

See discussions, stats, and author profiles for this publication at: <https://www.researchgate.net/publication/372512499>

Using Networked Ontologies to Support the Development of Software Systems with Adaptive User Interface

Article in *Journal on Interactive Systems* · July 2023

DOI: 10.5753/jis.2023.3256

CITATIONS

0

READS

79

4 authors, including:



Alexandre Adler Freitas

Universidade Federal do Espírito Santo

2 PUBLICATIONS 2 CITATIONS

[SEE PROFILE](#)



Simone Dornelas Costa

Universidade Federal do Espírito Santo

12 PUBLICATIONS 87 CITATIONS

[SEE PROFILE](#)




Monalessa Perini Barcellos

Universidade Federal do Espírito Santo


111 PUBLICATIONS 721 CITATIONS

[SEE PROFILE](#)

Using Networked Ontologies to Support the Development of Software Systems with Adaptive User Interface

Alexandre Adler Cunha de Freitas  [Federal University of Espírito Santo | alexandre.a.freitas@edu.ufes.br]

Simone Dornelas Costa  [Federal University of Espírito Santo | simone.costa@ufes.br]

Murilo Borghardt Scalser  [Federal University of Espírito Santo | muriloborghardt7@gmail.com]

Monalessa Perini Barcellos  [Federal University of Espírito Santo | monalessa@inf.ufes.br]

Abstract The new ways of interacting with computers, smartphones, and other devices have brought new challenges, such as the need to ensure that different types of users can easily use the same system. Adaptive User Interface (AUI) systems have been recognized as a solution to this matter. They change the user interface to better meet the needs of different users. However, developing such systems is not trivial. It is necessary to capture the users' characteristics and preferences and constantly adapt the system accordingly. In this paper, we discuss the use of ontologies to support the development of AUI systems. We argue that by providing structured knowledge about such systems, ontologies help understand how they work and offer a basis to structure them, identify the necessary adaptations and implement mechanisms to make them happen in run-time. We have explored the use of ontologies from an ontology network (i.e., networked ontologies) to develop a social network about academic subjects that automatically adapts its interface according to the low vision and colorblind user's needs and usage characteristics. This exploratory study showed that using networked ontologies to develop an AUI system is useful and feasible. The ontology was useful at the conceptual level by serving as a basis to define the system's structural model and at the operational level by providing the semantics used in a reasoning engine to adapt the UI at run-time. The first version of an ontology-based process to guide the development of AUI systems emerged from this experience and it is also presented in this paper.

Keywords: *Adaptive User Interface, Ontology, Ontology Network.*

1 Introduction

In the current technological landscape, the development of interactive systems that prioritize human needs and preferences has become increasingly important. Ensuring that such systems are designed with a human-centered approach is a critical issue (Gurcan et al., 2021). The digital and contemporaneous society has required interactive systems even more intuitive and suitable for user needs. For that, a proper user interface (UI) is needed. Efficient UIs promote effective communication between users and the system. Thus, a well-designed UI is essential for the success of any interactive system. However, the increasing number of different electronic devices, environments, and types of users has been a challenge when developing interactive systems (Sebek et al., 2015).

A well-known and currently discussed issue in this context concerns usability problems when different types of users use the same system (Rathnayake et al., 2019). Users differ in a wide range of variables, including demographic characteristics, background, education, personality, cognitive skills, and preferences. Users' motivation, goals, and moods also vary.

For a UI to facilitate effective communication between the user and the system, it should consider the different needs of different users. The difference can take into account characteristics, such as the ones aforementioned, accessibility needs, and others. Thus, it is important that the UI be able to adapt to the requirements of different types of users (Sebek et al., 2015), i.e., we need Adaptive User Interfaces (AUI). The purpose of an AUI is to enhance the user's experience with the UI by adapting various aspects of the UI based on the user current goals and needs (Machado et al., 2018). UI adap-

tations can be performed in an adaptive system and usually require identification and classification of user characteristics to constantly adapt the system (Firmenich et al., 2019). In this paper, we adopt the term *AUI system* referring to such systems, i.e., a system that adapts its UI.

Developing AUI systems is a complex and knowledge-intensive activity (Yigitbas et al., 2019). UI adaptations may occur according to a diversity of user information. Hence, structuring and organizing knowledge about the user and the system to promote the proper adaptations in the UI becomes necessary. In this work, we argue that using ontologies is a promising approach to aid in this matter. Ontologies capture and organize knowledge and, thus, can be used to structure knowledge about the interactive systems and the users' characteristics, helping understand how such systems work and serving as a basis to structure them, identify the necessary adaptations, and implement mechanisms to make them happen in run-time.

In the literature, some works have explored the use of ontologies to develop AUI systems (Costa et al., 2021) (e.g., (Bonacin et al., 2022), (Braham et al., 2021)), (Fedasyuk and Lutsyk, 2021), (Stefanidi et al., 2022), (Sala et al., 2021) and (Khan and Khusro, 2019). However, the ontologies often are very specific, i.e., they can only be used to solve a particular problem in the context of the system to which they were created, and are used mainly at the operational level. This may work for isolated solutions, but systems have been required to be more comprehensive and constantly evolve according to the user needs. Isolated solutions are usually hard to be extended to incorporate new requirements or reused in the development of new solutions.

Therefore, we advocate that ontologies should also be used at the conceptual level to structure knowledge about the system and user characteristics. Thus, it is possible to provide a general knowledge framework that can be used as a basis to define UI adaptations and develop AUI systems. We also argue that, ideally, we should use ontologies from an ontology network (ON), i.e., a set of interconnected ontologies that provide a comprehensive conceptualization of the domain of interest and have a common global conceptual structure that helps share their concepts (Sattar et al., 2021). By doing so, it is possible to constantly evolve the set of possible adaptations by considering different concepts from the networked ontologies.

We hypothesize that networked ontologies can help develop AUI systems at both conceptual and operational levels. Thus, we performed an exploratory study in which we explored the use of ontologies from an ON to aid in developing AUI systems to verify whether and how networked ontologies support developing an AUI system. Our research aims at identifying key steps for using networked ontologies in this context and providing guidance on how to develop ontology-based AUI systems. In the study, we used an extract of the Human-Computer Interaction Ontology Network (HCI-ON)¹ (Costa et al., 2020, 2022), to develop SNOPI (Social Network with Ontology-based adaptive Interface), a social network about academic subjects that automatically adapts its UI according to the low vision and colorblind user's needs. HCI-ON is an ON that contains several ontologies addressing HCI subdomains. The ontology (i.e., the ON extract) was useful at the conceptual level by serving as a basis to define the system structural model and at the operational level by providing the semantics used in a reasoning engine to adapt the UI at run-time. From this experience, the first version of an ontology-based process to guide the development of AUI systems emerged.

This paper is aimed to share the main results of our experience. It extends (Freitas et al., 2022) by improving the paper background, providing information about the research approach we have followed, introducing HCI-ON and its ontologies relevant to this work, describing further details about SNOPI and the use of ontologies to develop it, and presenting the main results of an interview performed with the SNOPI developer.

The paper is organized as follows. Section 2 provides the background for the paper and discusses some related work; Section 3 regards the research approach used in this work; Section 4 presents HCI-ON and its fragment relevant to this paper; Section 5 introduces SNOPI and discusses how we used networked ontologies to develop it; Section 6 presents the main results of an interview performed with the SNOPI developer to get his feedback about using an extract of HCI-ON to develop an AUI system; Section 7 describes the ontology-based process to develop AUI systems resulting from the exploratory study; and, finally, Section 9 concludes the paper.

2 Background

2.1 Adaptive User Interfaces

Interactive systems are systems designed with a focus on the user (Farooqui et al., 2021) and, thus they have a UI. Improving user experience and usability is a necessary concern when developing such systems. Interactive systems can also be adaptive systems, i.e., systems that have the ability to automatically adapt to changes in their environment or context of use (Yigitbas et al., 2020). It is possible to focus the adaptation specifically on the UI, resulting in an AUI. Adaptive systems can adjust their functionalities, resources, and behaviors to meet the needs of users or the environment in which they are being executed (Yigitbas et al., 2020). Thus, a UI is adaptive when it automatically adapts itself according to the user's characteristics.

Adaptable UI, which also refers to UI that undergoes adaptations, can be confused with AUI. However, they are distinct concepts. A UI is adaptable when the user can explicitly and deliberately adapt it. For example, users may be able to adjust the font size or color scheme of the interface to make it easier to read. The user explicitly requests the modifications they want to make, and the system responds by making the necessary changes. On the other hand, an adaptive UI is a user interface that can automatically adjust itself at run-time to changes in context (Yigitbas et al., 2019). The system collects information about the user's preferences, behavior, and context and uses this information to modify the interface accordingly. For example, an adaptive interface may automatically adjust the font size based on the user's visual acuity or change the layout of the interface based on the user's device screen size. In this paper, we focus on adaptive interfaces.

The need for adaptive interfaces arises when interactive systems become increasingly complex and effective, allowing different types of users to access them from various computing devices, such as computers, smartphones, tablets, and notebooks. At this point, systems are developed to meet the needs of the majority of its users, in order to improve the usability of the interface. However, due to the heterogeneity (i.e., a multitude of end users, computing platforms, input/output resources, interaction modes, and user work environments, among others), this mode of development can lead to the exclusion of certain types of users or user characteristics can change over time (Hussain et al., 2018). Developing an adaptive interface is a good strategy to reduce the impact of problems caused by heterogeneity, as it considers different characteristics of different users and shapes the interface to meet them, making the system more accessible and improving its usability.

Developing an AUI has some challenges, whether in collecting information from the user, measuring user characteristics, or organizing knowledge through the information collected. Also, there are issues related to the protection and privacy of the user's data, the user's adaptability to a new interface, and the difficulty in adapting the conceptual model of the UI to achieve the desired goals (Oppermann, 1994).

Another important challenge when developing an AUI is designing for the diversity of users and contexts of use, which implies making alternative design decisions at various levels

¹HCI-ON is available at <https://dev.nemo.inf.ufes.br/hcion/>

of interaction design. Furthermore, creating an AUI requires modeling information related to the user and the context of the interaction, which can be complex and require additional resources. However, research on AUIs continues to advance to provide highly usable systems for people with different needs and characteristics in different usage contexts (Gullà et al., 2015).

2.2 Ontologies

Gruber (1993) defines ontology as “a formal explicit, specification of a shared conceptualization”. The conceptualization is an abstract and simplified view of the world. In this definition, “conceptualization” refers to an abstract model of some phenomenon in the real world that identifies the relevant concepts of this phenomenon; “explicit” means that the types of concepts used and the constraints imposed on their use are explicitly defined; “formal” refers to the fact that an ontology should be interpretable by machines; and “shared” reflects that ontologies must capture consensual based knowledge accepted by a community.

