adaptações. Com isso, o nível de complexidade das interfaces aumenta e torna a prototipação ainda mais necessária.

No primeiro momento da prototipação foi utilizado o *Figma*[21] para desenvolver os esboços do corpo da aplicação. Como a aplicação se trata de um protótipo de rede social, foram utilizadas como fontes de inspiração outras redes sociais presentes na internet, sendo

[21] https://www.figma.com/

algumas delas: *Facebook* [22]e *Twitter*[23]. A segunda etapa da prototipação consistiu na criação dos *mockups*[24] para identificação do design antes da implementação. Este processo também foi realizado no *Figma*. Na Figura 13, é apresentado como exemplo, uma tela do do *mockup* produzido. Durante a fase de criação de *mockups,* todas as cores escolhidas para fazerem parte da paleta de cores da interface do sistema foram testadas e aprovadas pela ferramenta online de acessibilidade *Adobe Color*[25], sendo assim, o sistema, desde em seu estado inicial, utilizou cores adequadas para pessoas com algum grau de daltonismo.

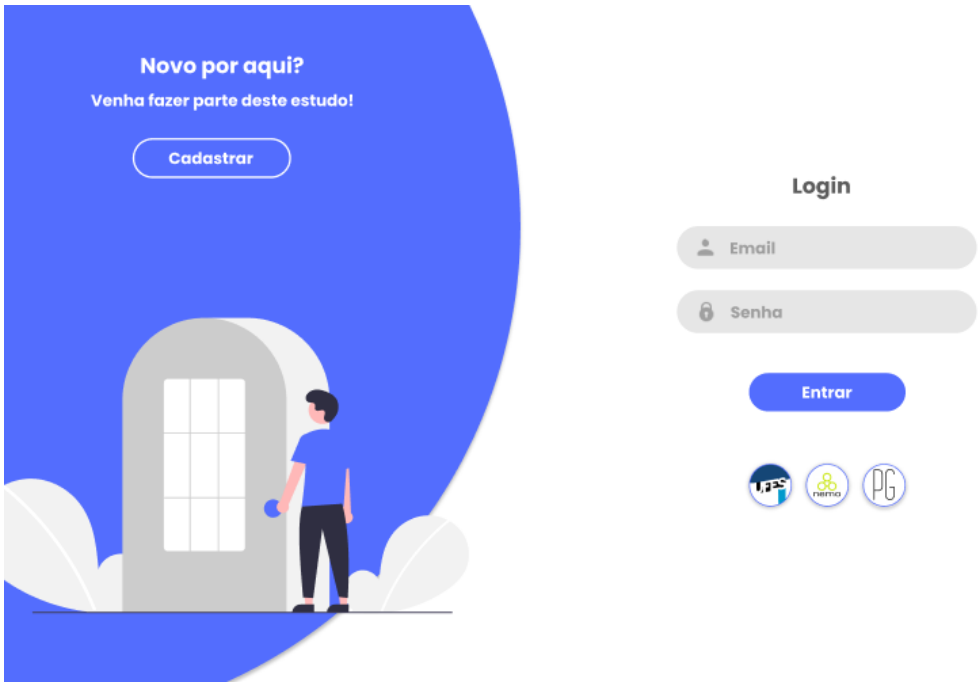

Figura 13 - Mockup da tela de login do sistema

Utilizando o *Angular* no desenvolvimento da interface do usuário, foi possível facilitar a implementação e escalabilidade do projeto, pois o mesmo possui padrão modular. Na Figura 14, podemos observar que o *Angular* segue o conceito de Entidade como uma representação

[22] https://www.facebook.com/
[23] https://www.twitter.com/
[24] Um *mockup* consiste em uma representação mais próxima do software a ser desenvolvido.
[25] https://color.adobe.com/pt/create/color-accessibility

visual para o usuário. Cada entidade pode possuir um template (**HTML** e **Typescript**), um *style* (**SCSS**), uma lógica de negócio (**Service**) e um módulo (**Module**). Dessa forma, o *Angular* consegue realizar a componentização de estruturas, fazendo ser possível criar subcomponentes para serem utilizados em várias partes do sistema. A Figura 15, mostra um fragmento de código de um *template* da aplicação.

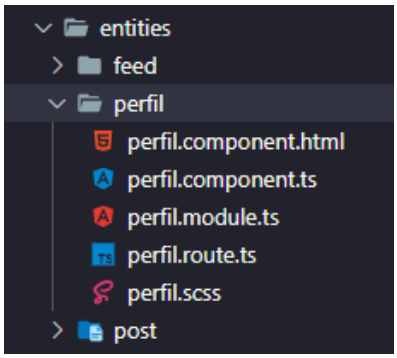

Figura 14 - Estrutura de organização Angular

```
<div class="body-adapative">
    <div class="sidebar-left">···
    </div>

    <div class="main main_perfil">
        <div id="header_perfil" class="card-post header_perfil">
            <div class="avatar">
                <figure *ngIf="user?.imageUrl">
                    <img [src]="user?.imageUrl" alt="avatar do usuario" class="avatar" style="border: 2px solid;">
                </figure>
                <div *ngIf="!user?.imageUrl" class="avatar"></div>
            </div>
            <div class="name">{{user?.login}}</div>
            <div class="seguir" *ngIf="user?.id != account?.id">
                <button class="btn_feed btn_seguir" *ngIf="!isFriend" (click)="friends()">
                    <i class="bi bi-bookmark-heart"></i> Seguir</button>
                <button class="btn_feed btn_feed-cor-secondary btn_seguir" *ngIf="isFriend" (click)="friends()">
                    <i class="bi bi-bookmark-heart-fill"></i> Deixar de Seguir</button>
            </div>
            <div class="descricao"></div>
        </div>
        <div id="body_perfil" class="body_perfil">···
        </div>
    </div>

    <div class="sidebar-right">···
    </div>
</div>
```

Figura 15 - Fragmento de código do componente de template Perfil do Angular.

