A Core Ontology on the Human-Computer Interaction phenomenon ⋆

Simone Dornelas Costa[a,b,*], Monalessa Perini Barcellos[b,*], Ricardo de Almeida Falbo[b], Tayana Conte[c], Káthia M. de Oliveira[d]

[a] *Computer Department, Federal University of Espírito Santo (UFES), Alegre, Brazil*
[b] *Ontology & Conceptual Modeling Research Group (NEMO), Computer Science Department, Federal University of Espírito Santo (UFES), Vitória, Brazil*
[c] *USES Research Group, Institute of Computing (IComp), Federal University of Amazonas (UFAM), Manaus, AM, Brazil*
[d] *Université Polytechnique Hauts-de-France Valenciennes, LAMIH UMR CNRS 8201, France*

Abstract

Human-Computer Interaction (HCI) is a complex communication phenomenon involving human beings and computer systems that gained large attention from industry and academia with the advent of new types of interactive systems (mobile applications, smart cities, smart homes, ubiquitous systems and so on). Despite of its importance, there is still a lack of formal and explicit representations of what the HCI phenomenon is. In this paper, we intend to clarify the main notions involved in the HCI phenomenon, by establishing an explicit conceptualization of it. To do so, we need to understand what interactive computer systems are, which types of actions users perform when interacting with an interactive computer system, and finally what human-computer interaction itself is. The conceptualization is presented as a core reference ontology, called HCIO (HCI Ontology), which is grounded in the Unified Foundational Ontology (UFO). HCIO was evaluated using ontology verification and validation techniques and has been used as core ontology of an HCI ontology network.

Keywords: Human-Computer Interaction, User Interface, Interactive Computer System, Ontology, Ontology Network

1. Introduction

Human-Computer Interaction (HCI) is currently in evidence with the large use of interactive systems supporting daily activities and the advent of new technologies. HCI is a knowledge and a multidisciplinary area that aggregates a vast and multifaceted community Carroll (2014). It is connected to other research areas, involving knowledge from multiple fields, such as ergonomics, cognitive science, user experience, human factors, among others Sutcliffe (2014). As the HCI area continues to mature, new terms are proposed and new meanings are assigned to existing terms. As a consequence, it is not trivial to achieve a common conceptualization of HCI, leading to semantic interoperability problems (such as ambiguity and imprecision when interpreting shared information) and hampering communication and knowledge transfer Carroll (2014).

The lack of a common conceptualization causes misunderstanding and interoperability problems when dealing with HCI references (e.g., books, standards, research papers) coming from different sub-communities. Many times, even references from the same sub-community are not harmonized. For example, the Software and Systems Engineering Vocabulary (SEVOCAB) ISO (2017) presents three slightly different definitions for "user interface": (i) all components of an interactive system (software or hardware) that provide information and controls for the user to accomplish specific tasks with the interactive system ISO (2014); (ii) ensemble of software and hardware that allows a user to interact with a system ISO (2008, 2018); (iii) interface that enables information to be passed between a human user and hardware or software components of a computer system ISO (2012). Definition (i) explicitly mentions interactive system (e.g., tablets, smart phones, wearable devices), while definitions (ii) and (iii) refer respectively to system and computer system (e.g., servers). In these definitions, do interactive system, computer system and system have the same meaning? This is not clear. Hence, it is not clear if user interface exists only in interactive systems or in any kind of computer system or system. It is worth pointing out that the three definitions come from different standards defined by the same sub-committee (SC 7 - Software and Systems Engineering) of the Joint Technical Committee JTC 1 (Information Technology) that joins two standardization organizations, namely the International Organization for Standardization (ISO) and the International Electrotechnical Commission (IEC). Therefore, to properly understand different terminologies and conceptualizations used in the HCI domain, it is important to establish a reference conceptualization of what the HCI phenomenon is.

Ontologies have been successfully used in several domains (e.g., IT Service Management Pardo et al. (2013), Health Liyanage et al. (2015); Sene et al. (2018) and Education Yago et al. (2018)) to capture and organize knowledge to deal with interoperability and knowledge-related problems, as the ones aforementioned. Aiming to investigate the use of ontologies in the HCI domain, we carried out a systematic literature review and identified 22 ontologies. By analyzing them, we noticed that there are inconsistencies among their conceptualizations, even in HCI core concepts.

We argue that to properly solve interoperability and knowledge-related problems, we need a reference ontology, i.e., a special kind of conceptual model representing a model of consensus within a community. It is a solution-independent specification with the aim of making a clear and precise description of domain entities

⋆doi: `https://doi.org/10.1016/j.datak.2021.101977`
*Corresponding author
Email addresses: `simone.costa@ufes.br` (Simone Dornelas Costa), `monalessa@inf.ufes.br` (Monalessa Perini Barcellos), `tayana@icomp.ufam.edu.br` (Tayana Conte), `kathia@uphf.fr` (Káthia M. de Oliveira)

for the purposes of communication, learning and problem-solving [12]. Reference ontologies have been used in several domains such as Criminal (Rodrigues et al. (2020)), Business Modeling (Andersson et al. (2006)), Biomedical Informatics (Rosse and Mejino (2003)) and Software Engineering (Bastos et al. (2018); Renault et al. (2018)). Moreover, considering that results of our systematic literature showed a lack of a common understanding of the HCI phenomenon, we advocate that a core conceptualization of the HCI phenomenon is necessary and should be the first step towards defining a comprehensive conceptualization of HCI. The core conceptualization provides central concepts that are shared across the HCI domain and can be reused to address other HCI aspects, contributing to the establishment of a consistent and more comprehensive conceptualization of the HCI domain. Reference core ontologies have also been used in different areas such as Service (Nardi et al. (2015)), Software Measurement (Barcellos et al. (2014)), Customer Relationship Management (Magro and Goy (2012)) and Video Scene Interpretation (Sikos (2018)).

Hence, in this paper we introduce HCIO (Human-Computer Interaction Ontology), a reference ontology that focuses on core aspects of the HCI phenomenon. HCIO is grounded in the Unified Foundational Ontology – UFO Guizzardi et al. (2015) and reuses concepts from the System and Software Ontology (SysSwO) Bringuente et al. (2011); Duarte et al. (2018). HCIO is organized in three sub-ontologies: (i) Interactive Computer System sub-ontology focuses on what an interactive computer system is and its constituent elements, including the user interface; (ii) User sub-ontology focuses on the user and intentional or unintentional actions performed by users when interacting with an interactive computer system; and (iii) Human-Computer Interaction sub-ontology links concepts from the other two sub-ontologies to define what a human-computer interaction is. HCIO can be used as a reference model for communication and learning purposes and can serve as a bridge to solve semantic interoperability problems, such as harmonizing the understanding of different standards or other HCI references. Moreover, as we said above, HCIO can be reused in the development of ontologies addressing specific aspects of HCI (e.g., HCI design, HCI evaluation, etc.) or to extend and integrate existing ontologies. In this sense, we have used HCIO as core ontology of an HCI ontology network (named HCI-ON Costa et al. (2020)) in which HCIO provides the core concepts shared by all domain ontologies of the network.

This paper is organized as follows. Section 2 presents briefly the theoretical basis for this paper. Section 3 presents the 22 HCI-related ontologies we found in a systematic literature review and discusses related works. Section 4 presents the adopted methodological approach. Section 5 presents two real cases of HCI that are later used in Section 6 to demonstrate that HCIO is able to represent real-world situations. Section 6 presents HCIO and how we evaluated it. Section 7 makes some discussions about HCIO conceptualization. Section 8 concerns the use of HCIO as core ontology of an HCI ontology network (HCI-ON) and to support the evaluation of user experience in an immersive application. Some envisioned applications of HCIO are also discussed. Finally, Section 9 concludes the paper.

2. Background

2.1. Human-Computer Interaction

Human-Computer Interaction (HCI) can be defined as the discipline responsible for the analysis, design, implementation and evaluation of interactive computer systems for human use Preece et al. (2015). This discipline has evolved since the 1980s[1] through various terminologies, classifications and studies.

Dix et al. (2004) consider the communication between user and interactive computer system the interaction itself. User and system are, thus, participants in the interaction. Briefly, a human-computer interaction is the communication process that occurs during the use of an interactive computer system and that involves user actions on the system interface (user input) and user interpretations of the system responses (system output) revealed through the user interface (Figure 1). The user interface includes all parts of the system that a user has contact with, physical, perceptually or conceptually Benyon (2010). Interactive computer systems aid in goals achievement by supporting the accomplishment of tasks in some application domain or context of use where users interact with the system through its interface.

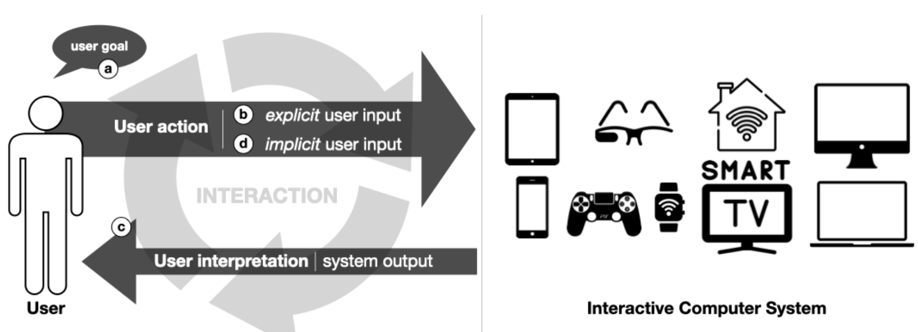

Figure 1: Human-Computer Interaction: (a) user goal triggering the interaction, (b) user action, explicit user input, (c) system output (triggering the interaction or not) and user interpretation, (d) user action (does not rely on the user's intentionality), implicit user input

According to Norman (2013), the interaction cycle can start from the top, in a *goal-driven behavior* (Figure 1, (a)), where the user first establishes a goal to be achieved and then goes through user actions to accomplish the goal. In Figure 1, (a) together with (b) represents that the interaction starts with the

[1]HCI term started to be largely used after the publication of the book entitled "The Psychology of Human–Computer Interaction" by Stuart K. Card, Thomas P. Morton and Allen Newel, in 1983.

goal establishment and a user action that triggers the interaction cycle. For example, Mary has the goal of sending an email to David (Figure 1, (a)). Thus, she types a message in her laptop (user input) and sends it to David (Figure 1, (b)). The email system shows a notification that the email was sent (system output) and Mary interprets that she has achieved her goal (Figure 1, (c)). The interaction cycle can also start from the bottom, in a *data-driven* or *event-driven behavior*, triggered by some event in the world (e.g., an event caused by an interactive computer system) and then can go through user actions (Figure 1, (c) when the system output triggers the interaction). In the previous example, consider that David (who was not expecting to receive an email from Mary at that moment) is notified by his smartphone (system output) that he has received an email (Figure 1, (c)). He interacts with his smartphone (inputting data in it and interpreting its output) due to the system notification (Figure 1, (b)). After that, he can start a new interaction cycle, now in a *goal-driven behavior* (starting from Figure 1, (a)), motivated by his goal of answering Mary's email.