In the literature, ontologies have been classified considering diverse perspectives, such as according to their levels of generality, formality, applicability, among others. Regarding Generality, ontologies can be classified into foundational, core, and domain generality levels (Scherp et al., 2011). Foundational ontologies aim at modeling the very basic and general concepts and relations that make up the world (including domain-independent notions, such as objects, events, participation and parthood) (Guarino, 1998). They are generic across any area and are highly reusable in different modeling scenarios (domain-independent) and represent the highest-level ontologies. Core ontologies provide a refinement to foundational ontologies by adding detailed concepts and relations in a specific area (such as service, process, organizational structure) that still spans across various (sub)domains. Core ontologies are situated in between foundational and domain ontologies and despite being more general than domain ontologies, are domain-dependent. Finally, domain ontologies describe knowledge that is specific to a particular domain, such as a soccer ontology (Guarino, 1998), and represent the lowest-level ontologies (e.g., an ontology about the anatomy of the human body). They can make use of/be based on foundational ontologies or core ontologies by specializing their concepts. Higher-level ontologies can be used to support the development of lower-level ontologies, i.e., foundational ontologies can be used as the basis for building core and domain ontologies, and core ontologies can support the development of domain ontologies.

Another important distinction concerns the ontology intended application and differentiates ontologies as conceptual models, called *reference ontologies*, from ontologies as computational artifacts, called *operational ontologies* (Guarino et al., 2009; Guizzardi, 2007). A reference ontology is as a special kind of information object that aims to make the best possible description of the domain in reality, regardless of its computational properties. Operational ontologies, in turn, are computational artifacts designed with the focus on guaranteeing desirable computational properties and, thus, are machine-readable ontologies.

Ontologies have been used in software development to aid in several contexts. Ontology-oriented software development can use both reference and operational ontologies. The former is suitable for supporting the description of the application domain itself and is applied in development time, a.k.a, *ontology-driven development* (ODD) (Seedorf et al., 2006). The latter is appropriate for use as primary artifacts in run-time and plays a major role in application logic, a.k.a, *ontology-based architecture* (OBA) (Seedorf et al., 2006).

For large and complex domains (such as HCI), ontologies should be organized in an *Ontology Network* (ON), where ontologies are modular and related together through a variety of relationships (e.g., modularization, alignment, dependency), sharing concepts and relations with other ontologies and, thus, forming a network of interlinked semantic resources. A *networked ontology* is an ontology that belongs to a network and has relationships with a potentially large number of other ontologies (Suárez-Figueroa et al., 2012).

Especially for a complex domain, representing its knowledge as a single ontology results in a large and monolithic ontology that is hard to manipulate, use, and maintain (Suárez-Figueroa et al., 2012). On the other hand, representing each sub-domain in isolation is a costly task that leads to a very fragmented solution that is again hard to handle (Ruy et al., 2016). In such cases, building an ON is an adequate solution (Suárez-Figueroa et al., 2012). ONs enable to establish a comprehensive conceptualization that provides a common understanding of the domain and can be used as a reference to solve semantic interoperability and knowledge problems related to the conceptualization as a whole or to extracts of it. Hence, integrating several ontologies into an ON provides a framework that can be explored to potentialize and increase the set of solutions in the universe of discourse addressed by the ON (Costa et al., 2020).

3 Methodological Approach

The work presented in this paper is part of a research project in which we have followed the Design Science Research (DSR) paradigm, which concerns extending human and organizational capabilities by creating new and innovative artifacts (Hevner, 2007; Hevner et al., 2008). DSR is an iterative process involving three cycles (Hevner, 2007): Relevance Cycle, Design Cycle and Rigor Cycle.

A DSR project begins with the Relevance Cycle, which involves defining the problem to be addressed, the project goal, the requirements to be met, and the criteria for evaluating the results. The problem addressed by this work refers to difficulties to develop AUI systems. The problem was identified in the literature (Norcio and Stanley, 1989; Yen and Acay, 2009; Akiki et al., 2014) - we performed informal and systematic literature reviews - and also based on the experience of the first author when developing such systems. Considering the identified problem, the perceived gaps and the benefits reported in the literature of using ontologies to address semantics and support software development, we decided to develop an ontology-based approach to support AUI systems development. The main requirements for the approach are (R1) it must guide its users on the steps to develop AUI systems,

and (R2) it must provide a knowledge framework related to AUI systems. In addition to the requirements to be met by the approach, as criteria to evaluate it, we defined that it must be useful and its use must be feasible.

The Design Cycle involves developing and evaluating artifacts or theories to solve the identified problem (Hevner, 2007). Therefore, in this cycle we aim to develop OADAPT (Ontology-based Approach to Develop Adaptive Interfaces), which is the main artifact produced in the project. OADAPT consists of two components: (i) a knowledge framework given by an ON containing ontologies that address aspects relevant to adaptive systems and AUI, and (ii) a process describing the steps to use it to develop AUI systems. The first component aims at meeting R2, while the second is aimed to satisfy R1.

The approach development includes design activities and some empirical studies. As suggested in (Barcellos et al., 2022), the studies have been organized as learning iterations - i.e., studies performed in iterations that allow the researcher to learn something about the research, providing useful knowledge to understand the problem, develop the artifact, evaluate or improve it. Until this moment, we have performed two learning iterations. The first one refers to a systematic mapping of the literature to investigate the use of ontologies to support AUI systems development. The results helped us to get a panorama about the state of the art of the research topic and better understand the problem. The second learning iteration, which is the focus of this paper, refers to an exploratory study to experience the use of networked ontologies to develop AUI systems and resulted in the first version of OADAPT. In addition to the two learning iterations already performed, we plan another three. The third learning iteration (ongoing) consists of a case study in which the first version of OADAPT (the one presented in this paper) is being used by a developer (different from the one that developed SNOPI) to evolve SNOPI. The fourth learning iteration (started) is a case study in which a different developer will build an AUI system from scratch. This studies will allow us to evaluate the use of OADAPT by third parties, identify the approach strengths and weaknesses and evolve it accordingly. Finally, the fifth learning iteration will be an experimental study to evaluate the new version of OADAPT and will involve other developers. In these three learning iterations, we will evaluate OADAPT considering its usefulness and feasibility, as defined in the Relevance Cycle.

The Rigor Cycle refers to using and generating knowledge during the work. Rigor is achieved by appropriately using foundations and methodologies from a knowledge base grounding the research, and adding knowledge generated by the research to contribute to the growing knowledge base (Hevner, 2007). As foundations, we have used relevant literature related to AUI systems, including standards, theories, models and others. We have also used foundations about ontologies and ontology networks, and about primary and secondary studies. Our main contribution for the knowledge base is OADAPT itself.

Next, we introduce the knowledge framework of OADAPT, which consists of networked ontologies of HCI-ON (Costa et al., 2020, 2022).

4 The Human-Computer Interaction Ontology Network (HCI-ON)

The Human-Computer Interaction Ontology Network (HCI-ON) is a knowledge framework of HCI that provides a general and solution-independent conceptualization resulting from an intensive HCI domain analysis (Costa et al., 2020, 2022; Costa, 2022). Figure 1 presents an overview of its current version. In the figure, each circle represents an ontology. Dotted circles represent HCI-ON ontologies under development. Arrows denote dependency relationship between networked ontologies. Dependency relationship indicates that concepts from the target ontology are reused by the source ontology.

HCI-ON addresses HCI core aspects and sub-domains, adopting an architecture that promotes knowledge organization. HCI-ON adopts a three-layered architecture by following the classification proposed in (Scherp et al., 2011). In the background, we have a foundational ontology (the Unified Foundational Ontology – UFO (Guizzardi, 2005; Guizzardi et al., 2008, 2013)) to provide the general ground knowledge for classifying concepts and relations in the ON². In the center, core ontologies are used to represent the general domain knowledge, being the basis for the sub-domain networked ontologies. Last, domain-specific ontologies appear, describing more specific knowledge.

Currently, HCI-ON includes ten ontologies and more than 100 concepts. At the core layer, there is the *Human-Computer Interaction Ontology* (HCIO), which addresses what an interactive computer system is, user actions taken in an interaction, and how an interaction happens (Costa et al., 2022). At the domain layer, there are nine ontologies, namely: *User Characterization Ontology* (UCO), which concerns aspects related to user characteristics; *UI Types and Elements Ontology* (UIT&EO), which addresses UI types and its components; *Adaptive Interface Ontology* (AIO), which deals with AUI, its components and customizations; *User Profile Ontology* (UPO), which addresses general aspects of the user profile and useful information for managing and improving interface adaptations; *Context of Use Ontology* (CUO), which involves concepts describing the elements that characterize a context of use, such as physical and social environments wherein the interaction occurs; *HCI Modality Ontology* (HCIMO), which treats HCI styles/paradigms (modalities of interaction); *HCI Quality Characteristics Ontology* (HCIQCO), which concerns interactive computer system quality characteristics (e.g., usability, communicability); *HCI Design Ontology* (HCIDO), which involves concepts related to HCI design and design components (Costa et al., 2020; Castro, 2021); and *HCI Evaluation Ontology* (HCIEO), which regards concepts related to HCI evaluation and measurement.

HCI-ON allows the use of the complete framework or extracts of it. Since the concepts are integrated in a consistent way, one can just select the fragment that reflects the domain of interest (Costa et al., 2020; Costa, 2022).

In the context of the work addressed in this paper, we have used concepts from HCIO (core ontology) (Costa et al.,

²Discussions about UFO and its use to ground HCI-ON core and domain ontologies are out of the scope of this paper. Information about that can be found in (Costa et al., 2020, 2022; Costa, 2022).