Após o desenvolvimento dos elementos de interação humana, os códigos implementados foram testados na W3C *Markup*[26], sendo possível que leitores de textos/programas dos usuários consigam navegar pela página sem dificuldades e também possam identificar os elementos da interface, contribuindo para que se tenha um ambiente com melhor acessibilidade.

4.1.2.2 Componente de Controle de Interação

O componente de controle de interação tem como função realizar a comunicação entre as regras de lógica de negócio e a interface do usuário. No *Angular*, este componente é implementado utilizando a linguagem de programação ***Typescript*** (Figura 16), que desenvolve métodos para auxiliar os elementos visuais e outros componentes da aplicação.

```
import { HttpResponse } from '@angular/common/http';···

You, last month | 1 author (You)
@Component({
  selector: 'jhi-perfil',
  templateUrl: './perfil.component.html',
  styleUrls: ['perfil.scss'],
})
export class PerfilComponent implements OnInit {
  account: any = null;
  user: IUser = null;
  profile: IProfile = null;
  post: IPost[] = null;
  loading = false;
  feed: IPost[] = [];
  loginProfile: string;
  isFriend = false;

  constructor(
    private postService: PostService,
    private loginService: LoginService,
    private router: Router,
    protected activatedRoute: ActivatedRoute,
    private userService: UserService,
    private accountService: AccountService,
    private profileService: ProfileService
  ) {}

  ngOnInit(): void {
    this.accountService
      .getAuthenticationState()
      .subscribe(account => (this.account = account));
```

Figura 16 - Fragmento do código do componente script do Angular

[26] *https://www.validator.w3.org/*

4.1.2.3 Camada de Gerência de Dados

O banco de dados do sistema segue o modelo relacional (***Postgresql***) e é utilizado pelo *back-end* com o *Spring Boot*. Ele cria um ambiente de desenvolvimento para lidar com a persistência dos dados do sistema, sendo apenas necessário configurar o banco de dados no *Spring Boot* e, utilizar a interface e os recursos fornecidos pelo *framework* para gerenciar a persistência das classes de interesse.

4.2 SNOPI - Social Network with Ontology-based Adaptive Interface

Nesta seção, são apresentadas as telas da ferramenta e uma breve descrição do fluxo de utilização da aplicação. As Figuras, apresentadas nesta seção ilustram o acesso de um usuário hipotético ao sistema.

A aplicação está hospedada[27] e disponível para uso. Para conseguir utilizar as funções do sistema é necessário realizar a autenticação. Na Figura 17, é apresentada a tela de *login*, que é a primeira tela visível ao usuário é nela que ele deve realizar a autenticação.

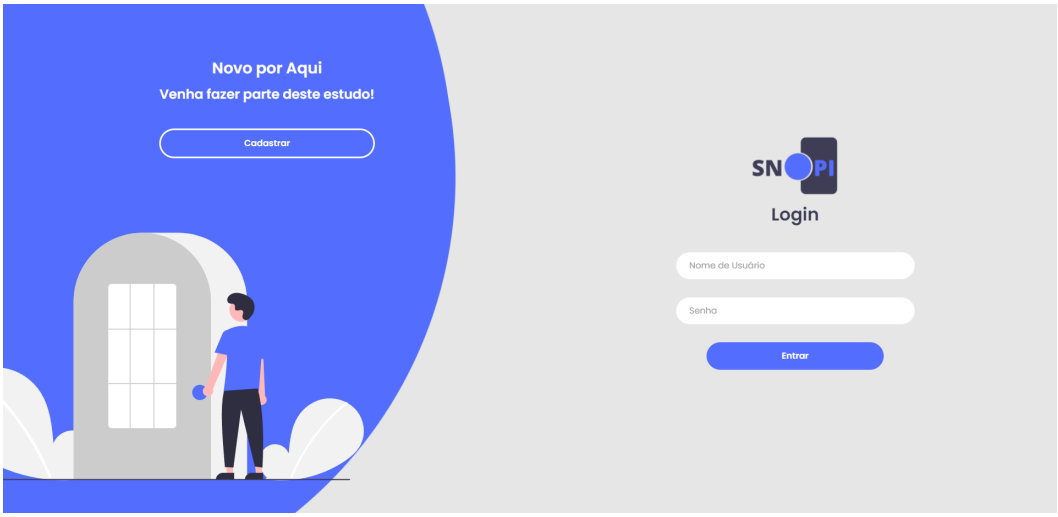

Figura 17 - Tela de Login

Caso o usuário ainda não tenha um *login* de acesso, ele deve se cadastrar na aplicação. Para isso, deve clicar em *Cadastrar* (Figura 17) e preencher os dados necessários para o cadastro (Figura 18).