The aforementioned perspective refers to *traditional* and *explicit human-interaction* Schmidt (2000), i.e., the user explicitly enters data through the user interface (explicit input) and the interactive system, in turn, uses output devices to deliver information Schmidt (2005). In Mary's example, she types the message because of her goal and, when she is typing the message (explicit input), she is aware of the changes occurring in the user interface (system output).

Some interactions, said *implicit human interaction*, are more transparent, natural and not explicit. They are performed by the user that is not aimed to interact with the interactive system, but the system understands the user actions as inputs Schmidt (2000). For example, David wears a smartwatch and is lying down reading a book. The smartwatch constantly captures physiological information (e.g, David's pressure and heart rate), i.e., David unintentionally provides inputs to the system (Figure 1, (d)). Suddenly, the wind blows through the window and David gets up very fast and runs to hold a door that would slam. The smartwatch notifies David (Figure 1, (c) system output) that his heart rate increased faster than usual in the last seconds. David then interprets that it was because he got up very quickly and ran (Figure 1, (c) user interpretation).

2.2. Ontologies

An ontology is a formal and explicit specification of a shared conceptualization Gruber (1995). According to Scherp et al. (2011), ontologies can be classified into foundational, core, and domain ontologies. *Foundational ontologies* aim at modeling the very basic and general concepts and relations that make up the world (e.g., objects, events, participation and parthood). They are generic across any field and are highly reusable in different modeling scenarios. *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. Finally, *domain ontologies* describe knowledge that is specific for a particular domain, such as a soccer ontology. They can make use of/be based on foundational ontologies or core ontologies by specializing their concepts. Falbo et al. (2013) argue that Scherp et al. (2011) classification should be perceived as a continuum, ranging from pure foundational ontologies to domain ontologies. Thus, there can be different levels of generality in ontologies classified in a certain type. For instance, there are more general core ontologies, such as UFO-S Nardi et al. (2015), which addresses services in general, and more specific core ontologies, such as the Software Process Ontology Bringuente et al. (2011), which addresses core concepts about process in the Software Engineering area and spans across several subdomains in that area, such as software measurement process, software design process, software test process and so on.

Another important distinction differentiates ontologies as conceptual models, called *reference ontologies*, from ontologies as computational artifacts, called *operational ontologies* Guizzardi (2007). The former is constructed with the goal of making the best possible description of the domain in reality, representing a model regardless of its computational properties Studer et al. (1998). The latter is designed with the focus on guaranteeing desirable computational properties and, thus, are machine-readable.

In this work, we present HCIO (Section 6), a *reference* and *core ontology* that aims to provide a conceptualization of the HCI phenomenon. In order to be more faithful to reality Scherp et al. (2011); Guarino (1998), HCIO is grounded in UFO (Unified Foundational Ontology) Guizzardi et al. (2015). Moreover, considering that HCI core conceptualization involves aspects related to interactive computer systems, HCIO reuses concepts from the System and Software Ontology (SysSwO) Bringuente et al. (2011); Duarte et al. (2018), a core reference ontology that addresses aspects related to computer system and software. Next, we present the UFO fragment and the SysSwO view relevant to this work.

2.2.1. UFO

UFO is a foundational ontology that is constituted by an ontology of endurants (objects) Guizzardi (2005); Guizzardi et al. (2018), an ontology of events (perdurants) Guizzardi et al. (2008, 2013), and, an ontology of social entities Guizzardi et al. (2008). Figure 2 presents the fragment of UFO used to ground HCIO. The description below is based mainly on Guizzardi et al. (2008) and Guizzardi et al. (2013).

In UFO, *Individuals* are entities that exist in reality possessing a unique identity and are those that necessarily cannot be instantiated (e.g., John). *Endurants* are said to be wholly present whenever they are present (e.g., a person), i.e., they *are in time*. In UFO, there are three main categories of endurants: Substantials, Moments and Situations. *Substantials* are existentially independent individuals (such as a person, a house). *Moments*, in contrast, are individuals that denote properties and can only exist in other individuals. *Intrinsic Moments* are those moments that depend on one single individual (e.g., the color of a car, the temperature of a person). *Situations* are complex entities that are constituted by possibly many endurants (including other situations). A situation represents a state of affairs, i.e., a portion of reality that can be understood as a whole (e.g., "John being with fever and influenza"). Endurants are present in the situations they constitute. For instance, both the substantial John, and the intrinsic moments "John's fever" are present in the situation cited above. Finally, *Disposition* is a special kind of Intrinsic

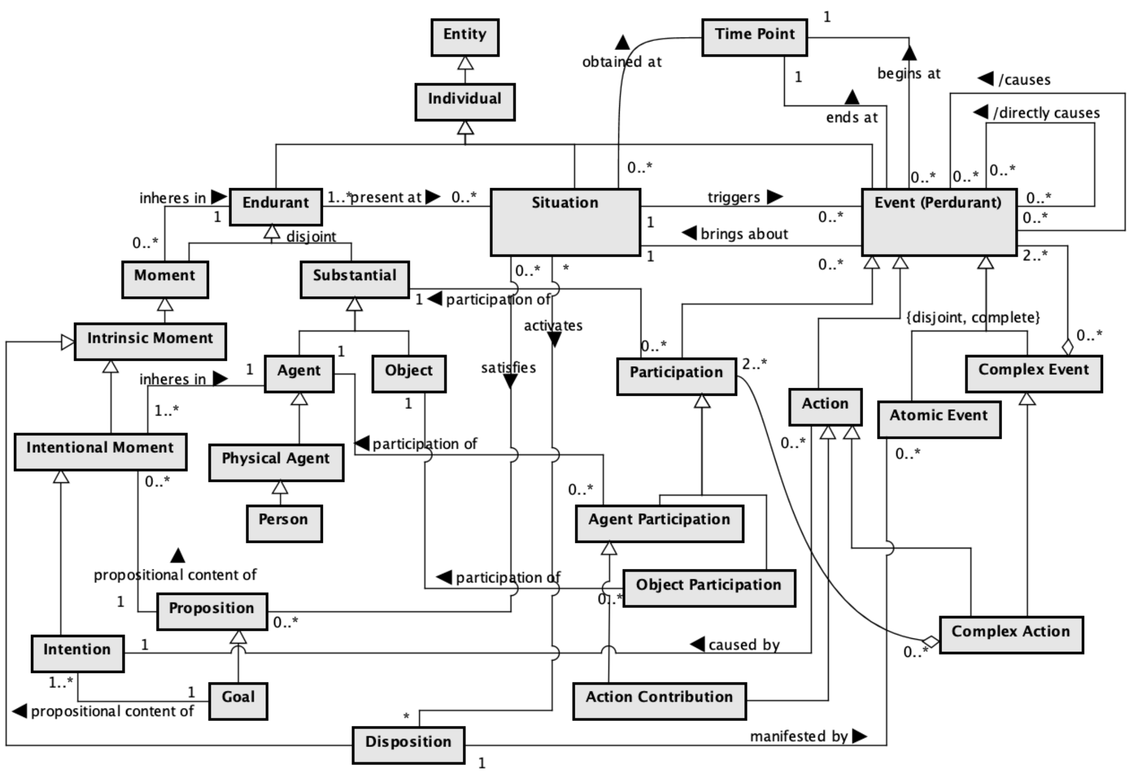

Figure 2: UFO Fragment relevant to this work

Moment, representing properties that are only manifested in particular situations and that can also fail to be manifested (e.g., the fragility of a glass, or the disposition of a magnet to attract metallic material). When manifested, they are manifested through the occurrence of Events.

Perdurants (Events) are individuals composed of temporal parts (e.g., a soccer game), i.e., they *happen in time* in the sense that they extend in time accumulating temporal parts. Whilst *Atomic Events* have no proper parts, *Complex Events* are aggregations of at least two disjoint events. Events depend on substantials to exist. A complex event can be partitioned by separating each part of this event which is existentially dependent on each of its participants. The portion of an event that depends exclusively on a single substantial is said a *Participation* (e.g., the participation of a player in a soccer game). Events are transformations from a portion of reality to another, i.e., they may change reality by changing the state of affairs from one situation to another. There are two possible relations between situations and events: (i) a situation *triggers* an event in the case that the event occurs because of the obtaining of the situation, and; (ii) an event *brings about* a situation in the case that the occurrence of the event results in the situation. Suppose that a situation s was brought about by an event e, and that is triggered another event e'. In this case, we can state that the occurrence of e *directly causes* the occurrence of e'. Moreover, if e' *directly causes* another event e", then we can state that the first event e *causes* the last event e".

In UFO, agentive and non-agentive substantial individuals are termed *Agents* and *Objects*, respectively. *Person* is a subtype of *Physical Agent*. Agents are substantials that can bear special kinds of intrinsic moments named *Mental Moments*. Every mental moment has a *Proposition* as its propositional content. "Intending something" is a specific type of mental moment termed *Intention*. The propositional content of an Intention is a *Goal*. Situations in reality can satisfy the propositional content of a mental moment. Thus, intentions can be fulfilled or frustrated. Intentions are desired situations for which the agent commits to pursuing. For this reason, intentions cause the agent to perform actions. *Actions* are intentional events, i.e., events with the specific purpose of satisfying (the propositional content of) some intention. Participations can also be intentional (i.e., be themselves actions) or unintentional events. For example, the stabbing of Caesar by Brutus includes the intentional participation of Brutus and the unintentional participation of Caesar and the knife. Thus, it is not the case that every *Agent Participation* is considered an action, but only those intentional participations, which are said *Action Contributions*. Only agents (entities capable of bearing intentional moments) can perform actions. The participations of objects in events are always unintentional and are said *Object Participations*.