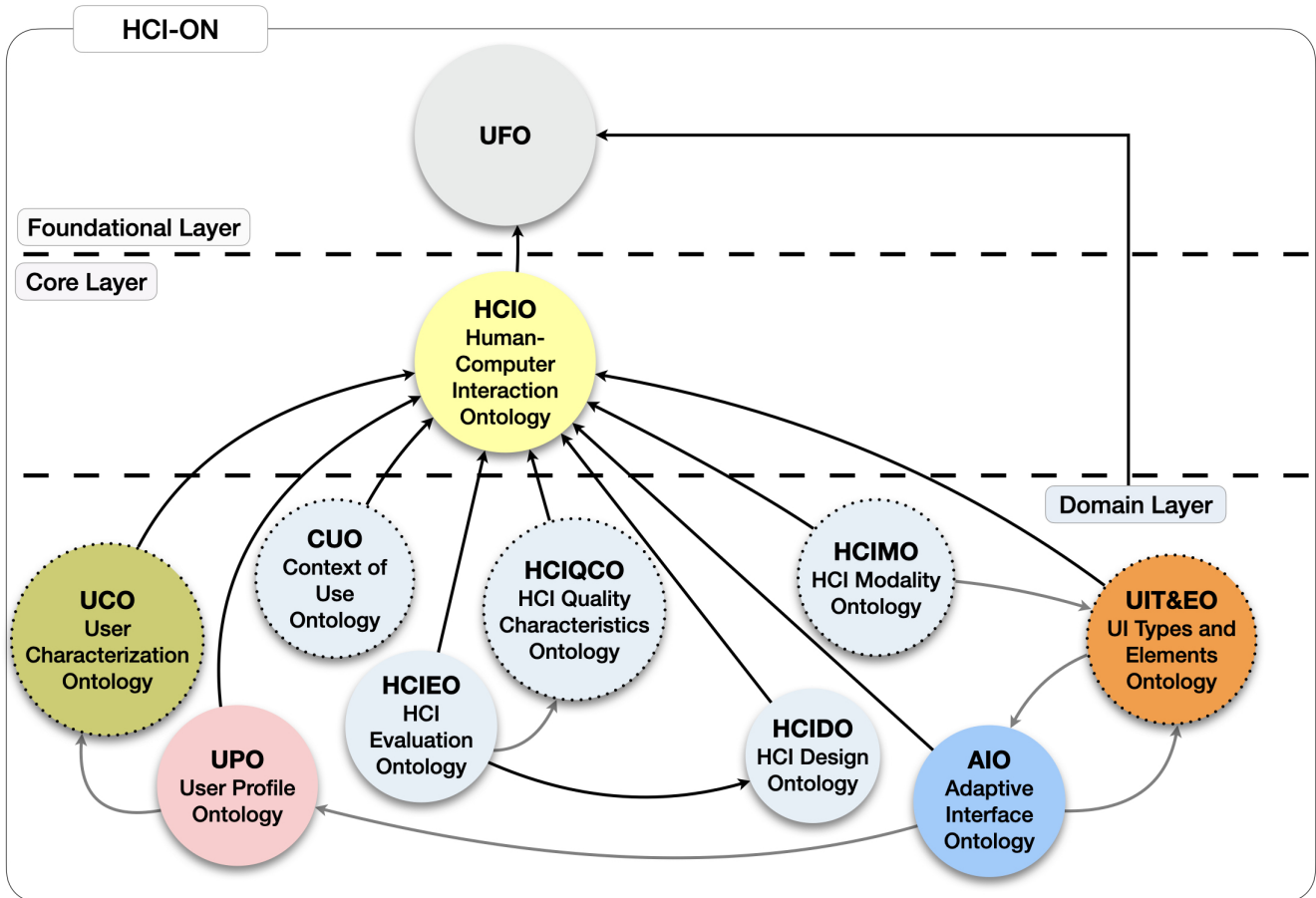


Figure 1. HCI-ON current version.

2022) and developed/evolved UCO, UIT&EO, AOI and UPO. Figure 2 shows a fragment of the HCI-ON extract used to develop SNOPI (blue circles highlight the concepts directly used in this paper). In the figure, concepts are represented with the same colors used in the ontologies shown in Figure 1 (e.g., yellow concepts are HCIO concepts).

Briefly, an *Interactive Computer System* is a computer system that has a *User Interface* composed of *Output Equipment* and *Input Equipment*. An *Adaptive User Interface*, in turn, is a *User Interface* that adapts itself and, thus, it is part of an *Adaptive Interactive Computer System*. *User* participates in *Human-Computer Interactions* to communicate with the system. *User* has *User Characteristics* (e.g., Mary - a user - has 18 years) that are considered to define the *User Profile*, which is composed of several properties (e.g., *User Gender*, *User Disability*, *User Age*, *User Education*, *User Language*, *User Experience Level*, among others). Each property can be specialized into others (e.g., *Cognitive Disability*, *Motor Disability*, *Auditory Disability* and *Vision Disability* are different types of *User Disability*). *User Profiles* are defined by *Adaptive Interactive Computer Systems*. A *User Interface Customization* refers to an adaptation to be made in the *Adaptive User Interface* based on a *User Profile* (the minimum cardinality of this relation is 0 because a customization can be based on other factors such as use context – not represented in Figure 2). A *User Interface Customization* is performed by an *Interface Component Program* (which is a program that composes the *Adaptive Interactive Computer System*) and changes the *Adaptive User Interface*.

5 Using Networked Ontologies to Develop an AUI System

We performed an exploratory study with the purpose of verifying whether the use of an ontology extracted from an ON would support (and how) the development of an AUI system. Since the beginning of the COVID pandemic, many scientific conferences have been held online, making it possible for many people to participate. Information is usually spread in many channels and people can miss it. Thus, we decided to develop a social network devoted to information about scientific events (dates, call for papers, links, publications, etc.). The system, called SNOPI (Social Network with Ontology-based adaptive Interface), should present the basic features of a social network (e.g., feed, add comments, upvote, downvote) and cover specificities of the application domain. We defined some Personas (Salminen et al., 2020) to identify the main user profiles and needs. In this context, we included Personas with some special needs related to low vision and colorblind.

Once we identified the system requirements, users and their characteristics, we selected an extract of HCI-ON covering aspects related to interactive system, UI, user characteristics, user profile and adaptations (a fragment is shown in Figure 2).

SNOPI³ development followed ODD and OBA approaches. The ontology was used as a reference conceptual model at development time to structure the system and its relational

³SNOPI is available at <https://dev.nemo.inf.ufes.br/snopi/>

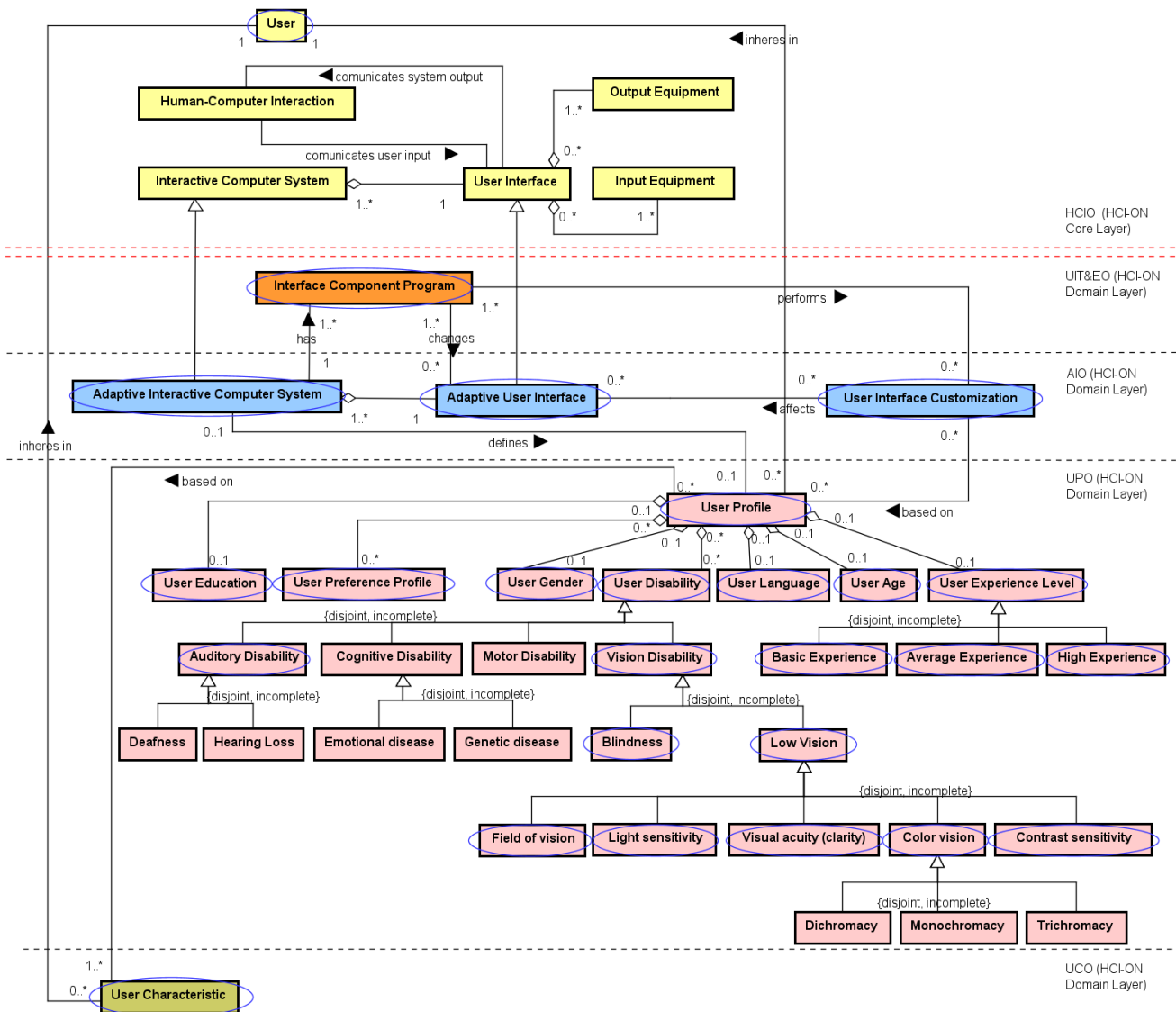


Figure 2. Fragment of the HCI-ON extract used to develop SNOPI.

database. The reference ontology also served as a knowledge body that helped better understand AUI systems and related concepts. Moreover, it was transcribed into a language that allows machine-reading, resulting in an operational ontology (ontoSNOPI)⁴ used at run-time to support UI adaptation through reasoning.

ontoSNOPI was implemented using OWL (Ontology Web Language)⁵. In addition to the ontology concepts and relationships, ontoSNOPI implements the adaptation rules by means of axioms, on how to adapt the UI according to the user’s profile (e.g., considering degrees of color blindness and low vision). The axioms were defined considering the recommendations of the W3C⁶ Accessibility Standard and the Web Content Accessibility Guidelines (WCAG)⁷.

Shortly, we used the reference ontology to identify the different properties related to *Vision Disabilities* to be considered in the system (i.e., *Blindness* and five types of *Low*

Vision, namely: *Field of Vision*, *Light Sensitivity*, *Visual Acuity (clarity)*, *Color Vision*, *Contrast Sensitivity*) and the aforementioned standards to help identify the adequate adaptations. For example, for *Visual Acuity (clarity)* user needs, there is a recommendation related to Perception that makes changes in text size, font, style, capitalization, and all interface elements. For *Contrast Sensitivity* user needs, there is a recommendation related to Brightness and Color that allows users to set background and text colors from a full-color spectrum. After defining the adaptation rules (i.e., *User Interface Customizations*), by following the ontology conceptualization, we created *Interface Component Programs* that implement such rules and materialize the adaptation in the UI.

Figure 3 presents an overview of SNOPI. The UI layer contains the components (e.g., screens, controllers) that communicate user and system. The application layer regards the system functionalities, while the data layer concerns data structure and storage. Finally, the semantic layer concerns a reasoning engine that uses the operational ontology and adaptation rules to change the interface according to the user characteristics.

⁴ ontoSNOPI is available at <https://dev.nemo.inf.ufes.br/hcion/ontoSNOPI.owl>

⁵ <https://www.w3.org/TR/owl-guide/>

⁶ <https://www.w3.org/standards/webdesign/accessibility>

⁷ <https://www.w3.org/TR/WCAG/>

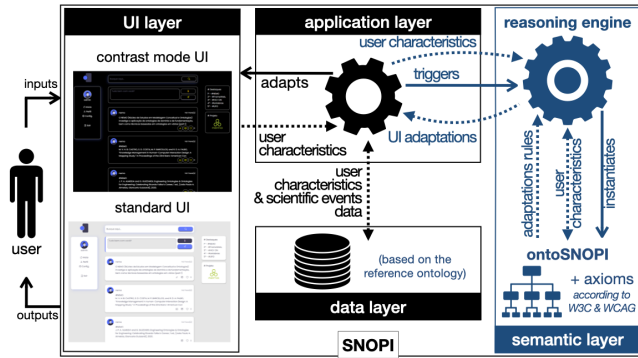


Figure 3. SNOPI overview.