[27] https://bit.ly/adaptativeinterface

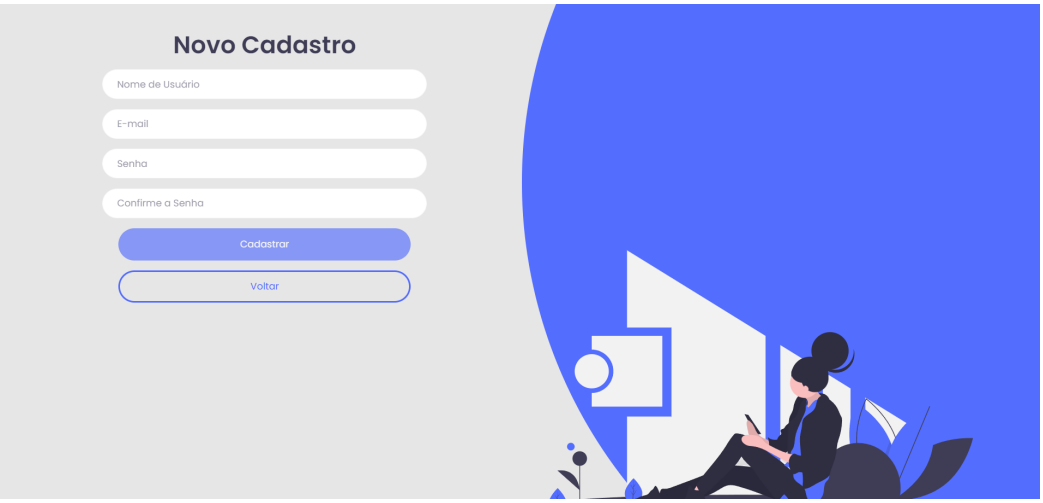

Figura 18 - Tela de Cadastro de Usuário

Após o login, o usuário é direcionado para a página do *Feed* (Figura 19), que é a página que concentra as postagens de todos os usuários da plataforma. Nessa página, à esquerda, o usuário encontra um menu de acesso rápido e, à direita, ele encontra informações complementares sobre o sistema. No *Feed*, o usuário pode realizar pesquisas na barra de busca, pode publicar postagens, que podem possuir categorias e *links*. No *Feed*, o usuário também pode curtir, visualizar outros perfis de usuários e comentar outras postagens.

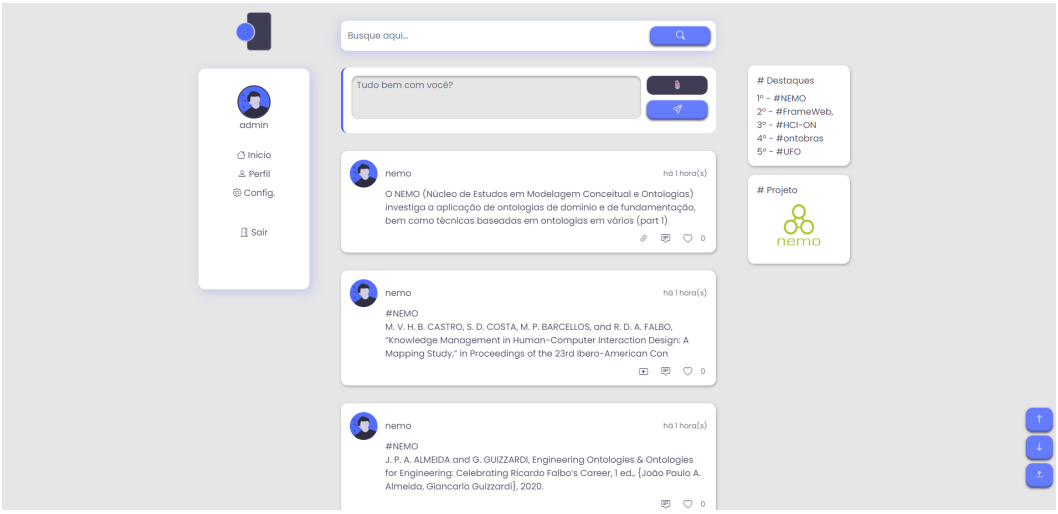

Figura 19 - Tela do Feed

Logo após o primeiro *login*, o sistema identifica o novo usuário e apresenta para ele algumas telas com perguntas (Figura 20 e Figura 21) para obter dados do usuário e avaliar seu

perfil. A pesquisa é composta por algumas perguntas (Figura 21) que ajudarão a identificar as adaptações que serão realizadas na interface para aquele usuário. Responder a pesquisa é opcional, mas, se o usuário não respondê-la, a interface não sofrerá adaptações.