2.2.2. System and Software Ontology (SysSwO)

SysSwO is a core reference ontology about the nature of system and software, including, software artifacts, software constitution, software execution, computer system and hardware equipment. Figure 3 presents the SysSwO view that was reused in HCIO. SysSwO is grounded in UFO. The blue arrows in Figure 3 indicate specializations from UFO's concepts). In the text, concepts from SysSwO are written in *underline italics* and concepts from UFO are written in *italics*.

Computer System is an *Object* combining hardware and software. Concerning hardware, a *Computer System* is composed of a set of *Computer Machines*. *Computer Machine* is a *Hardware Equipment* which other *Hardware Equipment* connect to it. *Hardware Equipment* is a physical *Object* used for running software programs or to support some software process activity (e.g., a printer, a router, a mouse, a keyboard).

Regarding software, a *Computer System* has a set of copies of software systems that are installed and loaded in the *Computer Machines* that comprise the *Computer System*. A copy of a software system installed and loaded in a *Computer Machine* is said a *Loaded Software System Copy* and it is a *Disposition* inhering in the *Computer Machine* where it is loaded. A *Loaded Software System Copy* is the materialization of a

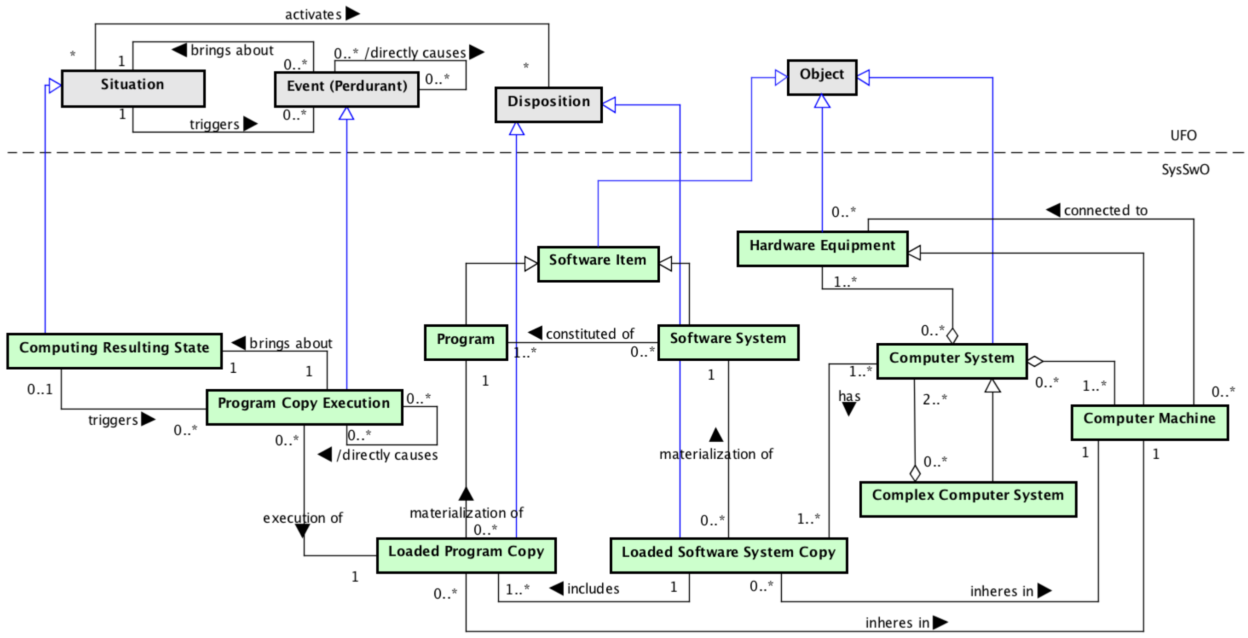

Figure 3: SysSwO view relevant for this work

Software System, which is a subtype of *Software Item*. A *Software Item*, in turn, is a piece of software produced during the software development process, not considered a complete software product (e.g., a program).

A *Software System* is constituted of *Programs* (also a subtype of *Software Item*). Analogously to a *Software System*, to run a *Program*, one must have a copy of the program loaded in a *Computer Machine* (*Loaded Program Copy*) and execute it. *Program Copy Execution* is then an *Event* that brings about a particular *Situation* (the post-state of the event termed as a *Computing Resulting State*) resulting from the *Program Copy Execution*. A *Computing Resulting State* (*Situation*) can trigger other *Program Copy Execution* (*Event*). The *Program Copy Execution* (*pce1*) that brought about a *Computing Resulting State* that triggered a second *Program Copy Execution* (*pce2*) is said to directly cause it (*pce1* directly causes *pce2*).

Finally, a *Computer System* can be composed of others *Computer Systems*. In this case, it is said a *Complex Computer System*.

3. Related Works

Before developing HCIO, we carried out a systematic literature review (SLR) to investigate the state of the art of ontologies addressing HCI aspects. A SLR is a rigorous study used to gather and synthesize evidence into a research topic, guided by research questions Jalali and Wohlin (2012); Kitchenham and Charters (2007). To perform the study, we followed the process defined in Kitchenham and Charters (2007). The study goal was to identify existing HCI ontologies, understand how they have been developed and used to solve HCI problems. For achieving this goal, we defined the following main research questions:

- RQ1: Which aspects of the HCI domain have been covered by the existing ontologies?
- RQ2: What have been the uses of HCI ontologies?
- RQ3: Have the HCI ontologies been developed by following ontology engineering methods?
- RQ4: How have the HCI ontologies been evaluated?
- RQ5: Which quality characteristics have HCI ontologies exhibited?

Since the SLR itself is not the focus of this paper, in this section we summarize general information about the SLR and discuss characteristics of the ontologies we have found. An updated version of the SLR is featured in the paper by Costa et al. (2021).

We applied the following search string *("user interface ontology" OR "user interface ontologies" OR "UI ontology" OR "UI ontologies") OR (("human-computer interaction" OR "HCI" OR "user interface design" OR "user interaction design" OR "user centered design" OR "human-centered design" OR "UI design" OR "user interface evaluation" OR "user interface assessment" OR "user interaction evaluation" OR "user interaction assessment" OR "UI evaluation" OR "UI assessment" OR "user interaction") AND ("ontologies" OR "ontology"))* to Scopus, Engineering Village and Web of Science digital libraries. We selected these sources because Scopus is one of the largest databases of peer-reviewed literature. It indexes papers from other important sources such as IEEE, ACM and Science Direct, providing useful tools to search, analyze and manage scientific research. Complementarily, to increase coverage, we selected Engineering Village and Web of Science, which are also widely used in secondary studies.

We found 1,598 papers published until 2018. After eliminating duplicates, 847 papers remained. Then, we applied two filters to select relevant publications. In the 1st filter, we read the abstracts of the publications considering the following inclusion (IC) and exclusion (EC) criteria: (IC1) the paper presents an ontology about the HCI domain; (EC1) the paper does not present an ontology about the HCI domain; (EC2) the paper does not have an abstract; (EC3) the paper was published only as an abstract; (EC4) the paper is a secondary study, tertiary study, editorial, summary of keynote, tutorial or the proceedings of a scientific event; (EC5) the paper is not written in English. 110 papers were selected in the 1st filter. In the 2nd filter, we read the full text of the publications selected in the 1st filter and analyzed them considering the aforementioned criteria plus the following: (EC6) the paper is an older version (outdated) of another paper already considered; and (EC7) we did not have access to the full text of the paper. EC6 served to avoid repetition and EC7 to treat publications whose full text was not available for the researchers. As a result, from 25 papers selected in the 2nd filter, we identified 22 ontologies (some ontologies are complementarily addressed in more than one paper), which are listed in Table 1.

Table 1: HCI-related ontologies found in the literature

Id	Description	Ref.
#01	It presents a conceptual model, based on references such as Norman (1986) in the form of a textual description which addresses HCI phenomenon and the involved parts (user and system), representing concepts such as Participants, User and Interaction.	Storrs (1994)
#02	It presents a conceptual model on task modeling and interaction with focus on UI (User Interface) design.	Suàrez et al. (2004)
#03	It presents a conceptual model on UI design addressing concepts such as Target User, Usability Requirement and Graphic Design Property.	Bakaev and Avdeenko (2010, 2012)
#04	It presents an ontology for interactive adaptive systems, addressing concepts such as DialogSystem, DialogDomain, UserModelItem and Adaptation.	Bezold and Minker (2010)
#05	It presents a small extract of an operational ontology implemented in OWL that addresses interaction patterns and web interfaces aspects such as web structural and visual elements. Widget, Event and Presentation Properties are some of the ontology concepts.	Celino and Corcoglioniti (2010)
#06	It presents an abstract widget ontology, by means of a conceptual model, addressing concepts related to UI objects such as Abstract Interface Element, Element Exhibitor, Variable Capture.	Martín et al. (2010)
#07	It presents a UI ontology that addresses UI objects. It is implemented using OWL/RDF and includes concepts such as Widget, Frame, Tab/Page, Textual and Multimedia. The ontology is used to generate a UI model, which is used to generate user interfaces.	Shahzad et al. (2011)
#08	It presents a conceptual model of an HCI ontology which addresses UI design under the perspective of interaction modality (the interaction itself is not covered). Represents concepts such as User, Device, Modality, InputModality and OutputModality.	Tourwé et al. (2011)
#09	It presents a gesture ontology addressing aspects related to gesture under the perspective of gesture interaction (the interaction itself was not covered). Represents concepts such as User, Body Movements, Execution Enhancer and Device. Its intended use is to inform and describe the software architecture design for controlling smart homes with gesture commands.	Chera et al. (2012)
#10	It presents an ontology-based model addressing user actions by relating it with the user and the resources used during his/her interaction.	Devaurs et al. (2012)
#11	It presents a markup ontology regarding the structure of a web form. It is implemented using RDF and includes concepts such as Widget Element, Button, Textbox, Widget. The ontology is used in a RESTful Web service to semantically annotate web widgets.	Haller et al. (2012)
#12	It presents a persona ontology that addresses persona characterization and aspects related to usability tests. The ontology is shown as a conceptual model and an operational ontology (implemented using OWL). It includes concepts such as Person, Persona, Usability Test.	Negru and Buraga (2013, 2012)
#13	It presents a persona ontology by means of a conceptual model and an operational ontology implemented using OWL. Includes concepts such as Persona, Person, Postal Address, Image, Goals, Personal Goal, Context. A web application uses the operational ontology in an ontology-based search for personas.	Salma and Marouane (2016); Salma et al. (2012)
#14	It presents a haptic application software modeling ontology implemented using OWL/SWRL. The ontology is intended to serve as a basis to design effective user interface and assist the development of software modeling for haptic devices. It includes concepts such as Human Haptic System, User, Perception, User Interaction with a Haptic Device.	Myrgioti et al. (2013)
#15	It presents a conceptual model and an operational ontology (implemented in OWL) that addresses UI interface and HCI process under the WIMP user interface perspective. Includes concepts such as User, Interaction, Interaction Style, Software, User Action, System Response. It is intended to be used in requirements recovery.	Zamzami et al. (2013)
#16	It presents a conceptual model that addresses ergonomics, UI design and evaluation. It includes concepts such as HCI Ergonomics, User Evaluation, Interface Evaluation, Design.	Elyusufi et al. (2014)
#17	It presents a user feedback ontology by means of a conceptual model that addresses characterization of user feedback on UI adaptation.	Mezhoudi and Vanderdonckt (2015)
#18	It presents a graphical web design ontology that addresses web UI structure and elements and web design property. Includes concepts such as Interface Element, Icon, Menu, Design Property, Content Design Property, Interface Design Property.	Bakaev and Gaedke (2016)
#19	It presents an interaction ontology under the craftswoman profile. It is implemented using OWL/SWRL and includes concepts such as Craftswoman, Input-Mode, Output-Mode, Input-Modality, Output-Modality, Input-Medium, Output-Medium, Sensory-Capacity.	Lebib et al. (2016)