When accessing the system, the user is asked to answer a questionnaire (a partial view is illustrated in Figure 4). The questionnaire was designed around common parameters noted in the W3C Recommendations on Web Accessibility Initiative⁸. The purpose is to capture some user characteristics to support UI adaptation. User data is stored in the database. At run-time, this data is instantiated in the ontoSNOPI and inferences (reasoning) using the axioms are performed from these instances to identify the adaptations suitable for the user profile. The UI adaptations are recorded in the database and the system adapts the UI accordingly. This procedure is performed in the first time the user accesses the system because it does not know the user yet. In the next accesses, the system checks the captured user information and adapts the UI. At any time, the user can update personal characteristics in the system. Moreover, they can also make new changes in the UI by themselves (the UI is adaptive and adaptable).

Figure 4. Fragment of the User Characterization questionnaire.

Figure 5 illustrates the adaptation of the UI to the dark mode (low light emission) and increased font size. This adaptation is suitable for colorblind or light-sensitive user and users with impaired vision. SNOPI performs other UI adaptations, such as to the high contrast mode, which increases contrast, decreases visual interference, and increases the focus on the components, being suitable for contrast sensitive users.

In the following sections, we provide further information about the use of an HCI-ON fragment to develop SNOPI. Section 5.1 concerns the use of the ontology at run-time to identify and apply the UI adaptations. Section 5.2, in turn, discusses the use of the ontology at development time and presents information about SNOPI development.

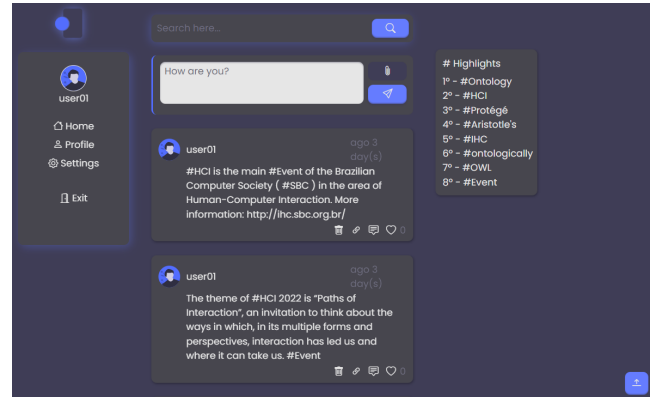


Figure 5. Feed UI adapted for dark mode and large font size.

5.1 Behind SNOPI UI Adaptations

As explained before, adaptations in the SNOPI UI occur in run-time (OBA approach), using ontoSNOPI, the operational version of the used HCI-ON fragment. To develop ontoSNOPI we used Protégé⁹, an open-source tool for ontology handling, developed and maintained by Stanford University (Musen, 2015).

Each concept from the reference ontology was represented in the operational ontology. Figure 6 illustrates ontoSNOPI class hierarchy defined in Protégé. In some cases, names were changed for simplification or implementation reasons. In this section, we use *bold italics* to refer to concepts from the operational ontology (Figure 6) and *italics* to refer to concepts from the reference ontology (Figure 2).

At first, we created two classes, *Customization* and *User_Profile*, as subclasses of the native class *owl:Thing*, referring respectively to *User Interface Customization* and *User Profile* concepts. *User_Profile* specializes classes referring to the concepts that inherit from *User Profile* in the reference ontology, dealing, thus, with user characteristics, difficulties, and general information. *Customization*, in turn, encapsulates the UI adaptation rules. For example, *Font_Increase* refers to the rule that is applied if the user profile has a disability in some field of vision (*Field_Of_Vision*) or visual acuity (*Visual_Acuity*) or has age (*User_Age*) greater than or equal to 40. The rules are implemented as axioms in the operational ontology and are generated based on user profile information and ontology concepts. In SNOPI, the rules are ‘fixed’ in the sense that they are predefined considering a set of user characteristics and possible adaptations, and they do not change during the system execution. Different rules will be applied to different user characteristics. New rules can be created by the developer considering other ontology concepts, user characteristics, or UI adaptation guidelines.

Using the defined classes, it was possible to capture and handle the necessary information to adapt the UI. For example, the *User_Disability* class captures whether the user has any disabilities. After creating the operational ontology, we exported the OWL file, which is used at run-time by the SNOPI back-end. To enable reasoning, we created an algorithm that loads the OWL file in the SNOPI back-end (the reasoning is performed on the server).

After the operational ontology is created, it can receive information from the users and perform inferences to identify

⁸<https://www.w3.org/WAI/>

⁹<https://protege.stanford.edu/>

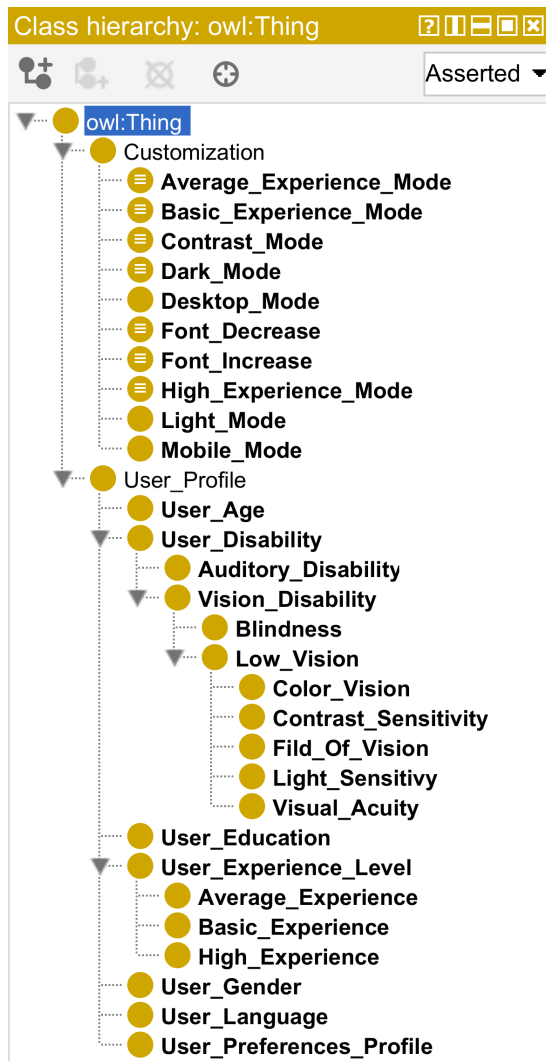


Figure 6. ontoSNOPI class hierarchy.

the UI adaptations to be applied. Information provided by the user (characteristics, preferences) is loaded (instantiated) in the ontology. For example, John (*User_Profile*) is a user who has little experience (*Basic_Experience*) with technology and has color blindness (*Color_Vision*). When a new user logs into the system, or when an existing user changes its data on the profile settings page, the system captures this information and stores it in the back-end, thus creating an individual on ontoSNOPI to represent the user.

After the creation of the individual in an instance of ontoSNOPI, Hermit Reasoner (Glimm et al., 2014)¹⁰ is called by the algorithm to perform inferences and identify the changes to be made according to the user profile. The data related to the adaptations applied are persisted in the system to adapt the user interface every time the user uses the system.

Hermit Reasoner is a widely used OWL 2 reasoner that can be used for performing reasoning tasks on ontologies. Thus, Hermit Reasoner can be used not only for ontology development but also for validating the rules of an operational ontology. For example, after creating an individual in an instance of ontoSNOPI, Hermit Reasoner can be used to validate the rules and ensure that they are consistent with the user's profile. Figure 7 shows an example of the user "01" characteristics, that has 62 years old and is sensitive to light.

¹⁰<http://www.hermit-reasoner.com/>

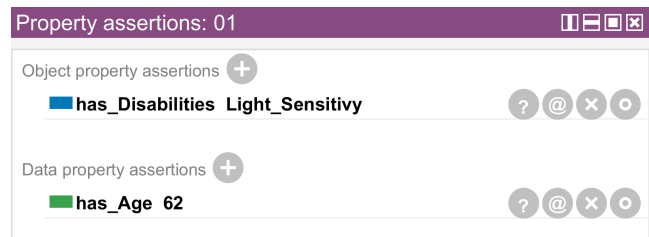


Figure 7. User "01" Properties.

Based on this information, Hermit Reasoner is used to perform run-time inference using the rules defined in the operational ontology and identify the most appropriate UI adaptations. For example, Figure 8 shows the most appropriate UI adaptations indicated for user "01" after inference. *Dark_Mode* indicates a UI adaptation due to the user's sensitivity to light while *Font_Increase* indicates a UI adaptation in which the font of characters is increased to improve readability because of the age of the user. To validate the rules, we ran tests considering different user characteristics. The obtained results were consistent with the accessibility guidelines considered.



Figure 8. Inference About the User "01".

5.2 SNOPI Development

In this section, we present information about SNOPI conceptual modeling, architecture and user interface.

5.2.1 Conceptual Modeling

The class diagram represents the initial model of the system, as it defines entities, attributes, and relationships (Gorman and Choobineh, 1990). The ontology (i.e., HCI-ON extract) was used in development time (ODD approach) as a source of knowledge and a basis for SNOPI class diagram. For building the class diagram, we considered the ontology conceptualization and represented it in a way closer to the system. Moreover, we added elements to address the main actions performed in a social network (creating posts, liking posts, and commenting on posts). Figure 9 depicts a fragment of SNOPI class diagram. In the diagram, we highlight the classes defined based on the ontology (by using the same colors used in Figure 2) and indicate the ontology concepts by using red font. In this section, underlined words refer to concepts from the class diagram (Figure 9) and words in *italics* refer to concepts from the reference ontology (Figure 2).

Profile class is based on the *User_Profile* concept of the ontology. It stores information about the user characteristics, which are recorded in the User class. The Profile class is related to: Gender (user's gender), StatusProfile (new: user who has just registered to the system; current: user who uses the system regularly and is familiar with it; legacy: user who has

not used the system for a long time), ExperienceLevel (user's experience with technologies), Language (user's language), and Education (user's level of education). The Preference class stores user preferences regarding the system interface and defines how the interface should be for this user. The Preference class was defined based on the *User Interface Customization* concept and refers to the customizations (i.e., adaptations) that must be made in the system interface for a given user, considering the inferences made using the operational ontology. It is related to: DarkMode (enables or disables dark mode), ExperienceLevelMode (user's level of experience with technology) and FontMode (to increase or decrease font size). The Sharing class is intended to store important information about Post sharing. Finally, the Post class stores the messages posted on the social network and the relationships between different messages, which materializes the interactions between users of the system. Post is related to PostType, which indicates if it is a video, a PDF file or others.

5.2.2 System Architecture

SNOPI was developed following the client-server paradigm and, thus, it was divided into two subsystems: one performed on the *Client* side and the other on the *Server* side.

Figure 10 illustrated SNOPI architecture. The *back-end*, (Server side) was developed using the Framework Spring, which is based on the *Spring Boot* standard. It contains the *Controller*, *Service*, and *Model* layers plus the operational ontology *ontoSNOPI*. The *Controller* layer is responsible for dealing with the *Requests* made by the *Server* side, authorizing and applying a preview of the business logic to validate the *Requests*. The *Web Server* is responsible for processing the HTTP requests received by the clients and returning the corresponding responses. The *Service* layer executes the *ontoSNOPI* OWL file to the *Back-End*, identifying the appropriate interface adaptations for each user. The *Model* layer deals with data persistence in the *Database*.