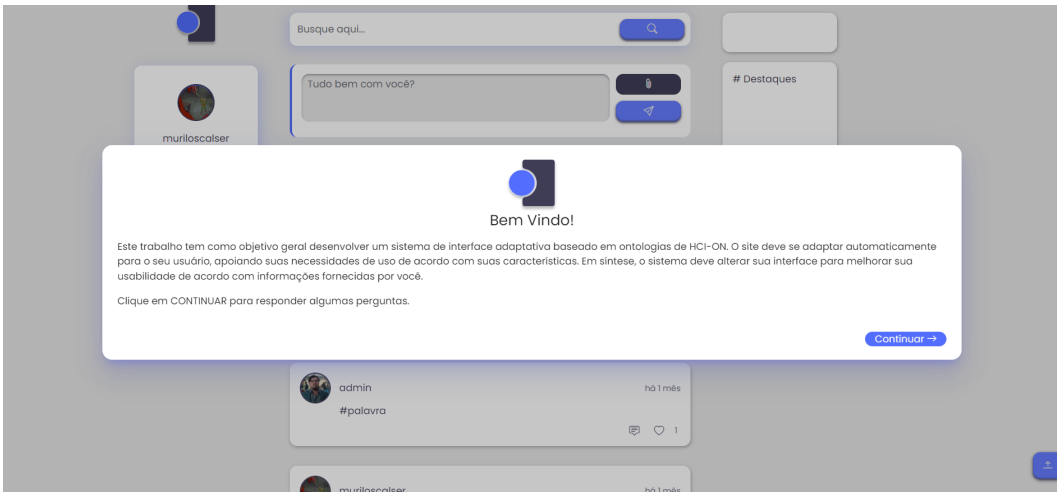

Figura 20 - Modal de Pesquisa ao Usuário

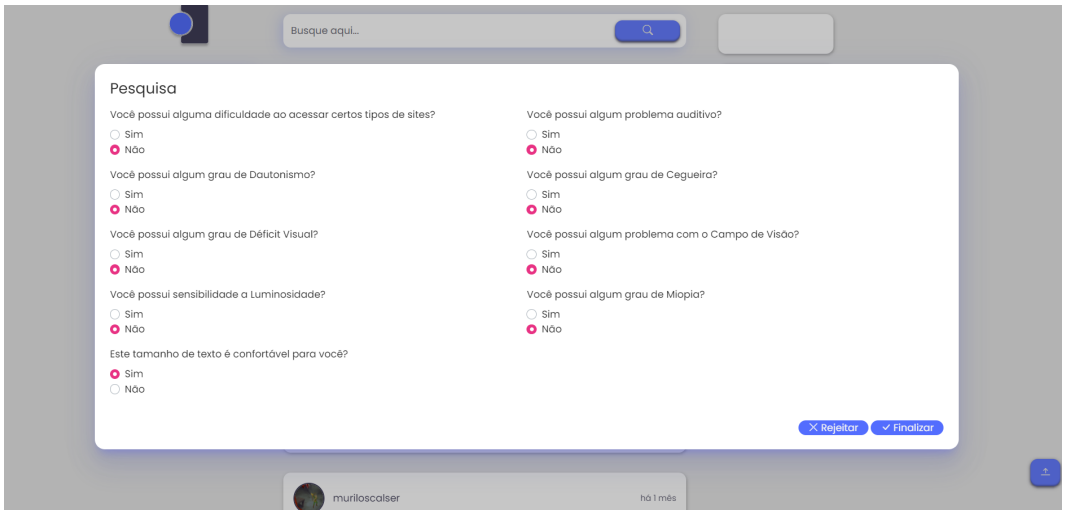

Figura 21 - Modal de Pesquisa ao Usuário com perguntas respondidas

Após o usuário responder a pesquisa, o sistema envia as respostas ao servidor *back-end*, que processa as informações e utiliza a ontologia operacional para inferir um perfil de adaptações ao usuário de acordo com suas respostas. Esses dados são armazenados no banco de dados e uma requisição retorna ao *front-end* da aplicação contendo as adaptações inferidas. Neste ponto, a página se recarrega e aplica as alterações necessárias. A Figura 22 e a Figura 23 são exemplos de adaptações realizadas na interface e exibidas ao usuário. A adaptação

observada na Figura 22 leva em conta que o usuário se encaixou no perfil de Modo Escuro, no qual a interface tem as cores alteradas visando à pouca emissão de luz, deixando a aplicação mais confortável em certas ocasiões. Já a adaptação apresentada na Figura 23 leva em conta que o usuário se encaixou no perfil de Modo Alto Contraste, onde a interface tem as cores alteradas para aumentar o contraste, deixando-a com pouca interferência visual, aumentando o foco para os componentes.