Table 1 – *Continued from previous page*

Id	Description	Ref.
#20	It presents an ontological model of adaptive UI for people with disabilities. The ontology addresses concepts such as User, Disability, Device, Interface, Software Component, Hardware Component.	Kultsova et al. (2017)
#21	It presents a usability guideline ontology. The ontology is implemented using OWL and includes concepts such as GuidelineElement, ContentType, Guideline, UsabilityGuideline.	Robal et al. (2017)
#22	It presents an accessibility ontology by means of a conceptual model. It includes concepts such as User, Impairment, Disability, ActivityParticipation, SupportAssistance.	Romero Mariño et al. (2018)

We analyzed the ontologies taking the following characteristics of "beautiful ontologies" D'Aquin and Gangemi (2011) into account: (i) good domain coverage; (ii) being modular; (iii) being formally rigorous; (iv) implementing also non-taxonomic relations; (v) following an evaluation method; and (vi) reusing foundational ontologies. We did not include in our analysis characteristics that could not be evaluated based only on the papers' content (e.g., commercial impact). The characteristics of beautiful ontologies resulted from a study that analyzed the most varied and useful quality criteria to evaluate ontologies and selected the ones more relevant to build an ontology that provides a reference conceptualization and can also support practical solutions. Despite quality criteria define a good ontology, being a good ontology does not mean to be a beautiful ontology. A beautiful ontology is one that reflects an elegant solution for modeling a problem and it is at the same time good (in terms of formal quality), usable and practicable D'Aquin and Gangemi (2011); Vrandečić (2009).

Table 2 presents an overview of the ontologies considering the criteria used in the study. Concerning *good domain coverage*, in the table, we marked all ontologies in the sense that each of them covers the domain portion it is intended to. As for *being formally rigorous*, we marked the ones that are represented in some degree of formalism, even if not very rigorous.

Table 2: Analysis of HCI-related ontologies

	"Beautiful ontology" criteria						**Focus of this paper**
Ontology	good domain coverage	being modular	being formally rigorous	implementing also non-taxonomic relations	following an evaluation method	reusing foundational ontologies	addressing HCIO phenomenon
#01	✓						✓
#02	✓	✓	✓	✓	✓		
#03	✓		✓	✓	✓		
#04	✓	✓	✓	✓	✓		
#05	✓	✓	✓	✓	✓		✓ (system)
#06	✓			✓	✓		✓ (system)
#07	✓	✓	✓		✓		✓ (system)
#08	✓		✓	✓			
#09	✓	✓		✓			✓ (gesture)
#10	✓	✓		✓	✓		
#11	✓	✓		✓	✓		✓ (system)
#12	✓	✓	✓	✓	✓		✓ (user)
#13	✓		✓	✓	✓		✓ (user)
#14	✓	✓	✓	✓	✓	✓	✓ (haptic)
#15	✓	✓	✓	✓	✓		✓
#16	✓	✓			✓		
#17	✓						
#18	✓	✓		✓			✓ (system)
#19	✓		✓	✓	✓		
#20	✓	✓	✓	✓	✓		
#21	✓		✓	✓	✓		
#22	✓	✓	✓	✓	✓		

Since our focus in this paper is on the HCI phenomenon, next, we analyze the ontologies #01, #05, #06, #07, #09, #11, #12, #13, #14, #15 and #18. Concerning *domain coverage*, #05, #06, #07, #11 and #18 address the HCI system part of the interaction focusing only on web UI aspects; #12 and #13 address the HCI user part by means of Persona. #01, #09, #14 and #15 describe the HCI phenomena. From these, #09 and #14 address both user and system, but focusing specifically on gesture and haptic interaction, respectively.

With respect to *modularity*, #01, #06, #13 are non-modular, while the others are modularized in some way: #05 and #15 are organized in modules; #18 consider classes hierarchy; #11 and #14 are divided into subsystems; #12 considers schemas; #09 is organized in dimensions; and #07 in abstraction levels.

Regarding *being formally rigorous*, ontologies have been represented with different degrees of formalism. Most ontologies are presented as a graphical conceptual model (#06, #09, #11, #12, #13, #15, #18) and/or as an artifact implemented in a formal language, such as OWL or logics (#05, #07, #12, #13, #14, #15). The exception is #01, which is presented only in natural language, and thus cannot be considered as being formally rigorous. Only #14 define axioms in the form of rules.

As for *ontology evaluation*, the ontologies were evaluated through two main approaches: task-based (#05, #07, #11, #12, #13, #14, #15) and data-driven (#06, #14) evaluation. In a data-driven evaluation, the ontology is compared to existing data about the problem domain to which the ontology refers Brank et al. (2005). For example, #14 presents a set of instances that concern haptic interaction with a specific haptic device. In a task-based evaluation, the ontology is used in some kind of application or task, and the outputs of the application, or its performance on the given task, are used to evaluate the ontology Brank et al. (2005). Five ontologies were implemented and used in a software application (#05, #07, #11, #13, #14). Three ontologies (#01, #09, #18) were not evaluated.

Finally, concerning *implementing also non-taxonomic relations*, all the ontologies consider both taxonomic and non-taxonomic relations, although some works do not clearly represent them (such as #01 and #07).

Regarding *reusing foundational ontologies*, only #14 reuses a foundational ontology.

Summarizing, the analyzed ontologies cover different but related HCI aspects and most of the analyzed ontologies did not satisfy at least one of the considered characteristics of beautiful ontologies. We highlight that only one of the ontologies (#14) is grounded in a foundational ontology, but it is focused on a specific kind of interaction (haptic).

From the analyzed ontologies, we consider #01 and #15 the ontologies most related to our work (HCIO). Concerning #01, it is the oldest ontology addressing the HCI phenomenon we found. It has a good coverage of this phenomenon, centered on four main notions: Participant, Interaction, Purpose and Interface. Its top-level statement of the relationships between these notions is: "Any interaction takes place through one or more interfaces and involves two or more participants who each has one or more purposes for the interaction". Based on this statement, Storrs (1994) clarifies the four notions. The ontology is presented only in a textual form, in natural language. Concepts can be easily identified, but relations are often difficult to capture. In sum, this ontology lacks formality. HCIO, on the other hand, goes deeper into the computer side, describing what an interactive computer system is and how it participates in an interaction. Moreover, HCIO talks about types of user actions and how they are linked to events processed by machines. HCIO is grounded in a foundational ontology (UFO), modular (divided into three sub-ontologies) and presented as a conceptual model with axioms added to capture important constraints that are not possible to be captured by the graphical model.

Regarding #15, the ontology is divided into two modules. The Window Icon Menu Pointer User Interface ontology model (WIMP-UI) aims to capture the semantic meaning of user interface that is used by users of a software application in an interaction. The focus is on the graphical elements that compose a WIMP Interface. The User Interaction (USI) ontology model aims to represent the interaction between user and software. Comparing to HCIO, #15 deals with graphical UI elements that are not explicit in HCIO. On the other hand, it says nothing about how software produces responses, neither about what is an interactive computer system and how hardware and software are connected to it.

Given that the 22 ontologies found in the SLR address HCI aspects, some core concepts should be common to them. However, we noticed that there are inconsistencies among the conceptualizations, even concerning core concepts, which indicates a lack of a common conceptualization of the HCI phenomenon. Even concepts from ontologies covering the HCI phenomenon are not consensual. We believe that this is mainly because most of the ontologies were developed to solve specific problems for the purpose of practical applications, in specific contexts. Moreover, most of the ontologies do not present the concepts clearly. This demands interpretation from the reader, which is susceptible to misunderstanding. Finally, several core HCI phenomenon aspects are not covered by any of the aforementioned ontologies (e.g., how the system participates in the interaction and the different ways a user can participate in an interaction). Thus, we decided to develop a core ontology to describe the HCI phenomenon and serve as a reference to the HCI domain.

The main distinguishing feature of our ontology when contrasted to the other HCI ontologies is that it is a core reference ontology and has been developed taking characteristics of "beautiful ontologies" D'Aquin and Gangemi (2011) into account. In summary: (i) HCIO covers core aspects regarding the interaction phenomenon, providing explanation about the interaction itself and the involved parts; (ii) HCIO is a modular ontology, which favors understanding and reuse; (iii) HCIO is represented in a good level of formalism by means of conceptual models, axioms and textual descriptions; (iv) HCIO is a well-founded ontology grounded in UFO; and, finally, (v) HCIO was developed and evaluated by following SABiO method Falbo (2014), a well-established method used in several ontology development efforts.