The *Front-End* (*Client* side), was developed using the *Angular Framework*, based on the MVVM¹¹ (Model-View-ViewModel) standard. It comprises the user interface layer (*Template*, *Style*, and *Script*), whose function is to deal with the components responsible for content display and is directly related to the *View Model* of the MVVM standard (Microsoft, 2021).

5.2.3 Adaptive User Interface Development

We started the AUI development by prototyping some screens using Figma¹² to better understand the application domain (social network) and, thus, explore possible adaptations. First, we developed the sketch¹³ of the application. Then, we created mock-ups¹⁴ to define the application design before im-

plementation. Figure 11 shows a mock-up produced during prototyping. All colors used in the UI were tested and approved by the online accessibility tool Adobe Color¹⁵, so the system, since its initial stage, used appropriate colors for people with some degree of color blindness. After creating the mock-ups, we used Angular to develop the UI.

6 Developer's Perception of Using Ontologies

Aiming to get feedback about the use of an HCI-ON extract to develop an AUI system, we carried out an interview with the SNOPI developer. We must clarify that, although all the authors have participated in SNOPI conception, its development was performed by only one developer. Next, we explain the study and summarize its main results obtained from the participant's answers and comments.

6.1 Study Planning

The interview goal was to investigate, from the developer's point of view, whether the use of networked ontologies (particularly an extract of HCI-ON) helps the development of an AUI system. Aligned with this goal, we defined two main questions: **(Q1)** *How does the use of networked ontologies help in the development of an AUI system?* and **(Q2)** *What are the benefits and difficulties of using networked ontologies in the development of an AUI system?*

The **instruments** used in the study consisted of: (i) a consent form to participate in the study, which aims to safeguard the participant's rights regarding the study and its results; (ii) a form to characterize the participant's profile, and (iii) an interview questionnaire used by the interviewers to guide the interview. The used instruments are available in (Freitas et al., 2023).

The **procedure** adopted in the study consisted of a face-to-face approach and semi-structured interview. In the face-to-face approach an interviewer asks the questions in the presence of the respondent (Robson and McCartan, 2016). In the semi-structured interview, the interviewer has an interview guide that serves as a checklist of topics to be covered and the order of the questions. Based on the flow of the interview, the order can be substantially modified and additional unplanned questions can be asked to follow up on what the interviewee says (Robson and McCartan, 2016).

In the interview questionnaire, Q1 and Q2 were detailed in more specific questions. They are listed below. For simplification, in the questions we adopted the term ontology referring to the HCI-ON extract (i.e., the networked ontologies) used to develop SNOPI.

- Q1.1. In which stages of SNOPI development (e.g., analysis, design, implementation) was the ontology useful? Why?
- Q1.2. In which stages the ontology was not helpful? Why?

¹¹<https://learn.microsoft.com/en-us/xamarin/xamarin-forms/enterprise-application-patterns/mvvm>

¹²<https://www.figma.com/>

¹³Sketch is popular among software UI/UX designers "to create either wireframes or pixel-perfect static images of GUIs that comprise mock-up artifacts" (Moran et al., 2020).

¹⁴A mock-up is a closer representation of the software to be developed (Acuña et al., 2012).

¹⁵<https://color.adobe.com/pt/create/color-accessibility>

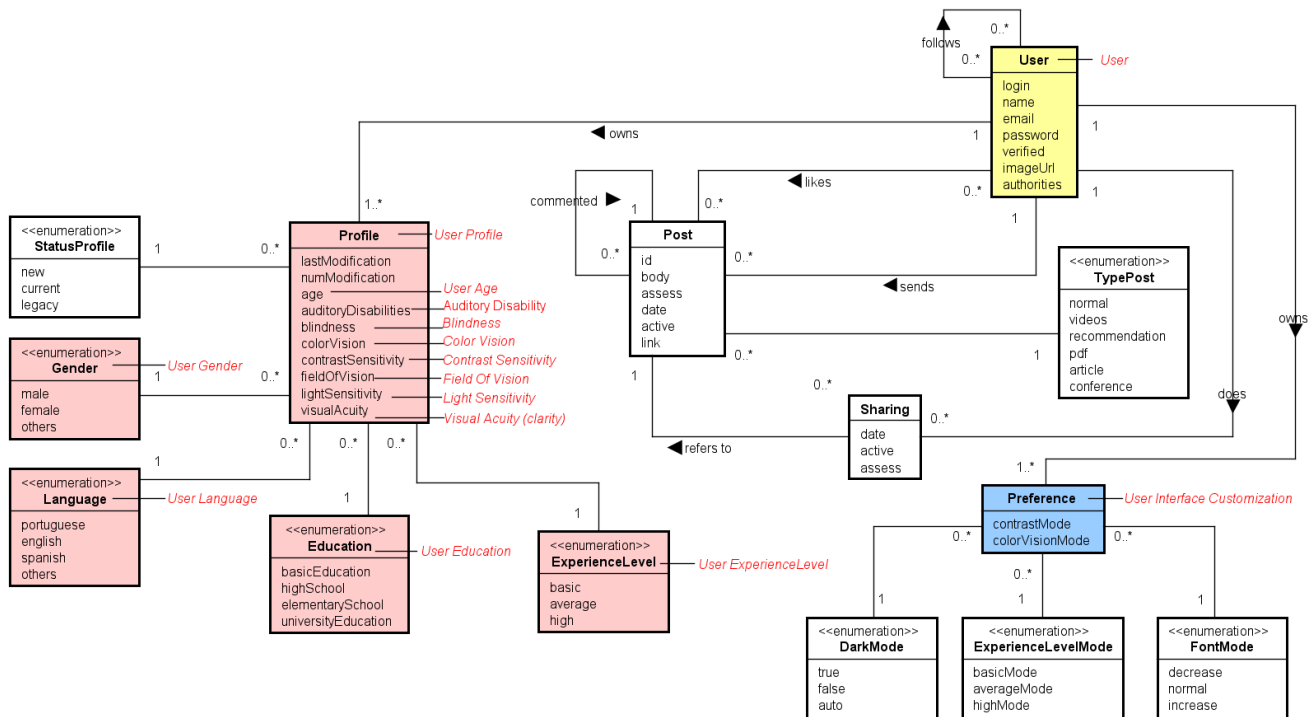


Figure 9. SNOPI Class Diagram.

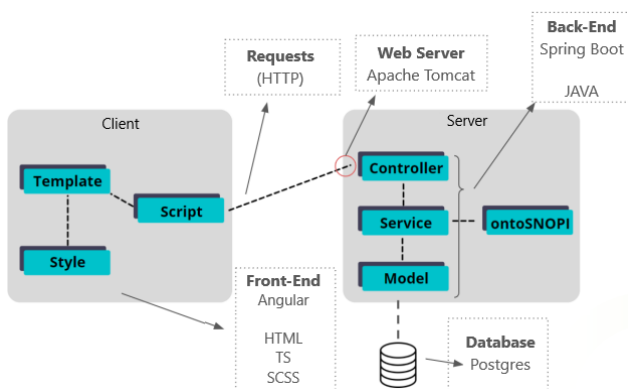


Figure 10. SNOPI architecture.

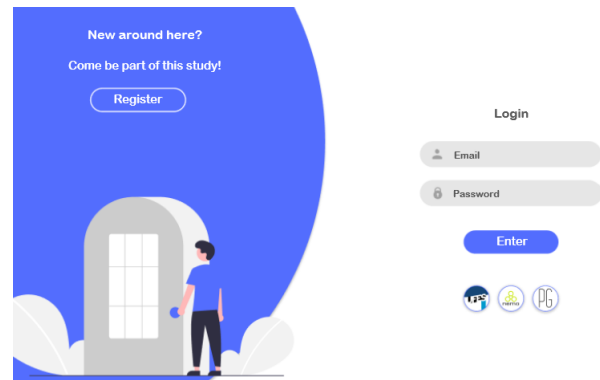


Figure 11. SNOPI Login Screen Mock-up.

- Q1.3. Do you consider that the ontology extract helped you to have a better understanding of the domain?
- Q2.1. What benefits have you noticed when using the ontology in SNOPI development?
- Q2.2. What difficulties have you faced when using the ontology in SNOPI development?
- Q2.3. Was this the first time you developed an ontology-based system? Briefly describe your experience.
- Q2.4. Would you use ontologies again to develop another system? Why?
- Q2.5. How was the SNOPI development process (e.g., you perform it based on your tacit knowledge and experience, you defined a process beforehand or did you follow an existing one, etc.)?

6.2 Study Execution and Results

The participant in this study was the SNOPI developer, who has an undergraduate degree in Computer Science and declared to have high theoretical and practical knowledge of systems development, medium theoretical knowledge of

ontology-based system development, medium theoretical knowledge of ontologies, and low practical experience with ontologies.

The interview was carried out by two interviewers (first and second authors). During the interview, the interviewers followed the questionnaire. With the consent of the interviewee, the interview was recorded.

Following the planned procedure, the interview was conducted face-to-face. After presenting the purpose of the interview, the interviewers started the interview following the order of the questions in the questionnaire. During the interview, the interviewers rephrased some questions, asked new ones, presented examples and clarifications to improve the interviewee’s understanding and collect feedback. The interview lasted approximately one hour. Next, we summarize the main answers given by the participant.

When asked about which steps of SNOPI development the ontology was most useful, the participant reported that it was in the Requirements Specification and Analysis and Design steps (Q1.1). According to him, the ontology did not help choose the technologies to be used (Q1.2). He stated that

the ontology helped him understand the interactive system and AUI domains but did not help him have a better understanding of the application domain because the application was about social networks and the ontology addresses AUI systems aspects (Q1.3). Regarding the benefits of using the ontology in SNOPI development, he reported that ontology concepts related to adaptive interfaces were primarily used for system modeling, class diagram development, and database modeling (Q2.1). Regarding the difficulties, he said that the greatest challenge was building the operational ontology and figuring out its logical operation (reasoning) (Q2.2). Because of that, he said that a tool to transform reference ontologies (i.e., conceptual models) into operational ontologies would be very useful, as well as guidelines to help operational ontology development. He informed that SNOPI development was the first time he used an ontology to develop a system (Q2.3) and that he would use ontologies again (Q2.4). Regarding the SNOPI development process, he said that he used his knowledge and followed the classic stages of the software development process (Q2.5). He pointed out that a process with well-defined steps and guidelines would be helpful because he had doubts about how to perform some steps and make the most of the use of the ontology.

6.3 Discussion

In this section, we briefly discuss the results considering the main questions of the study (Q1 and Q2).

Concerning Q1, the results indicate that the use of the ontology was most useful in Requirements Specification & Analysis and Design stages. The reference ontology supported understanding the interactive system and AUI domain, which helped define how the system should work. In addition, it aided in identifying user properties that could be considered by the system to define different profiles and served as a basis for the system structural model (the ontology model was used to create the class diagram and define the database structure). Moreover, the operational ontology helped capture user characteristics and change the UI at run-time.