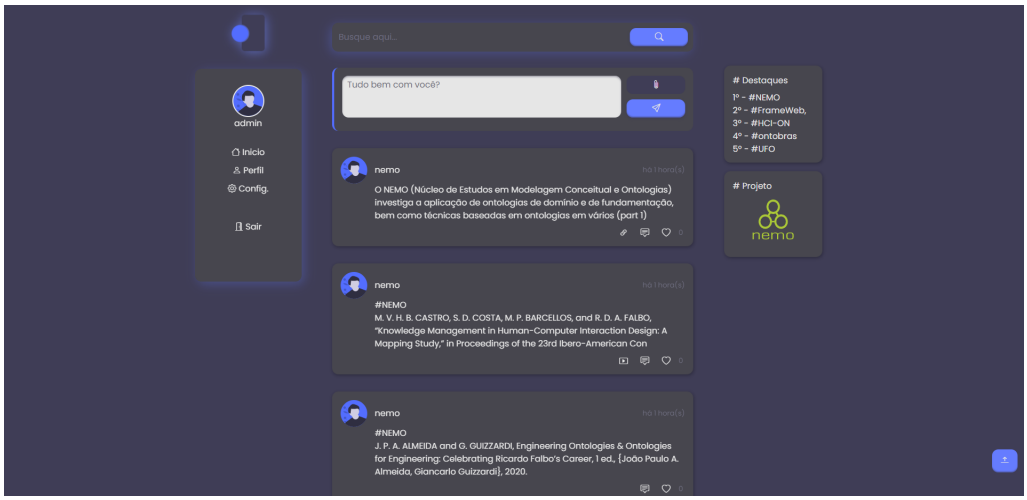

Figura 22 - Tela de Feed do sistema com adaptação - Modo Escuro

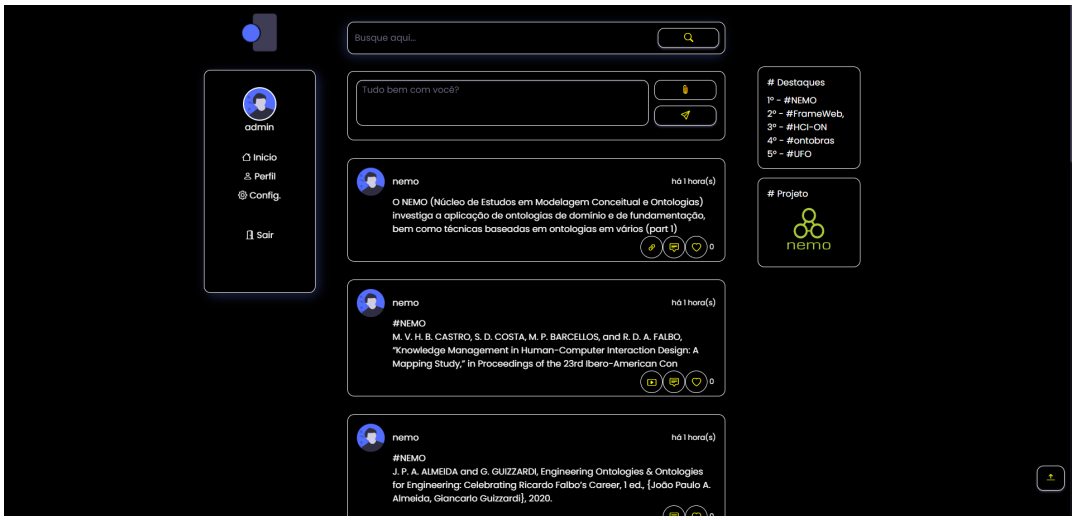

Figura 23 - Tela de Feed do sistema com adaptação - Modo Alto Contraste

A aplicação também conta com a página de usuários, chamada Perfil, onde mostra as postagens do usuário e este nome 'Perfil' não equivale ao conceito *Profile* mencionado na ontologia operacional. O Perfil do usuário que está logado pode ser acessado clicando-se no botão no menu lateral da esquerda. Para acessar o perfil de outros usuários, deve-se clicar na foto do usuário desejado. Na página Perfil (Figura 24), o usuário encontra as postagens realizadas por ele, quando acessando o seu próprio perfil, e encontra postagens realizadas por outros usuários, quando acessado o perfil de outro usuário. Também no perfil, é possível filtrar de acordo com a categoria de postagens, sendo elas: Geral, Respostas, Vídeos, PDF, Artigos ou Conferência. Desta forma, possibilita-se uma organização maior de suas postagens.

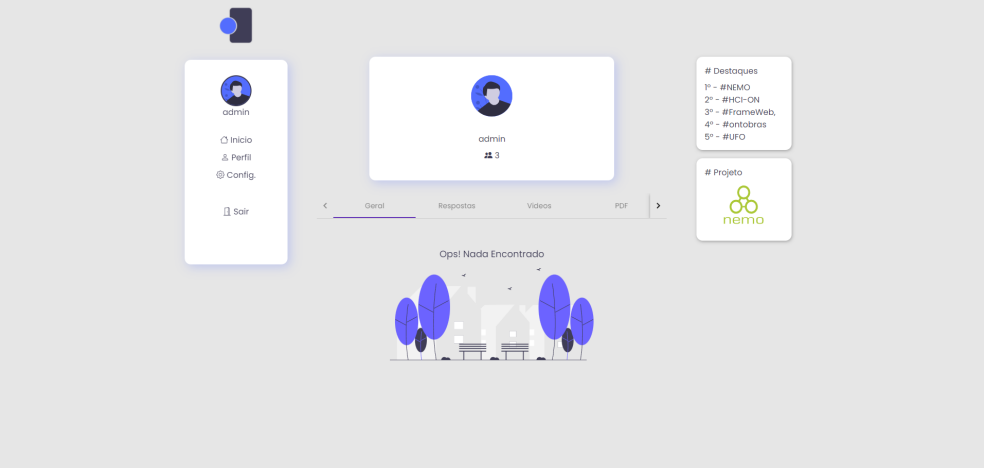

Figura 24 - Tela de Perfil do Usuário

O sistema também conta com uma página de configurações (Figura 25), na qual o usuário consegue alterar sua imagem de perfil (adicionando uma URL pública de uma imagem). Nesta tela, o usuário também pode acessar as configurações de interface definidas automaticamente pelo sistema com base em seu perfil e pode alterar essas configurações ou resetá-las para a configuração padrão. Caso o usuário faça isso, o sistema apresentará para ele a opção de responder a pesquisa novamente para que o sistema possa identificar novas adaptações de interface.