4. Methodological Approach

The work addressed in this paper followed the Design Science Research (DSR) paradigm, which is an iterative process including three cycles Hevner (2007): Relevance Cycle, Design Cycle and Rigor Cycle. A Design Science Research project begins with the *Relevance Cycle*, which involves defining the problem to be addressed, the requirements and the criteria for evaluating the results. The problem addressed by this work involves the need for a core reference conceptualization of the HCI phenomenon, which can be used for communication, learning and interoperability purposes. Moreover, such core conceptualization can serve as a basis to develop other ontologies addressing HCI aspects, contributing to the development of a comprehensive and consistent conceptualization of the HCI domain.

The problem was identified by analyzing the literature (mainly from the SLR results) and from practical experiences of the authors when dealing with different HCI standards and other knowledge sources. Considering the successful use of ontologies in several domains, we decided to investigate (through a SLR) how ontologies have been used in the HCI domain and also how HCI ontologies have been developed. As we discussed in Section 3, the SLR results revealed that although there are several HCI ontologies, there are inconsistencies among their conceptualizations, even in HCI core concepts. The ontologies have been developed to solve specific problems and are biased in the applications they support. This hampers ontology reuse and integration to support solutions to more comprehensive HCI problems (e.g., involving different HCI subdomains) and contributes to semantic conflicts. Thus, we noticed that developing HCI ontologies as a network could help address these issues. In an ontology network, core ontologies provide the core conceptualization that spans across several subdomains Borges Ruy et al. (2016) and, thus, is reused to develop the domain ontologies, keeping consistency in the conceptualization provided by the network as a whole. Since we did not found ontologies properly covering HCI core aspects and that such ontology is needed to support a proper understanding of the HCI phenomenon and for developing the network, we decided to develop HCIO. Representing knowledge in a structured way promotes knowledge organization, making domain assumptions explicit and sharing an understanding of the information structure; provides a domain specification as a result of an intensive domain analysis; provides a clear separation of operational and domain knowledge and enables knowledge reuse Feilmayr and Wöß (2016). We were also motivated by some knowledge-related problems that we noticed when dealing with the intersection of HCI with other related areas. For example, when working with HCI and Software Engineering professionals, we noticed that people with different backgrounds (e.g., industrial design, web design, software engineering) had different understandings of the same HCI concept. For instance, to talk about the design of an interactive computer system, it is necessary to understand (and agree on) what an interactive computer system is and how and why user and system interact, so that it is possible to design HCI solutions to meet the user requirements. The main stakeholders that can benefit from HCIO are HCI researchers, professors and professionals that work with HCI or its intersection with other areas, which can use HCIO conceptualization to better understand the HCI phenomenon and for teaching, learning and communications purposes. Ontology engineers, in turn, can use HCIO as core ontology to build other ontologies. Once the ontologies are built, people interested in producing knowledge-based or interoperability solutions using the ontologies can be ultimately benefited from HCIO.

As requirements to develop the ontology, we considered some characteristics of "beautiful ontologies" D'Aquin and Gangemi (2011): (R1) the ontology must cover core aspects regarding the interaction phenomenon; (R2) the ontology must be modular; (R3) the ontology must be formally rigorous; (R4) the ontology must be ground in a well-founded ontology; and (R5) the ontology must be developed by following an appropriate Ontology Engineering method. We also considered some use requirements: (R6) the ontology must serve as a reference ontology to describe the HCI phenomenon (covering the different types of interaction) and help understand it; (R7) the ontology must serve as a basis to develop, integrate and align other HCI ontologies; and (R8) the ontology must support semantic-based solutions. In addition to the requirements to be met by the ontology, we defined the following criteria to evaluate it: (C1) the ontology elements (concepts, relation and axioms) must be the ones sufficient and necessary to cover the scope defined by means of competency questions; and (C2) the ontology must be able to represent real-world situations.

The *Design Cycle* involves developing and evaluating artifacts or theories to solve the identified problem. Therefore, in this cycle, we developed and evaluated HCIO. To meet R1, we considered knowledge from the literature and domain experts. Regarding R2, we decomposed HCIO into three sub-ontologies. To satisfy R3, we defined HCIO by means of conceptual models, axioms and textual descriptions. Concerning R4, we grounded HCIO in UFO Guizzardi (2005). As for R5, we followed SABiO Falbo (2014). Concerning use requirements, to satisfy R6, we investigated the literature about HCI to properly cover this phenomenon and we had two HCI experts evaluating the conceptualization aiming to ensure that the HCI phenomenon was properly represented. In order to verify if the ontology helps understand the HCI phenomenon, we used HCIO to teach HCI to undergraduate students. Satisfying R1 to R5 contributed to meet R7. Thus, we used HCIO as the core ontology of a HCI ontology network (HCI-ON) Costa et al. (2020) in which HCIO was reused to develop new ontologies (e.g., HCI Design Ontology, HCI Evaluation Ontology, Cognitive HCI Ontology, Semiotic HCI Ontology), integrate and align existing ontologies. Finally, as for R8, we implemented an operational version of HCIO and we have applied it in a computational solution to evaluate user experience in an immersive application. To evaluate HCIO considering C1 and C2, we performed verification and validation activities, as suggested in SABiO Falbo (2014). To achieve the version of HCIO presented in this paper, we performed three design cycles involving HCIO development and evaluation.

Finally, the *Rigor Cycle* refers to using and generating knowledge. 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). The main contribution for the knowledge base is HCIO. As secondary contributions, we have the analysis of the 22 ontologies found in the literature, the initial version of the HCI ontology network (HCI-ON) Costa et al. (2020) and the design of the computational solution to evaluate user experience.

5. Human-Computer Interaction – Scenarios of use

In this section, we describe two scenarios of use of human-computer interaction, which are used in the next section to exemplify (i.e., instantiate) HCIO concepts. The cases were performed in the real-world and here they are presented by means of storyboards using fictitious names. In the first case, a person (John) interacts with his desktop computer to quote flights prices on the Internet. In the second case, a person (Rino) interacts with his smartwatch to monitor her performance in a run. Figure 4 illustrates the case where John quotes flights price on the Internet. John is a New Yorker student who intends to attend a conference in Rio de Janeiro, Brazil.

John's desktop computer is ready to start the interaction (1st picture), i.e., the computer is properly connected to the Internet and with Chrome running. John accesses Expedia's site[2] and faces the form "Search Flights" (2nd picture). He notices that some fields of the form appear filled, while others are blanked. John fills out the required fields (3rd picture). After that, John executes the search by clicking with the mouse the "Search" button, activating it (4th picture). John notices that Chrome is loading (5th picture) and, seconds after, he gets a new page with the list of flights and corresponding prices and he identifies the flight with the lowest price, achieving his goal (6th picture).

Figure 5 illustrates the case where Rino interacts with his smartwatch to monitor performance in a run. Every day, Rino runs for 30 minutes. Aiming to monitor his heart rate and performance, he interacts with his Apple watch (Figure 5, 1st picture).

Before starting to run, Rino presses the watch-crown that triggers Siri[3] (2nd picture), which asks "What can I help you with?" (3rd picture). Rino says "Start outdoor run" (4th picture) and the watch opens the