Regarding Q2, according to the developer's feedback, ontology concepts helped him with system modeling, class diagram development, and database modeling. Another advantage addressed by the author of using networked ontologies is that one does not need to look for a particular ontology able to address the application problem or develop a new one with that purpose. They can just look at the ON and identify the extract that fits their need. Concerning difficulties, despite the ontology usage both at development time (ODD) and at run-time (OBA), the developer pointed out that in ODD, he had difficulty transforming the reference ontology into the operational ontology and suggested the creation of guidelines to support this task. Furthermore, the developer felt lost in which software development stages he would use the ontology. Despite following the classic software development process, he believes it is not sufficient when software development is ontology based.

In conclusion, the results of the interview indicated that the use of networked ontologies is useful and feasible. However, there are still some challenges to overcome, particularly related to the lack of knowledge and guidelines on the steps

to be followed and how to perform them. Considering that, based on the SNOPI development experience, we structured the acquired knowledge and lessons learned (i.e., tacit knowledge) and defined a process (i.e., explicit knowledge) that contains the steps to use ontologies from an ON to develop an AUI system. The process is presented in Section 7.

6.4 Limitations

Any study has limitations that must be considered together with the results. The main limitation regards the participation of the authors in SNOPI development. The SNOPI developer carried out SNOPI development under the supervision of some of the authors, which may have influenced his perception of using HCI-ON and his answers in the interview. To minimize the influence of the relationship between the interviewee and interviewers during the interview, the interviewer followed some recommended procedures: he listened more than spoke; posed questions in a straightforward, clear, and non-threatening way; and tried to get the interviewee to talk freely and openly. Even so, it is not possible to eliminate biases.

Some limitations related to the interview in general also apply to this study. First, the interview is time-consuming. This can tire the interviewee, influencing their answers. The interview was very straightforward and lasted about one hour. It was recorded, so the interviewers did not have to waste time writing down the responses. Second, the participant may have misunderstood some questions. To avoid this, the interviewer exemplified some questions and/or reformulated the questions to facilitate understanding. Last, some questions can lead to confirmation bias. In such cases, the interviewer asked the participant to justify his answers. We also highlight the limitation of the interview has been conducted by only one interviewer. Besides the individual bias, the absence of other people to help raise new questions or comments limits the possibility of getting new information that was not considered in the interview planning.

We must also clarify that the SNOPI developer is one of this paper's authors, which is also a threat. However, he joined the research project exclusively to develop SNOPI (i.e., he did not participate in the project before that and he left the project just after concluding SNOPI development), which minimizes the threat, although not eliminate it. It is also important to consider that the HCI-ON extract used in the study was developed by some of the paper authors. Thus, they have knowledge of the ontologies, which helped clarify doubts during SNOPI developing. This may have influenced the developer's perception of using networked ontologies. Thus, other developers may have perceptions different from his.

The fact that we have got feedback from only one developer, who is a beginner in ontology-oriented software development, is also an important limitation. Relying on feedback from a single developer does not provide a complete picture of the proposal's effectiveness.

Therefore, considering the study limitations, the results are not conclusive, cannot be generalized, and should be considered as preliminary evidence that the use of networked ontology helps develop AUI systems is useful and feasible.

7 An Ontology-based Process to Develop AUI Systems

In this work, we argue that using an ON makes it easier to apply ontologies to develop AUI systems because an ON provides a comprehensive and consistent conceptualization and, thus, the ontology can be extracted from the ON instead of being developed from scratch. As a result, the effort to produce the used ontology decreases. Moreover, the quality of the used ontology increases because ontologies integrated into the ON are properly verified and validated (Ruy et al., 2016). Thus, the ON is a key pillar in our approach.

In the last years, we have worked on HCI-ON (Costa et al., 2020, 2022), an ON that addresses several HCI subdomains. In this context, recently, we have dedicated efforts to develop ontologies devoted to adaptive interactive computer systems and AUI (a fragment was shown in Figure 2). The ON serves as a knowledge framework that can be used to solve knowledge-related and interoperability problems (Costa et al., 2020, 2022). In this work, the knowledge-related problem refers to understanding interactive computer systems and AUI subdomains and applying the conceptualization to develop AUI systems. Besides providing knowledge about these subdomains, it is also needed to provide guidelines on how to use the ON to develop AUI systems. Thus, our approach, which we call OADAPT (Ontology-based Approach to Develop Adaptive Interfaces), consists of a knowledge framework (i.e., an ON containing ontologies that address relevant aspects to adaptive systems and AUI) and a process describing the steps to use it to develop AUI systems. The process emerged from the SNOPI development experience and considers the use of ontologies at development time (ODD) and run-time (OBA). Since it is a software development process, it includes some classic software development phases (steps (i), (ii), (iv), (v), and (viii)). In addition to them, the OADAPT process includes three steps (steps (iii), (vi), and (vii)) that were defined based on the experience gained during the SNOPI development study. These additional steps were designed to address the specific challenges of developing software systems with adaptive user interfaces using ontologies. By including these steps in the OADAPT process, we aim at developing AUI based on a solid foundation of ontological knowledge. The proposed process is the first version of the process component of OADAPT (Ontology-based Approach to Develop Adaptive Interfaces). Figure 12 shows an overview of the eight-step process to use an ON to develop AUI systems. Next, we briefly describe each step.

(i) *Identify System Scope and Users*: This step consists in delimiting the system scope, identifying high-level requirements, the system users and its characteristics. It focuses on understanding the system purpose and boundaries, the problem to be addressed and the expected different types of users. Knowing the user's characteristics will help define the adaptations needed in the AUI in next steps. The main user's needs should also be identified to establish an initial set of functionalities based on high-level requirements. The adoption of techniques such as Empathy Map (Gray et al., 2010) and Personas (Salminen et al., 2020) is helpful to identify the users, their needs and characteristics. Empathy maps help

understand user behavior by visualizing their thoughts, feelings, and actions (Vasilieva, 2018). Personas, in turn, aid in creating fictional characters that represent different user types (Pruitt and Grudin, 2003). They are created to gain a better understanding of the needs, desires, and goals of a specific group of users; this information is then used to make decisions about the features of a system, service, or product (Salminen et al., 2020). By using these techniques, developers can obtain a deeper understanding of the user's needs and characteristics and create a system that meets them.

(ii) *Elicit System Requirements*: In this step, the results of the previous one are refined by defining the system functional and non-functional requirements (Sommerville, 2020). Functional requirements describe the functions the system should contain. Non-functional requirements, in turn, represent constraints that the system should address. In this sense, non-functional requirements are particularly important to indicate UI adaptation needs. For example, if the system users need UI accessibility options, non-functional requirements specifying such needs should be defined to be further addressed in the UI. Low-fidelity prototypes can be useful to capture information about functionalities and non-functional requirements. They provide a basic representation of the system's design, making it easier for users to give feedback about the system's scope and express their needs. Moreover, they allow developers to identify potential issues and make necessary adjustments before moving forward with the development process. By refining the system's requirements, developers can ensure that the system will meet the user's needs and expectations.

(iii) *Select Ontology*: In this step, considering the system scope, requirements, and user characteristics, the ON fragment (i.e., the reference ontology) to be used must be selected. The ontology provides a shared vocabulary and a set of rules that will be used to define and communicate the meaning of terms and concepts. Moreover, it will serve as a basis for conceptual modeling and reasoning. If necessary, new concepts can be added to the selected fragment. Once the reference ontology has been selected or extended, it can be used to refine the results produced in steps (i) and (ii). For example, the ontology can reveal new user's characteristics to be considered and can help refine non-functional requirements.

(iv) *Perform System Analysis*: Comprises developing the system structural and behavioral models. The structural model represents the system's static structure, while the behavioral model represents the system's dynamic behavior. Together, these models provide a comprehensive view of the system and its components. The reference ontology is used as a basis for the structural model (e.g., class diagram). If necessary, the model can be adjusted to be more suitable for the system (e.g., some concepts can be turned into attributes of other concepts and new concepts related to the application domain and not covered by the ontology can be added). Guidelines on how to turn ontological models into information models (more adequate to structure systems) can be found in (Carraretto and Almeida, 2012).

(v) *Define System Architecture*: Consists in defining the system architecture, its components (e.g., domain component, UI component), and related technologies (Bass et al., 2003). When adopting Ontology-Based Approach (OBA), the architecture must include components that portray the use of

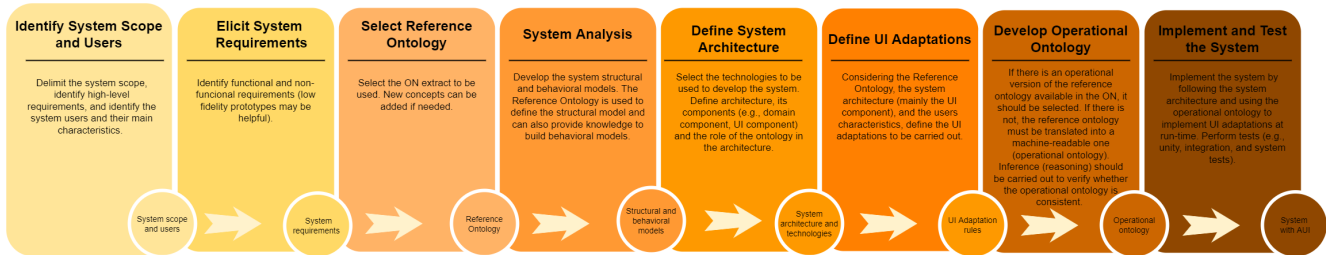


Figure 12. Overview of the ontology-based process to develop AUI systems.

operational ontology. The role of the operational ontology must be clearly defined in the architecture. For example, it can be used in a semantic reasoning engine to enable the system to make inferences based on the knowledge contained in the ontology. The operational ontology can also be used to support other aspects of the system, such as data integration or knowledge management. When selecting technologies for the system, it is important to consider the language and technologies used to implement the operational ontology. For example, OWL is a popular language for handling ontologies, and it has several compatible technologies for implementing operational ontologies, such as Apache Jena¹⁶ and Protégé.

(vi) *Define UI Adaptations:* In this step, the UI adaptation rules are defined. Considering the reference ontology, the system architectures (mainly the UI component), and the user's characteristics, the UI adaptations to be carried out must be defined. For example, it can be defined that if the user is brightness sensitive, the screen must be turned into the high contrast mode. It is recommended that the rules be stipulated very clearly (e.g., step by step) and structured as an algorithm in natural language (this will be helpful in the implementation step). It is important to note that at this stage, there may be no concern with the computer language that will implement the rules. The focus here is on defining the rules themselves, rather than worrying about how they will be implemented. Depending on the complexity of the UI adaptations, it may be necessary to define several rules. The reference ontology supports the mapping of user's characteristics to specific UI elements or adaptation needs. This helps define the adaptation rules. Moreover, existing standards, guidelines, and patterns can be considered to help define adequate adaptations (e.g., the W3C Accessibility Standard and the Web Content Accessibility Guidelines can be used to aid in defining accessibility adaptations).