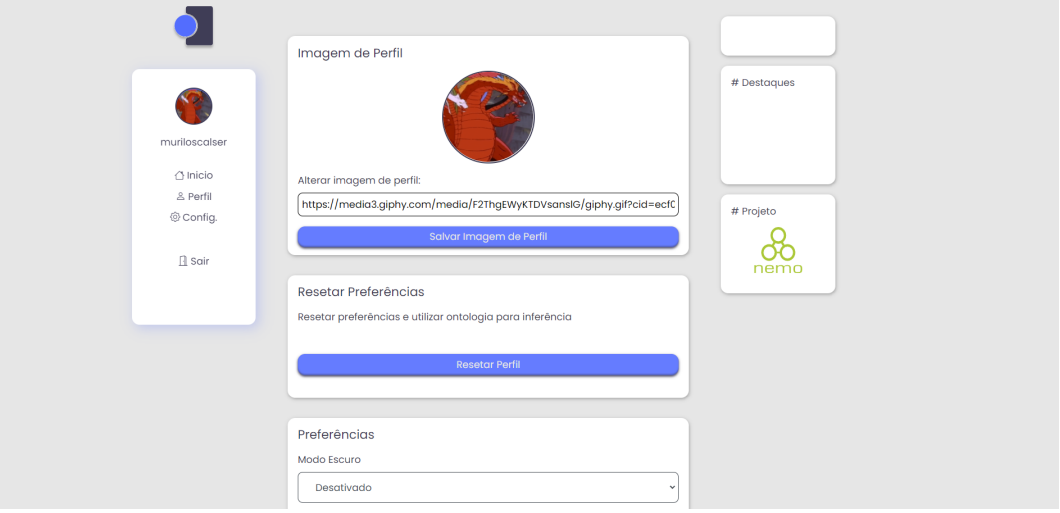

Figura 25 - Tela de Configurações

4.3 Considerações Finais do Capítulo

Este capítulo apresentou SNOPI, uma rede social desenvolvida para apoiar a disseminação de informações sobre eventos científicos. Este capítulo apresentou resultados das fases de Projeto do Sistema e Implementação. A arquitetura do sistema foi apresentada e alguns de seus componentes foram discutidos. Questões relacionadas à implementação foram discutidas e foram apresentadas telas do sistema desenvolvido. No próximo capítulo, são apresentadas as considerações finais do trabalho.

Capítulo 5

Considerações Finais

Neste capítulo são realizadas as considerações finais deste trabalho, sendo apresentadas suas principais contribuições e perspectivas de trabalhos futuros.

5.1 Conclusões

Neste trabalho, foi desenvolvido o sistema SNOPI, um protótipo de uma rede social com interface adaptativa. Este trabalho foi desenvolvido no contexto da tese de doutorado de seu coorientador, na qual investiga-se o uso de ontologias para apoiar o desenvolvimento de sistemas de interfaces adaptativas. Também está relacionado ao trabalho de doutorado de Costa (2021) , na qual está em desenvolvimento a HCI-ON (*Human-Computer Interaction Ontology Network*), uma rede de ontologias que tratam aspectos relacionados a IHC e destinada a apoiar o desenvolvimento de soluções nesse domínio.

Ao longo do desenvolvimento deste trabalho, foram aplicados alguns conhecimentos adquiridos durante a graduação em diferentes disciplinas. Os principais conhecimentos aplicados neste trabalho foram obtidos por meio das disciplinas a seguir:

- **Engenharia de *Software* e Engenharia de Requisitos**, onde foram apresentados conhecimentos sobre o processo no qual os requisitos de um produto de software são coletados, analisados, documentados e gerenciados;
- **Projeto de Sistema de *Software*,** onde foram apresentados conhecimentos para definir e especificar soluções de projeto de sistema, visando à implementação do sistema;
- **Programação II e III**, que deram a base de conhecimento para a lógica de programação, explorando particularidades sobre linguagens estudadas nas disciplinas e apresentando desafios de programação, que foram aplicados neste trabalho;
- **Engenharia de Ontologias**, onde foram apresentados conhecimentos sobre a construção de ontologias com foco em ontologias de referência, que neste

trabalho foi importante para o entendimento do domínio, desenvolvimento da ontologia de referência e construção da ontologia operacional;

- **Banco de Dados**, onde foram apresentados conhecimentos de base sobre estruturação, modelagem e utilização de persistência em bancos relacionais, os quais foram utilizados neste trabalho.

No Capítulo 1 desta monografia foram apresentados os objetivos que deveriam ser alcançados por este trabalho. Na Tabela 4 são apresentados os resultados relacionados a cada objetivo ao final do desenvolvimento deste trabalho.