[2] https://www.expedia.com/Flights
[3] https://www.apple.com/siri/

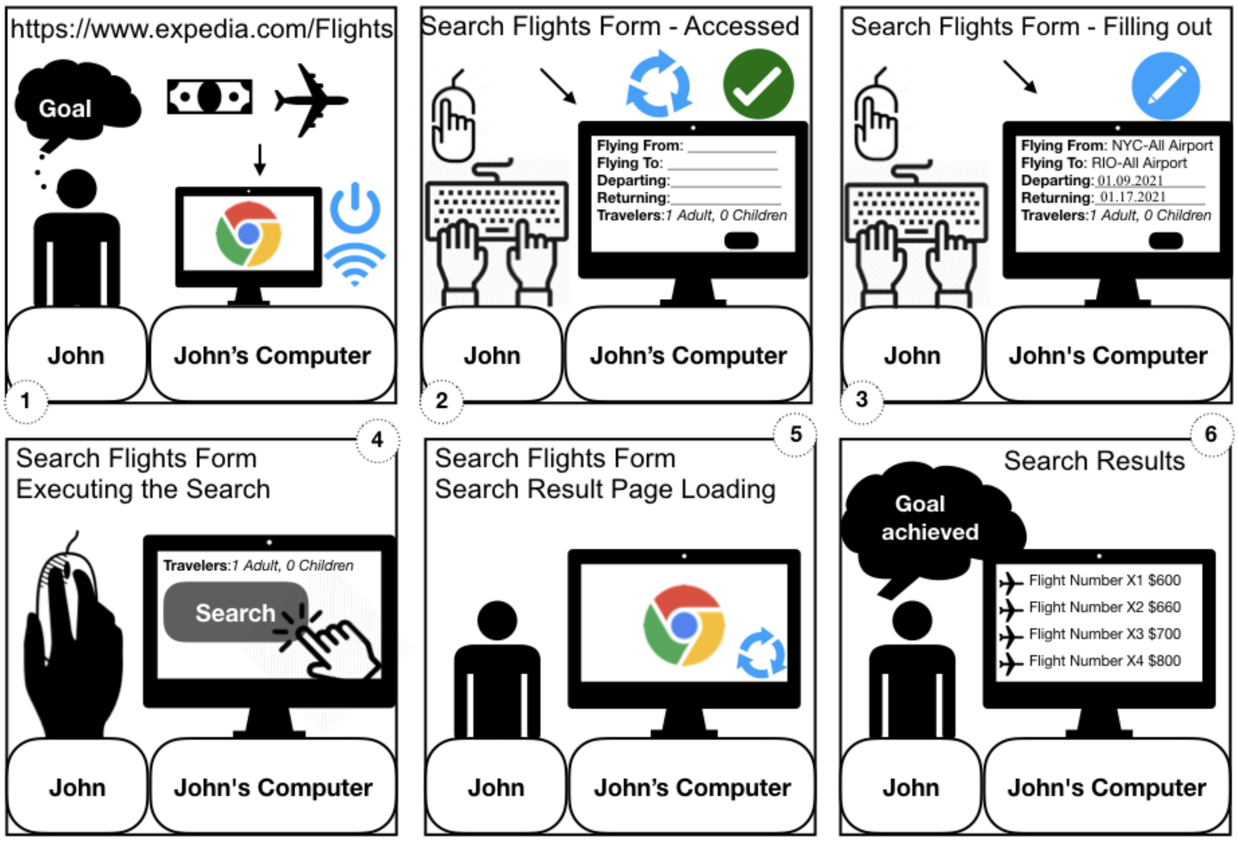

Figure 4: Storyboard: John searching the Internet to quote flights prices

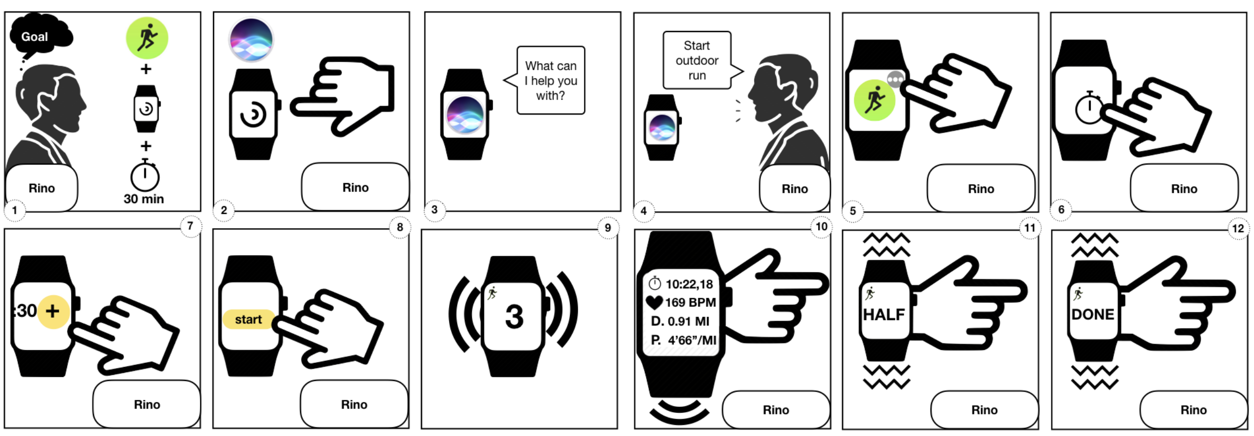

Figure 5: Rino monitoring his performance in a run

Activities app. To set the running time, Rino touches the widget [...] (5th picture) and then the time widget (6th picture). Rino touches the widget [+] until it reaches 30 minutes (7th picture) and then touches the start widget (8th picture). The watch shows a countdown (9th picture) and Rino starts running. After a few minutes, he turns his wrist to check his heart rate and performance (10th picture). He sees the following data: 10:22, 18 of running, heart rate at 169bpm, distance 0.91mi and pace 4'66"/mi. Rino continues running. After 15 minutes, Rino feels the watch vibrate, turns his wrist and sees a message informing that he has completed half of his way (11th picture). He continues running until he feels the watch vibrate again. He looks at the watch and sees a message informing that he has run 30 minutes and thus his goal has been achieved (12th picture).

6. Human-Computer Interaction Ontology – HCIO

The purpose of the Human-Computer Interaction Ontology (HCIO) is to establish an explicit and shared conceptualization of the HCI phenomenon, describing the main concepts involved in this phenomenon. The knowledge sources used to build HCIO include standards, such as ISO (2019), theories Norman (1986); De Souza (2005), models Abowd and Beale (1991); Hewett et al. (1992) and relevant literature in the HCI area, such as Carroll (2014), Sutcliffe (2014), Preece et al. (2015), Dix et al. (2004), Benyon (2010), Norman (2013), Schmidt (2000), Schmidt (2005), Krol et al. (2016), Fairclough (2009), Norman (2009), Fairclough and Gilleade (2014), Zander et al. (2018), Belkhiria and Peysakhovich (2020), Zander et al. (2010), Clites et al. (2018), Zander and Kothe (2011), Nielsen (1993), Oliveira et al. (2017), Rogers et al. (2011), Saffer (2010), Callan et al. (2016), among others. HCIO also includes knowledge obtained from the study of the SLR ontologies (we did not reuse the ontologies themselves because, as we explained in Section 3, they have several limitations). It is worth pointing out that understanding the different types of interaction (see Section 2.1) was very important to ensure that HCIO is capable of representing all of them. When we built the first version of HCIO, we did not consider different types of interaction. As a result, that version was not able to represent several interaction scenarios that occur in the real world. Domain experts evaluated the ontology and pointed out these problems. To solve them, we incorporated to HCIO knowledge of different types of interaction.

As we said before, HCIO is grounded in UFO Guizzardi (2005) and reuses concepts from the core System and Software Ontology (SysSwO) Bringuente et al. (2011); Duarte et al. (2018). Given that a human-computer interaction involves communication between user and interactive computer system, HCIO is composed of sub-ontologies to deal with the interaction participants and with the interaction itself. The *Interactive Computer System* sub-ontology addresses what an interactive computer system is and its elements, including the user interface. The *User* sub-ontology focuses on the user and its possible actions when interacting with an interactive computer system. Finally, the *HC Interaction* sub-ontology links concepts from the other two sub-ontologies to define what a human-computer interaction is. Figure 6 shows HCIO sub-ontologies and its relations with SysSwO and UFO. In the figure, dependency relations mean that the source ontology reuses concepts from the target ontology.

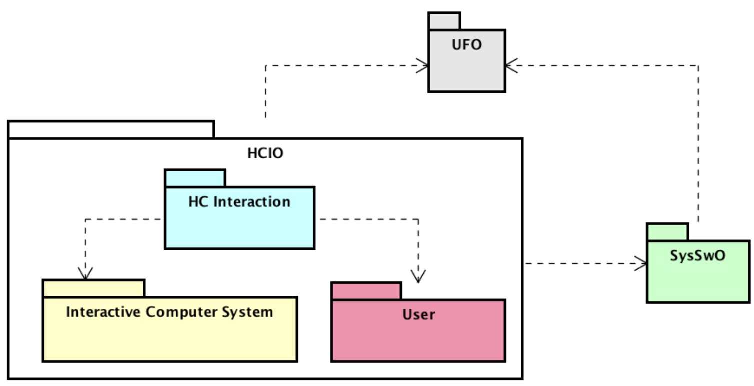

Figure 6: HCIO architecture

HCIO was developed by following SABiO Falbo (2014), which prescribes that the ontology scope must be defined by means of competency questions, i.e., questions the ontology must be able to answer and are used as a basis to develop the ontology conceptual model. To that end, we considered the basis of HCI previously presented (see Section 2) saying that **human-computer interaction** is the communication process that occurs during the use of an **interactive computer system** and that involves **user** actions. As a consequence, to cover the HCI phenomenon we should focus on concepts related to interactive computer system, user and HCI itself. Table 3 shows HCIO competency questions (CQ). CQ01 to QC04 help understand what an interactive computer system is and its software and hardware elements. QC05 to QC09 are to understand user, the actions he/she performs when interacting with interactive computer systems and what causes user to interact with the system. CQ10 to Q14 refer to the human-computer interaction itself, addressing user inputs, system outputs, and actions and interpretations involved in the interaction. QC15 is about goal achievement, which is relevant when the user interacts with the system aiming to achieve a certain goal.

Table 3: HCIO Competency Questions

Focus	Id	Description
Interactive Computer System	CQ01	What is an interactive computer system?
	CQ02	What is an interactive software system?
	CQ03	What is a complex interactive computer system?
	CQ04	What does make up the user interface of an interactive computer system?
User	CQ05	What is a User?
	CQ06	How can a user interact in human-computer interactions?
	CQ07	Considering intentionality, how can a user interact with an interactive computer system?
	CQ08	Why does a user intentionally interact with an interactive computer system?
	CQ09	What does make up a complex user participation?
HCI	CQ10	What is a human-computer interaction?
	CQ11	Considering the human-computer interaction, how can a user participation cause another user participation?
	CQ12	How is a user input processed by an interactive computer system?
	CQ13	How does a user receive an output from an interactive computer system?
	CQ14	How is a user input processed by an interactive computer system and how is the corresponding output presented to him/her?
	CQ15	How does a user evaluate if his/her goal was achieved in a human-computer interaction?

Next, we present the HCIO sub-ontologies. In the conceptual models, we kept the colors used in Figure 6 to identify the concept source. In the models' description we refer to the scenarios of use presented in Section 5 to exemplify HCIO concepts.

6.1. Interactive Computer System Sub-Ontology

This sub-ontology aims to answer CQ01 to CQ04 and, thus, focuses on defining interactive computer system. It is mainly an extension of SysSwO (see Section 2.2.2). Figure 7 shows the conceptual model of the Interactive Computer System sub-ontology. In the figure, we used dotted lines to separate the ontologies into layers. At the top, we have UFO, providing the general foundation. At the center, there is SysSwO, containing core concepts related to computer systems. At the bottom, there are HCIO concepts, grounded in UFO or specialized from SysSwO (which is also grounded in UFO). Relations that ground SysSwO or HCIO concepts in UFO (i.e., specializations) are shown in blue. In the text, we use *italics* to refer to UFO concepts, *underline italics* to SysSwO concepts and ***bold italics*** to HCIO concepts. In the models' description, we present some of the axioms defined to address constraints not captured in the models.