(vii) *Develop Operational Ontology:* Consists in producing the operational ontology that will be used at run-time. If there is an operational version of the reference ontology available in the ON, it should be selected. If there is not, the reference ontology must be translated into a machine-readable one. Inference (reasoning) should be carried out to verify the operational ontology consistency. An advantage of using HCI-ON is that it provides the reference ontologies and also their operational versions in OWL. The adaptation rules defined in step (iv) (*Define UI Adaptations*) must be incorporated as axioms into the operational ontology. During run-time, the semantic reasoning engine defined in the system architecture uses the axioms to make inferences on user data and identify the most suitable UI adaptation for a particular user. It is

recommended to create some test cases and instantiate them in the operational ontology to verify whether the adaptation rules (axioms) are consistent. For this verification, after the operational ontology population, it is necessary to perform reasoning. This can be done by using tools such as Protégé. The operational ontology (e.g., the OWL file) resulting from this step will be used in the system implementation (next step).

(viii) *Implement and Test the System:* In this step, the system is implemented by following the system architecture and using the operational ontology to implement UI adaptations at run-time. The operational ontology is used by the semantic reasoning engine to identify the most suitable UI adaptation for a particular user based on their characteristics. Unitary, integration, and system tests must be carried out to ensure that the system properly meets the established functional and non-functional requirements. Unitary tests are carried out to test individual components of the system, while integration tests aim to test the interaction between different components. In system testing, the system is tested as a whole and may include different types of tests, such as performance testing, security testing, and compatibility testing (Herzig et al., 2015). Usability tests are necessary to evaluate the AUI. They can provide feedback on the effectiveness of the UI adaptations and their impact on the user experience.

It is worth pointing out that although the process is represented as a sequence of steps (for simplification reasons), there is an interaction between stages (e.g., it is possible to go back to steps (i) or (ii) after step (iii) to refine user's characteristics and requirements based on the reference ontology). This flexibility allows for the process to be adapted to the project's specific needs and can help ensure that the resulting system is effective and meets user's needs. Moreover, the process can be performed iteratively.

8 Related Work

In the literature, there are some works that propose the use of ontologies in the development of AUI systems (Costa et al., 2021). In this section, we summarize six of them and highlight how our work differs from their proposals.

Bonacin et al. (2022) use a recoloring ontology to develop a functional web prototype that changes the colors of UI elements automatically for colorblind users. The ontology is used to represent key aspects of the color adaptation process, as well as logical rules based on empirical data regarding preferences and access contexts. Braham et al. (2021), in turn, use ontologies and UI design patterns to develop a mobile appli-

¹⁶<https://jena.apache.org/>

cation that supports run-time adaptation of the UI for people with disabilities. The ontology is used to select appropriate design patterns for adapting the UI in different contexts. In the work by Stefanidi et al. (2022), ontologies are used to support the development of an AUI system aimed to improve users' situational awareness. The authors use ontologies to represent domain knowledge and combinatorial optimization to decide which information to present in the AUI. Khan and Khusro (2019) propose an AUI system for visually impaired users on touchscreen devices. An ontology is used to model and store concepts and relationships to UI adaptation. Sala et al. (2021) developed an automated adaptation system to enhance the accessibility of public e-services on the web. Ontologies are utilized to annotate relationships between e-services process components, user characteristics, and adaptation techniques, with the aim of generating AUIs. Fedasyuk and Lutsyk (2021) propose an adaptive system to help people with cognitive disabilities. An ontology is used to adapt the functionalities and graphical UI to individual user needs. All these works focused on the use of operational ontologies and did not follow a systematic process, which makes it difficult for other people to repeat the process to develop other systems.

Like in the works aforementioned, in our work, we propose to use ontologies to help develop AUI systems (specifically their software constituent). However, our proposal has some important differences. First, we argue for the use of well-founded reference ontologies, which are application-independent and, thus, can be used to develop different AUIs and different systems. Moreover, they can be translated into operational ontologies to be used at run-time. Second, we propose the use of ontologies of an ON. Thus, different ON extracts (i.e., ontologies containing different concepts) can be used to develop different systems. In addition, the set of user characteristics and other concepts represented in the ON increases over time (because the ON continuously evolves) enabling one to address new adaptations. Finally, we are moving towards defining a systematic process that (i) describes the steps to be followed to use ontologies from an ON to develop AUI systems and (ii) provides a knowledge framework that addresses relevant aspects to adaptive systems and AUI to support AUI systems development. As a benefit, third parties will be able to use the proposed process and the knowledge framework to develop AUI systems.

9 Final Considerations

The development of AUI systems is a complex and knowledge-intensive activity. Ontologies have been recognized as important tools for solving knowledge-related problems (Feilmayr and Wöß, 2016). Therefore, this paper aimed to explore the use of ontologies, particularly networked ontologies (i.e., ontologies from an ON), in the development of AUI systems. Although some works have already used ontologies to develop AUI systems, most of them focus on operational ontologies that are developed specifically for the system where they are used, which hampers reuse and evolution. These works also do not describe the followed process. Moreover, none of them considered the use of ONs. With this research, we give the first step to explore ONs to aid in

AUI systems development, leverage the use of ontologies at conceptual and operational levels, and define a systematic process to help in this matter.

The ontology used in the work addressed in this paper is an extract of HCI-ON (Costa et al., 2020, 2022), an ON that contains several ontologies addressing HCI subdomains. Thus, other extracts can be considered to extend SNOPI or develop other systems that address different UI adaptations. For example, in HCI-ON, there is the Context of Use Ontology, which addresses aspects related to the context of the use of the system (e.g., the device where the system runs). Concepts from this ontology can be used to define adaptations considering the device context (e.g., SNOPI could be extended to consider such adaptations). Moreover, as HCI-ON grows, other extracts containing new concepts can be considered to develop more comprehensive and effective AUI systems.

It is important to note that although we have developed SNOPI, it is not the main contribution of this work. The system was developed motivated by some needs of our research group and it is not the end of this research, but a means that helped us explore the use of ontologies from an ON to support the development of AUI systems and give the first step towards an approach with that purpose. This work contributes to the state of the art by exploring an ON to develop AUI systems and evolving a knowledge framework (HCI-ON) to grow knowledge of such systems and AUI. Moreover, the work contributes to practitioners by giving the first step to defining an ontology-based process that can be used in the development of AUI systems. The work has some limitations that must be considered together with the results presented in this paper. The main limitation is that the ontologies used in the study were developed by the authors and the process resulted from a study also carried out by the authors.

It is worth noting that, being an ON, HCI-ON can be constantly evolving (Costa et al., 2020). Extending HCI-ON provides the opportunity to include new adaptations in SNOPI or develop new tools. However, extending HCI-ON does not necessarily imply changes in OADAPT or in the architecture of SNOPI. Therefore, the current contributions remain valid over time. This means that as HCI-ON evolves and new concepts are added to it, the proposed ontology-based approach for developing AUI systems can still be used without significant changes to its underlying architecture. By leveraging the extensibility of HCI-ON, developers can continue to improve and refine their ontology-based AUI systems over time, while keeping compatibility with existing implementations.

We performed an interview with the SNOPI developer and the general results indicate that the use of networked ontologies (particularly an HCI-ON extract) is useful and feasible. At development time, the reference ontology helped in understanding the interactive system and AUI domain and, thus, how the system should work. It also aided to identify user characteristics to define different profiles. Moreover, it was used to define the system class diagram and the database structure. At run-time, the operational ontology helped capture user characteristics and change the UI accordingly. Some difficulties were perceived, mainly related to the lack of clear guidance on the process to be followed to use ontologies to develop the system. Considering this feedback, we defined a process to help in this matter.

This paper aimed to present the general idea of using an ON to help develop AUI systems and the main results we have obtained so far. This is a long-term research, and there are several involved challenges. For example, regarding the proposed process, it is necessary to refine it and provide detailed guidelines to perform its steps, so that other people can properly use it to develop their own AUI systems. It is also necessary to evaluate the process in practical settings and by third parties. Moreover, it should be evolved to be more flexible and cover situations where it is not desirable to use operational ontologies. Concerning the knowledge framework (i.e., the ON), it is necessary to extend the conceptualization in such a way that it describes AUI systems comprehensively (e.g., by considering a wider range of user characteristics, accessibility needs, customizations, context-aware systems, context of use, among others). In addition, it would be interesting to investigate the potential of using machine learning to support UI adaptations.

Our intention is to work to address challenges like these. Currently, we are dedicating efforts to three main fronts. First, we have a new developer using OADAPT to evolve SNOPI and another one using OADAPT to develop another AUI system from scratch. This experience is a new learning iteration in our methodological approach and will help us to obtain feedback about the process and learn more about it. Second, we are detailing each step of the process to provide the necessary guidelines for other people to use it to develop AUI systems. In this context, our focus is particularly on the steps involving the selection of the ON extract, and its use at both conceptual and operational levels. Lastly, we are extending HCI-ON to achieve a comprehensive and consistent knowledge framework to support AUI systems development.

Acknowledgements

This work is supported in part by CAPES/Brazil (Finance Code 001) and by the Espírito Santo Research and Innovation Support Foundation (FAPES). **Authors' contributions:** **Alexandre Adler Cunha de Freitas:** Conceptualization, Methodology, Writing - Original draft preparation. **Simone Dornelas Costa:** Conceptualization, Methodology, Writing- Reviewing and Editing. **Murilo Borghardt Scalsler:** Conceptualization, Software. **Monalessa Perini Barcellos:** Conceptualization, Methodology, Writing- Reviewing and Editing.