Tabela 4 - Objetivos e seus resultados

Objetivo	Situação	Resultado
Investigar a literatura sobre o uso de ontologias no desenvolvimento de interfaces adaptativas.	Atendido	Conhecimento sobre interfaces adaptativas e ontologias vide seções 2.3, 2.4) e resultados do mapeamento sistemático que encontra-se em andamento e será registrado em um artigo (não foi incluído neste texto).
Especificar o sistema a ser produzido, estabelecer seus requisitos, bem como caracterizando quais aspectos e funcionalidades serão adaptados pelo sistema para apoiar ao usuário.	Atendido	Especificação de requisitos, modelo de casos de uso e diagrama de classes da aplicação e a construção da ontologia operacional (vide seções 3.1 e 3.2).
Projetar a solução a ser construída.	Atendido	Elaboração do projeto de sistema do SNOPI (vide Seção 4.1).
Implementar e testar o sistema com interface adaptativa.	Atendido	Sistema SNOPI desenvolvido (vide Seção 4.2).

Entre as dificuldades para o desenvolvimento deste trabalho, destaca-se a escolha das tecnologias envolvidas e como utilizá-las. O desenvolvimento da ontologia operacional também foi um desafio, pois o autor desta monografia nunca havia trabalhado com desenvolvimento de ontologias operacionais e a escolha da tecnologia para implementação poderia afetar todo o desenvolvimento do sistema.

É importante ressaltar que este trabalho propõe uma versão inicial do sistema, que possui algumas limitações. Algumas dessas limitações estão relacionadas à ontologia operacional e outras à rede social em si.

Em relação à ontologia operacional, foi necessário fazer um recorte da ontologia de referência para construção da ontologia operacional. Este recorte foi inevitável, pois o tempo de desenvolvimento foi limitado, sendo assim, alguns conceitos que poderiam levar a outras adaptações de interface foram desconsiderados, deixando a ontologia mais concisa e com escopo reduzido, tornando possível atender as funcionalidades propostas neste trabalho. Também devido à restrição de tempo, algumas funcionalidades da rede social ficaram incompletas (por exemplo, o requisito RF06, que trata do compartilhamento de postagem não foi implementado), mas, ainda assim, a aplicação é funcional e cumpre com seu propósito.

Cabe notar que, embora uma das motivações para o desenvolvimento do sistema tenha sido a necessidade de centralizar o compartilhamento de informações sobre eventos científicos e assuntos relacionados, o sistema não é limitado a esse domínio de aplicação. De fato, o sistema foi desenvolvido de forma independente do domínio de aplicação, para ampliar seu potencial de uso.

Por fim, vale ressaltar que o sistema foi disponibilizado abertamente em um link para público, mas é uma versão inicial produzida no contexto de um estudo para explorar o uso de ontologias no desenvolvimento de interfaces adaptativas. Não está no escopo desta monografia discutir o estudo em si, mas pode-se dizer que os resultados foram positivos, pois, como apresentado no Capítulo 3, ontologias podem ser usadas para realizar inferências considerando as características dos usuários e identificar as adaptações de interface adequadas.

5.2 Trabalhos Futuros

Como trabalhos futuros, os seguintes pontos foram identificados como melhorias na ferramenta:

- Melhorar o modo como as adaptações estão sendo carregadas pelo *front-end* (às vezes o carregamento não é rápido).
- Investigar novas diretrizes de adaptação de interface e definir novos axiomas para inferir adaptações necessárias para o usuário.
- Implementar um ranking de adaptações úteis aos usuários, permitindo que o usuário possa votar em uma adaptação.
- Monitorar as adaptações realizadas pelo usuário, para melhorar o entendimento acerca das preferências e comportamento do usuário e permitir melhores adaptações automáticas baseadas não só nas suas preferências, mas também no seu comportamento.
- Analisar grupos de usuários para definir adaptações para grupos de usuários (atualmente a inferência pela ontologia operacional leva em consideração apenas um usuário).
- Aumentar o número de adaptações possíveis.
- Armazenar informações referentes a curtidas e comentários dos usuários.
- Melhorar o carregamento de *posts* no *Feed* da aplicação.
- Solicitar validação por e-mail, para aumentar a segurança do cadastro do usuário.
- Realizar a avaliação de *user experience* da plataforma com usuários.

Referências Bibliográficas

BARCELLOS, Monalessa P.: Engenharia de Software: Notas de aula. 2018. Departamento de Informática, Universidade Federal do Espírito Santo, Vitória-ES. Disponível em: <https://nemo.inf.ufes.br/wp-content/uploads/Monalessa/EngSoftware/NotasDeAula-EngSw-EngComp-v2018.pdf >. Acesso em: 1 março 2022.

BENYON, David; INNOCENT, Peter; MURRAY, Dianne. System adaptivity and the modelling of stereotypes. In: Human–Computer Interaction–INTERACT'87. North-Holland, 1987. p. 245-253.

BENYON, David; MURRAY, Dianne. Experience with adaptive interfaces. The Computer Journal, v. 31, n. 5, p. 465-473, 1988.

CARROLL, John M. Human Computer Interaction (HCI). In: Soegaard, Mads and Dam, Rikke Friis (eds.). Encyclopedia of Human-Computer Interaction. Aarhus, Denmark: The Interaction Design Foundation. 2012.