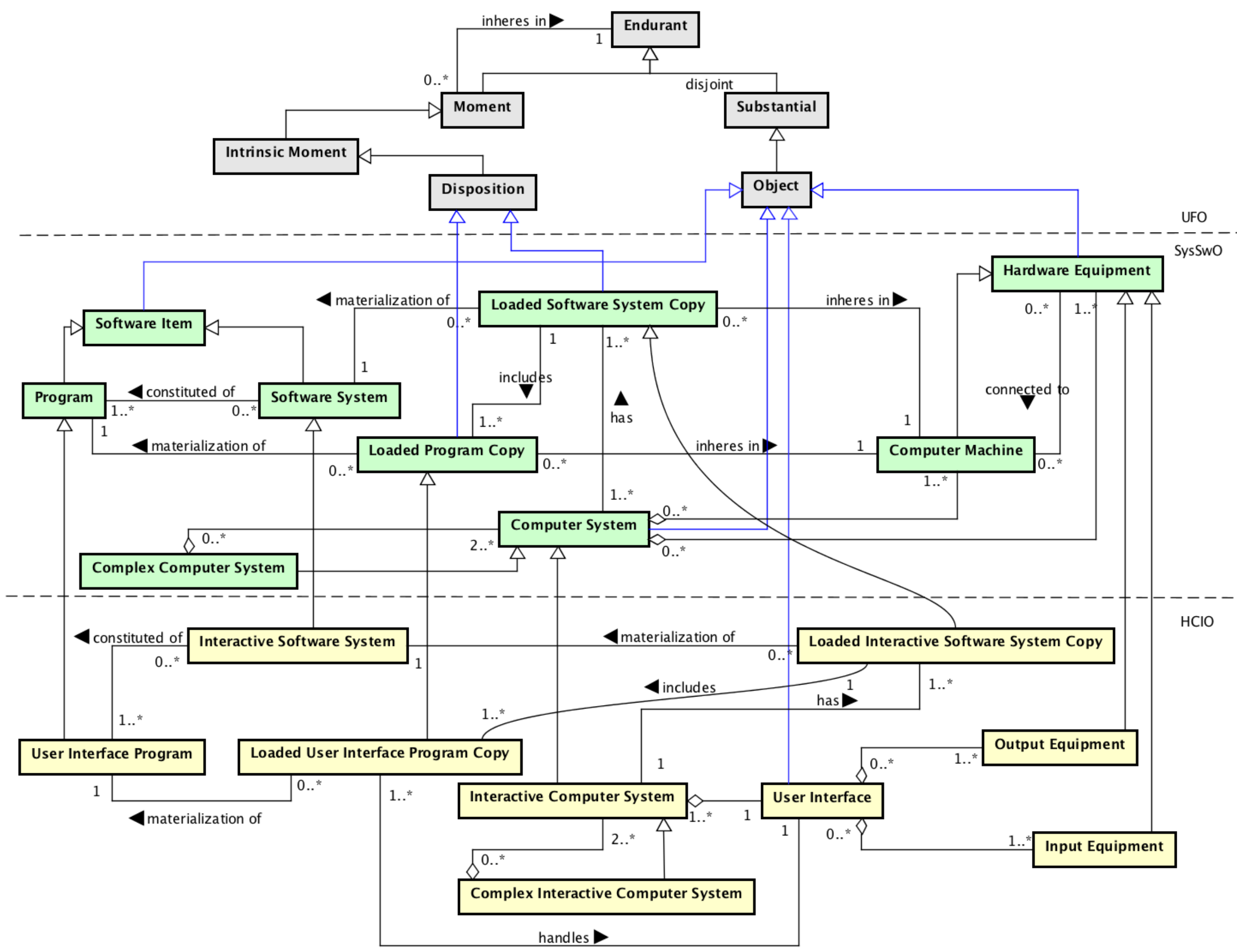

Figure 7: Interactive Computer System Sub-ontology

Interactive Computer System is a subtype of *Computer System*, and like the last, it combines hardware and software. Concerning hardware, an ***Interactive Computer System*** is (the *Computer System*) composed of a set of *Computer Machines* and peripheral devices (*Hardware Equipment*) connected to them. The striking feature of an ***Interactive Computer System*** is that it has a ***User Interface***, a complex *Object* that is composed of ***Input Equipment*** and ***Output Equipment*** connected to the *Computer Machine*. If an ***Output Equipment*** is part of a ***User Interface*** of an ***Interactive Computer System***, then this ***Interactive Computer System*** should be constituted of the *Computer Machine* that connects the ***Output Equipment***. The same applies to ***Input Equipment***. These constraints are addressed by the following axioms:

A 1. ∀ *oeq: Output Equipment, ui: User Interface, ics: Interactive Computer System partOf (oeq, ui)* ∧ *partOf (ui, ics)* → *(*∃ *cm: Computer Machine partOf (cm, ics)* ∧ *connectedTo (oeq, cm))*

A 2. ∀ *ieq: Input Equipment, ui: User Interface, ics: Interactive Computer System partOf (ieq, ui)* ∧ *partOf (ui, ics)* → *(*∃ *cm: Computer Machine partOf (cm, ics)* ∧ *connectedTo (ieq, cm))*

When a *Hardware Equipment* is both an ***Input Equipment*** and an ***Output Equipment*** it is said an ***IO Equipment***. Thus:

A 3. ∀ *he: Hardware Equipment Input Equipment(he)* ∧ *OutputEquipment(he)* → *IOEquipment (he)*

Regarding software, an ***Interactive Computer System*** has a set of *Software Systems* (***Interactive Software System***) loaded in its *Computer Machines* (***Loaded Interactive Software System Copies***). The programs that constitute these systems are instances of *Program*. Some of them deal with aspects related to the user interface and, thus, are instances of ***User Interface Program***. Thus, an ***Interactive Computer System*** has a ***User Interface*** and the copies of the programs loaded in its computer (***Loaded User Interface Programs Copy***) that handle its ***User Interface***. The following axiom applies:

A 4. ∀ *ui: User Interface, ics: Interactive Computer System, lissc: Loaded Interactive Software System Copy partOf (ui, ics)* ∧ *has (ics, lissc)* → ∃ *luipc: Loaded User Interface Program Copy includes (lissc, luipc)* ∧ *handles (luipc, ui)*

Interactive Computer Systems can form another ***Interactive Computer System***, which is said a ***Complex Interactive Computer System***.

In John's case (Figure 4), the ***Interactive Computer System*** is composed of John's desktop computer (a *Computer Machine*) and its peripheral devices (*Hardware Equipment*), Google's and Expedia's servers (*Computer Machines*) and other *Hardware Equipment* connected to them, plus the software systems loaded in those machines, such as the copies of the operating system and Chrome browser running in John's computer, as well as the copy of the Expedia's system running in one of the Expedia's servers (***Loaded Interactive Software System Copies***). Chrome, Expedia's Travel Booking System and the Operating System are instances of ***Interactive Software System***.

The ***User Interface*** is composed of, among others, the mouse, the keyboard (***Input Equipment***) and the monitor (***Output Equipment***) connected to John's desktop computer (*Computer Machine*), and has its elements handled by the copies of the programs loaded in that computer (***Loaded User Interface Programs***), such as the loaded copy of Chrome's program responsible for displaying Chrome's graphical window.

In Rino's case (Figure 5), the ***Interactive Computer System*** is the *Computer System* composed of Rino's Apple Watch computer (a *Computer Machine*) and its attached devices (*Hardware Equipment*), plus the software system loaded in the machine, such as the copies of the IOS, Siri and Activity (***Loaded Interactive Software System Copies***).

IOS, Siri and Activity (***Interactive Software System***) are constituted by programs and some of them handle user interface elements/widgets. The ***User Interface*** is composed of the microphone, the crown, the sensors (***Input Equipment***), the speaker, the taptic engine (***Output Equipment***) and the touch screen (***IO Equipment***), among others.

6.2. User Sub-Ontology

This sub-ontology focuses on actions users perform in the context of a human-computer interaction and aims to answer the competence questions CQ05 to CQ09. Figure 8 shows the conceptual model of the User sub-ontology.

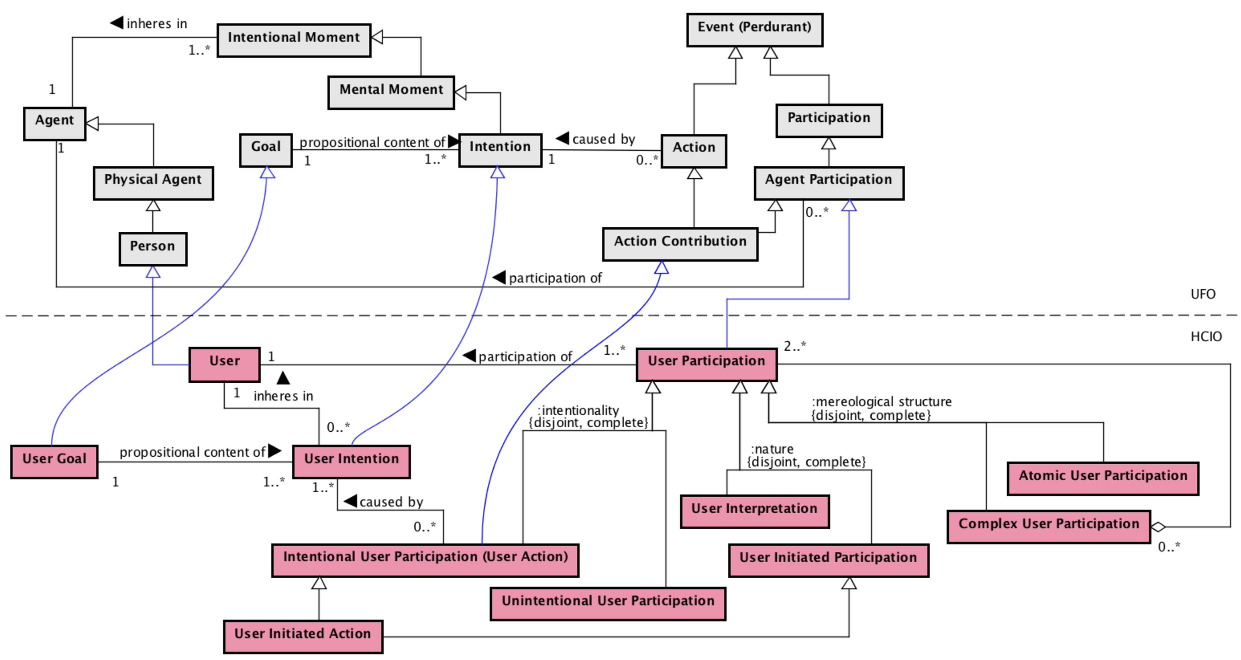

Figure 8: User Sub-ontology

User is the role played by a *Person* that participates in a human-computer interaction. Such participation is said a ***User Participation***, which can be either intentional (***Intentional User Participation*** – or simply ***User Action***) or unintentional (***Unintentional User Participation***). We adopted the terms intentional and unintentional referring respectively to the explicit and implicit interaction discussed in Section 2.1.