References

- Acuña, S. T., Castro, J. W., and Juristo, N. (2012). A hci technique for improving requirements elicitation. *Information and Software Technology*, 54(12):1357–1375. Special Section on Software Reliability and Security.
- Akiki, P. A., Bandara, A. K., and Yu, Y. (2014). Adaptive model-driven user interface development systems. *ACM Comput. Surv.*, 47(1).
- Barcellos, M., Santos, G., Conte, T., Trinkenreich, B., and Matsubara, P. (2022). Organizing empirical studies as learning iterations in design science research projects. In *Proceedings of the XXI Brazilian Symposium on Software Quality*, pages 1–10.
- Bass, L., Clements, P., and Kazman, R. (2003). *Software architecture in practice*. Addison-Wesley Professional.
- Bonacin, R., Reis, J. C. d., and de Araujo, R. J. (2022). An ontology-based framework for improving color vision deficiency accessibility. *Univers. Access Inf. Soc.*, 21(3):691–716.
- Braham, A., Buendía, F., Khemaja, M., and Gargouri, F. (2021). User interface design patterns and ontology models for adaptive mobile applications. *Personal and Ubiquitous Computing*, pages 1–17.
- Carraretto, R. and Almeida, J. P. A. (2012). Separating Ontological and Informational Concerns: Towards a Two-Level Model-Driven Approach. In *Proceedings of IEEE 16th International Enterprise Distributed Object Computing Conference Workshops*, pages 29–37, Beijing, China. IEEE.
- Castro, M. V. H. B. C. (2021). An Ontology to support Knowledge Management Solutions for Human-Computer Interaction Design. Master's thesis, Computer Science Department, Federal University of Espírito Santo (UFES).
- Costa, S. D. (2022). *An Ontology Network to support Knowledge Representation and Semantic Interoperability in the HCI Domain*. PhD thesis, Federal University of Espírito Santo.
- Costa, S. D., Barcellos, M. P., and Falbo, R. d. A. (2021). Ontologies in human-computer interaction: A systematic literature review. *Applied Ontology*, 16(4):421–452.
- Costa, S. D., Barcellos, M. P., Falbo, R. d. A., and Castro, M. V. H. B. (2020). Towards an Ontology Network on Human-Computer Interaction. In Dobbie, G., Frank, U., Kappel, G., Liddle, S. W., and Mayr, H. C., editors, *Proceedings of the 39th International Conference on Conceptual Modeling*, pages 331–341, Cham. Springer International Publishing.
- Costa, S. D., Barcellos, M. P., Falbo, R. d. A., Conte, T., and de Oliveira, K. M. (2022). A core ontology on the Human-Computer Interaction phenomenon. *Data & Knowledge Engineering*, 138:101977.
- Farooqui, T., Rana, T., and Maqbool, A. (2021). Categorization of user experience and usability issues (uxuis) and their prioritization w.r.t categorization for improving the uxu of interactive system (is). In *2021 International Conference on Communication Technologies (ComTech)*, pages 85–90.
- Fedasyuk, D. and Lutsyk, I. (2021). Tools for adaptation of a mobile application to the needs of users with cognitive impairments. In *2021 IEEE 16th International Conference on Computer Sciences and Information Technologies (CSIT)*, volume 1, pages 321–324.
- Feilmayr, C. and WöB, W. (2016). An analysis of ontologies and their success factors for application to business. *Data & Knowledge Engineering*, 101:1–23.
- Firmenich, S., Garrido, A., Paternò, F., and Rossi, G. (2019). User interface adaptation for accessibility. *Web Accessibility: A Foundation for Research*, pages 547–568.
- Freitas, A. A. C., Costa, S. D., Scalsler, M. B., and Barcellos, M. P. (2023). Supplementary material of the study "Using Networked Ontologies to Support the Development of Software Systems with Adaptive User Interface". (Version 1). Zenodo. <https://doi.org/10.5281/zenodo.7948977>.
- Freitas, A. A. C. d., Scalsler, M. B., Costa, S. D., and Barcellos,

- M. P. (2022). Towards an ontology-based approach to develop software systems with adaptive user interface. In *Proceedings of the 21st Brazilian Symposium on Human Factors in Computing Systems, IHC '22*, New York, NY, USA. Association for Computing Machinery.
- Glimm, B., Horrocks, I., Motik, B., Stoilos, G., and Wang, Z. (2014). Hermit: an owl 2 reasoner. *Journal of Automated Reasoning*, 53:245–269.
- Gorman, K. and Choobineh, J. (1990). The object-oriented entity-relationship model (ooerm). *Journal of Management Information Systems*, 7(3):41–65.
- Gray, D., Brown, S., and Macanufo, J. (2010). *Gamestorming: A Playbook for Innovators, Rulebreakers, and Changemakers*. O'Reilly Media, Inc., 1st edition.
- Gruber, T. R. (1993). A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5(2):199–220.
- Guarino, N. (1998). *Formal Ontology in Information Systems: Proceedings of the 1st International Conference June 6-8, 1998, Trento, Italy*. IOS Press, Amsterdam, The Netherlands, The Netherlands, 1st edition.
- Guarino, N., Oberle, D., and Staab, S. (2009). What Is an Ontology? In *Handbook on Ontologies*, pages 1–17. Springer Berlin Heidelberg, Berlin, Heidelberg.
- Guizzardi, G. (2005). *Ontological foundations for structural conceptual models*. PhD thesis, University of Twente.
- Guizzardi, G. (2007). On Ontology, Ontologies, Conceptualizations, Modeling Languages, and (Meta)Models. In Vasilecas, O., Edler, J., and Caplinskas, A., editors, *Proceedings of the Frontiers in Artificial Intelligence and Applications, Databases and Information Systems IV*, pages 18–39, Amsterdam, The Netherlands, The Netherlands. IOS Press.
- Guizzardi, G., Falbo, R., and Guizzardi, R. S. S. (2008). Grounding software domain ontologies in the unified foundational ontology (ufo): The case of the ode software process ontology. In *Proceedings of the 1th Iberoamerican Workshop on Requirements Engineering and Software Environments (IDEAS'2008)*, pages 127–140.
- Guizzardi, G., Wagner, G., Falbo, R. d. A., Guizzardi, R. S. S., and Almeida, J. P. A. (2013). Towards Ontological Foundations for the Conceptual Modeling of Events. In Ng, W., Storey, V. C., and Trujillo, J. C., editors, *Proceedings of the Conceptual Modeling*, pages 327–341, Berlin, Heidelberg. Springer Berlin Heidelberg.
- Gullà, F., Ceccacci, S., Germani, M., and Cavalieri, L. (2015). Design adaptable and adaptive user interfaces: a method to manage the information. In *Ambient Assisted Living: Italian Forum 2014*, pages 47–58. Springer.
- Gurcan, F., Cagiltay, N. E., and Cagiltay, K. (2021). Mapping human–computer interaction research themes and trends from its existence to today: A topic modeling-based review of past 60 years. *International Journal of Human–Computer Interaction*, 37(3):267–280.
- Herzig, K., Greiler, M., Czerwonka, J., and Murphy, B. (2015). The art of testing less without sacrificing quality. In *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering*, volume 1, pages 483–493.
- Hevner, A. R. (2007). A three cycle view of design science research. *Scandinavian journal of information systems*, 19(2):4.
- Hevner, A. R., March, S. T., Park, J., and Ram, S. (2008). Design science in information systems research. *Management Information Systems Quarterly*, 28(1):6.
- Hussain, J., Ul Hassan, A., Muhammad Bilal, H. S., Ali, R., Afzal, M., Hussain, S., Bang, J., Banos, O., and Lee, S. (2018). Model-based adaptive user interface based on context and user experience evaluation. *Journal on Multimodal User Interfaces*, 12:1–16.
- Khan, A. and Khusro, S. (2019). Blind-friendly user interfaces—a pilot study on improving the accessibility of touchscreen interfaces. *Multimedia Tools and Applications*, 78:17495–17519.
- Machado, E., Singh, D., Cruciani, F., Chen, L., Hanke, S., Salvago, F., Kropf, J., and Holzinger, A. (2018). A conceptual framework for adaptive user interfaces for older adults. In *2018 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, pages 782–787.
- Microsoft (2021). Enterprise application patterns with xamarin.forms: Mvvm. accessed on May 3, 2023.
- Moran, K., Bernal-Cárdenas, C., Curcio, M., Bonett, R., and Poshyvanyk, D. (2020). Machine learning-based prototyping of graphical user interfaces for mobile apps. *IEEE Transactions on Software Engineering*, 46(2):196–221.
- Musen, M. A. (2015). The protégé project: a look back and a look forward. *AI Matters*, 1(4):4–12.
- Norcio, A. F. and Stanley, J. (1989). Adaptive human-computer interfaces: A literature survey and perspective. *IEEE Transactions on Systems, Man, and cybernetics*, 19(2):399–408.
- Oppermann, R., editor (1994). *Adaptive User Support: Ergonomic Design of Manually and Automatically Adaptable Software*. L. Erlbaum Associates Inc., USA.
- Pruitt, J. and Grudin, J. (2003). Personas: practice and theory. In *Proceedings of the 2003 conference on Designing for user experiences*, pages 1–15.
- Rathnayake, N., Meedeniya, D., Perera, I., and Welivita, A. (2019). A Framework for Adaptive User Interface Generation based on User Behavioural Patterns. In *Proceedings of 2019 Moratuwa Engineering Research Conference (MER-Con)*, pages 698–703, Moratuwa, Sri Lanka. IEEE.
- Robson, C. and McCartan, K. (2016). *Real world research*, 4th edn. hokoben.
- Ruy, F. B., Falbo, R. d. A., Barcellos, M. P., Costa, S. D., and Guizzardi, G. (2016). Seon: A software engineering ontology network. In Blomqvist, E., Ciancarini, P., Poggi, F., and Vitali, F., editors, *Proceedings of Knowledge Engineering and Knowledge Management*, pages 527–542, Cham. Springer International Publishing.
- Sala, A., Arrue, M., Pérez, J. E., and Espín-Tello, S. M. (2021). Automated adaptations for improving the accessibility of public e-services based on annotations. In Ardito, C., Lanzilotti, R., Malizia, A., Petrie, H., Piccinno, A., Desolda, G., and Inkpen, K., editors, *Human-Computer Interaction – INTERACT 2021*, pages 373–382, Cham. Springer International Publishing.
- Salminen, J., Jung, S.-G., Chowdhury, S., Sengün, S., and Jansen, B. J. (2020). Personas and analytics: A comparative

- user study of efficiency and effectiveness for a user identification task. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20, page 1–13, New York, NY, USA. Association for Computing Machinery.
- Sattar, A., Ahmad, M. N., Surin, E. S. M., and Mahmood, A. K. (2021). An improved methodology for collaborative construction of reusable, localized, and shareable ontology. *IEEE Access*, 9:17463–17484.
- Scherp, A., Saathoff, C., Franz, T., and Staab, S. (2011). Designing core ontologies. *Applied Ontology*, 6(3):177–221.
- Sebek, J., Trnka, M., and Cerny, T. (2015). On Aspect-Oriented Programming in Adaptive User Interfaces. In *Proceedings of 2015 2nd International Conference on Information Science and Security (ICISS)*, pages 1–5, Seoul, Korea (South). IEEE.
- Seedorf, S., Informatik, F. F., and Mannheim, U. (2006). Applications of ontologies in software engineering. In *Proceedings of Workshop on Semantic Web Enabled Software Engineering (SWESE) on the ISWC*, pages 5–9, Athens, Geórgia. Citeseer, 2nd International Workshop on Semantic Web Enabled Software Engineering (SWESE 2006), held at the 5th International Semantic Web Conference (ISWC 2006).
- Sommerville, I. (2020). *Engineering software products*, volume 355. Pearson London.
- Stefanidi, Z., Margetis, G., Ntoa, S., and Papagiannakis, G. (2022). Real-time adaptation of context-aware intelligent user interfaces, for enhanced situational awareness. *IEEE Access*, 10:23367–23393.
- Suárez-Figueroa, M. C., Gómez-Pérez, A., Motta, E., and Gangemi, A. (2012). *Introduction: Ontology Engineering in a Networked World*, pages 1–6. Springer Berlin Heidelberg, Berlin, Heidelberg.
- Vasilieva, E. (2018). Developing the creative abilities and competencies of future digital professionals. *Automatic Documentation and Mathematical Linguistics*, 52:248–256.
- Yen, G. G. and Acay, D. (2009). Adaptive user interfaces in complex supervisory tasks. *ISA Transactions*, 48(2):196–205.
- Yigitbas, E., Josifovska, K., Jovanovikj, I., Kalinci, F., Anjorin, A., and Engels, G. (2019). Component-based development of adaptive user interfaces. In *Proceedings of the ACM SIGCHI Symposium on Engineering Interactive Computing Systems*, EICS '19, New York, NY, USA. Association for Computing Machinery.
- Yigitbas, E., Jovanovikj, I., Biermeier, K., Sauer, S., and Engels, G. (2020). Integrated model-driven development of self-adaptive user interfaces. *Softw. Syst. Model.*, 19(5):1057–1081.