COSTA, Simone Dornelas et al. A core ontology on the Human-Computer Interaction phenomenon. Data & Knowledge Engineering, p. 101977, 2022.

COSTA, Simone Dornelas; BARCELLOS, Monalessa Perini; FALBO, Ricardo de Almeida; CASTRO, Murillo Vasconcelos Henriques Bittencourt. Towards an Ontology Network on Human-Computer Interaction. In: 39th International Conference on Conceptual Modeling (ER 2020) Springer, Cham, 2020. p. 331-341.

COSTA, Simone Dornelas. An Ontology Network to support Knowledge Representation and Semantic Interoperability in the HCI Domain. Proposta de Tese de Doutorado-Programa de Pós-Graduação em Informática (PPGI), Universidade Federal do Espírito Santo (UFES), Vitória, 2021.

FALBO, Ricardo A.: Engenharia de Requisitos: Notas de aula. 2017. Departamento de Informática, Universidade Federal do Espírito Santo, Vitória-ES. Disponível em: <http://www.inf.ufes.br/~jssalamon/wp-content/uploads/disciplinas/engreq/Notas_Aula_Engenharia_Requisitos.pdf>. Acesso em: 1 março 2022.

FALBO, Ricardo A.: Engenharia de Software: Notas de aula. 2014. Departamento de Informática, Universidade Federal do Espírito Santo, Vitória-ES. Disponível em: <http://www.inf.ufes.br/~jssalamon/wp-content/uploads/disciplinas/engsoft/Notas_Aula_Engenharia_Software_Falbo_2014.pdf>. Acesso em: 1 março 2022.

FALBO, Ricardo A.: Engenharia de Software: Notas de aula. 2018. Departamento de Informática, Universidade Federal do Espírito Santo, Vitória-ES. Disponível em: <http://www.inf.ufes.br/~jssalamon/wp-content/uploads/disciplinas/projsistsoft/Notas_Aula_Projeto_Sistemas_2018.pdf>. Acesso em: 11 março 2022.

FREITAS, Alexandre Adler Cunha. Uma Abordagem Baseada em Ontologias para Apoio ao Desenvolvimento de Sistemas de Interfaces Adaptativas. 2022, No prelo.

GRUBER, Thomas R. Toward principles for the design of ontologies used for knowledge sharing. International journal of human-computer studies, v. 43, n. 5-6, p. 907-928, 1995.

GUIZZARDI, Giancarlo, ALESSANDERC, Botti Benevides, CLAUDENIR, M. Fonseca, DANIELED, Porello, JOÃO PAULO, Almeida, TIAGO, Prince Sales. UFO: Unified Foundational Ontology. Applied Ontology, n. Preprint, p. 1-44, 2021.

GUIZZARDI, Giancarlo. On ontology, ontologies, conceptualizations, modeling languages, and (meta) models. Frontiers in artificial intelligence and applications, v. 155, p. 18, 2007.

GUIZZARDI, Giancarlo. Ontological foundations for structural conceptual models. 2005.

JAMESON, Anthony. Adaptive interfaces and agents. The human-computer interaction handbook: Fundamentals, evolving technologies and emerging applications, p. 305-330, 2008.

LANGLEY, Pat. User modeling in adaptive interface. In: UM99 User Modeling. Springer, Vienna, 1999. p. 357-370.

NBR, ABNT. 9241-11. Requisitos ergonômicos para trabalho de escritório com computadores: Parte 11 - Orientação sobre usabilidade. ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. Rio de Janeiro: sn, p. 21, 2002.

NBR, ABNT. 9241-210. Ergonomia da interação homem-sistema: Parte 210 - Projeto centrado no ser humano para sistemas interativos. ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. Rio de Janeiro, 2011.

OPPERMANN, Reinhard (Ed.). Adaptive user support: ergonomic design of manually and automatically adaptable software. CRC Press, 1994.

PETERSEN, Kai; VAKKALANKA, Sairam; KUZNIARZ, Ludwik. Guidelines for conducting systematic mapping studies in software engineering: An update. Information and software technology, v. 64, p. 1-18, 2015.

PRESSMAN, R. S.; LOWE, D. Web engineering: a practitioner's approach. [S.l.]: McGraw-Hill Education, 2009. v. 1.

RUY, Fabiano Borges; FALBO, Ricardo de Almeida; BARCELLOS, Monalessa Perini; COSTA, Simone Dornelas; GUIZZARDI, Giancarlo. SEON: A software engineering ontology network. In: European Knowledge Acquisition Workshop. Springer, Cham, 2016. p. 527-542.

SCHERP, Ansgar; SAATHOFF, Carsten; FRANZ, Thomas; STAAB, Steffen. Designing core ontologies. Applied Ontology, v. 6, n. 3, p. 177-221, 2011.

SCHNEIDER-HUFSCHMIDT, Matthias; MALINOWSKI, Uwe; KUHME, Thomas (Ed.). Adaptive user interfaces: Principles and practice. Elsevier Science Inc., 1993.

STUDER, Rudi; BENJAMINS, V. Richard; FENSEL, Dieter. Knowledge engineering: Principles and methods. Data & knowledge engineering, v. 25, n. 1-2, p. 161-197, 1998.