In terms of UFO, ***User Participation*** is an *Agent Participation*. ***Intentional User Participation (User Action)*** is an *Action Contribution*, i.e., an intentional participation of a ***User***. ***Intentional User Participations (User Actions)*** are caused by *Intentions* (***User Intentions***) that inhere in the ***User***. As an *Intention*, ***User Intention*** has a *Goal* (more specifically a ***User Goal***) as its propositional content. ***User Action***, as an intentional participation, is performed by the ***User*** in order to achieve a ***User Goal***. ***User Actions*** performed by a ***User*** can only be caused by ***User Intentions*** that inhere in that ***User***. This constraint is addressed by the following axiom:

A 5. ∀ *user: User, ua: Intentional User Participation (User Action), uint: User Intention participationOf (ua, user)* ∧ *causedBy (ua, uint)* → *inheresIn (uint, user)*

In John's case (Figure 4), he is the ***User***. He has the intention of quoting air tickets (***User Intention***) to identify the one with lowest price (***User Goal***). This intention caused him to intentionally search the Internet and fill out the required form fields (***Intentional User Participation (User Action)***).

In Rino's case (Figure 5), he (***User***) has the intention of monitoring his performance (***User Intention***) by monitoring heart rate, time, distance and velocity in a run (***User Goal***). Thus, he intentionally sets his smart watch to do so. For example, Rino says "Start outdoor run" to Siri (***Intentional User Participation (User Action)***). When interacting with his smart watch, Rino also acts unintentionally. For example, he unintentionally feels the watch vibrate (***Unintentional User Participation***). In addition, Rino unintentionally inputs data from his pulse (***Unintentional User Participation***).

Considering its mereological structure, a ***User Participation*** can be an ***Atomic User Participation*** or a ***Complex User Participation***, which is composed of others ***User Participations***. In a ***Complex User Participation***, all the ***User Participations*** are participations from the same ***User***. Thus:

A 6. ∀ *cup: Complex User Participation, user: User, up: User Participation participationOf (cup, user)* ∧ *partOf (up, cup)* → *participationOf (up, user)*

In Rino's case, each touch in the [+] widget to set the time for the run can be considered an ***Atomic User Participation***. Thus, the set of 30 touches composing the whole act of set 30 minutes would be a ***Complex User Participation*** composed of 30 ***Atomic User Participations*** of Rino.

In another classification, which considers the nature of participations and is orthogonal to the ones discussed above, ***User Participations*** are classified into two disjoint types: ***User Initiated Participation*** and ***User Interpretation***. ***User Initiated Participation*** refers to an act performed by the user making an input in the system. ***User Interpretation***, in turn, regards interpreting a state of the system. When a ***User Initiated Participation*** is intentional, it is said a ***User Initiated Action***.

In John's case, he types the Expedia URL, making an input in the system. The act of typing the Expedia URL is a ***User Initiated Participation***. More than that, it is a ***User Initiated Action***, since John intentionally types the Expedia URL. When John looks at the monitor and perceives that the Expedia site was accessed, he interprets the state of the system. Thus, this act is a ***User Interpretation***. Since he was expecting to face Expedia site, his ***User Interpretation*** is also an ***Intentional User Participation***.

In Rino's case, the act of Rino unintentionally inputting data from his heartbeat is a ***User Initiated Participation***. On the other hand, his intentional act of setting 30 minutes for the run is a ***User Initiated Action*** (i.e., an intentional ***User Initiated Participation***). When Rino feels the watch vibrate, he interprets a state of the system (***User Interpretation***). Since he was not expecting the watch to vibrate, his ***User Interpretation*** is also an ***Unintentional User Participation***.

6.3. HC Interaction Sub-Ontology

Figure 9 presents the conceptual model of HC Interaction sub-ontology. This sub-ontology links concepts from the other two sub-ontologies to define what a human-computer interaction is. It aims to answer the competence questions CQ10 to CQ15.

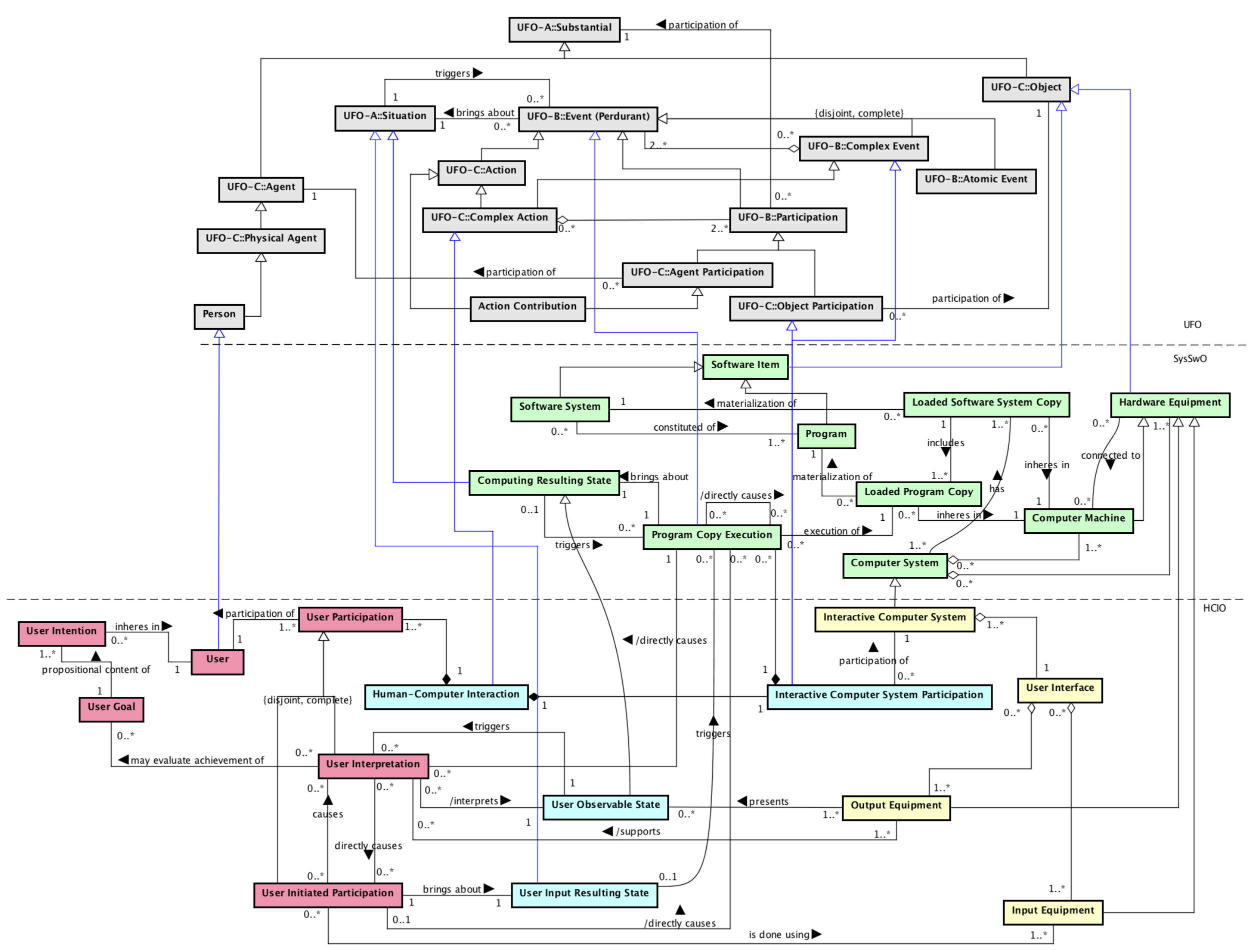

Figure 9: HC Interaction Sub-ontology

A ***Human-Computer Interaction*** is an interaction between a ***User*** and an ***Interactive Computer System***. Thus, a ***Human-Computer Interaction*** is composed by a ***User Participation*** and an ***Interactive Computer System Participation***, indicating the events performed by both parties in a specific interaction. For example, when John (***User***) types the Expedia URL and the system (***Interactive Computer System***) shows the Expedia site, we have an interaction (***Human-Computer Interaction***) in which John participates by inputting data into the system and interpreting the system response (***User***

Participation), and the system participates by receiving John's input, processing it, and showing Expedia site (***Interactive Computer System Participation***).

As said before (see User Sub-ontology), in terms of UFO, ***User Participation*** is an *Agent Participation*. ***Interactive Computer System Participation***, in turn, is a *Complex Object Participation*. It is a *Complex Event*, since it aggregates all events performed by the ***Interactive Computer System*** in the context of a single ***Human-Computer Interaction***. It is an *Object Participation* because, being the ***Interactive Computer System*** an *Object*, its participation is always unintentional.

A ***User Initiated Participation*** is performed using one or more ***Input Equipment***. As a result of a ***User Initiated Participation***, a ***User Input Resulting State*** is achieved. ***User Input Resulting State*** is a *Situation* representing the data entered by the user before any program execution. This situation triggers *Program Copy Executions*. Moreover, we can say that ***User Initiated Participation*** directly causes *Program Copy Execution*. *Program Copy Execution* brings about a *Computing Resulting State* (internal computer state), which, in turn, can trigger other *Program Copy Executions*. Thus, a *Program Copy Execution* can directly cause new *Program Copy Executions*.

In John's case, using the mouse (***Input Equipment***) and keyboard (***Input Equipment***), he types the Expedia URL in the Chrome navigation bar and presses the enter key (***User Initiated Participation***). The situation resulting from the input action (***User Input Resulting State***) triggers the execution of programs that search for the Expedia site and show it (*Program Copy Execution*). In this context, the execution of the program that searches for the Expedia site produces a *Computing Resulting State* (e.g., the state in which the Expedia site is found) that causes the execution of the program that shows the Expedia site in the navigator.

Some *Program Copy Executions* can bring about a special type of *Computing Resulting State*, the one that is perceivable by user, said ***User Observable State***. A ***User Observable State***, thus, triggers ***User Interpretation***. Therefore, we say that *Program Copy Execution* directly causes ***User Interpretation***. In this context, the following axiom applies:

A 7. $\forall$ *ui: User Interpretation, uos: User Observable State, pce: Program Copy Execution, uirs: User Input Resulting State, uip: User Initiated Participation triggers (uos, ui)* $\wedge$ *bringsAbout (pce, uos)* $\wedge$ *triggers (uirs, pce)* $\wedge$ *bringsAbout (uip, uirs)* $\rightarrow$ *causes (uip, ui)*

It is important to say that (A7) does not constrain who are the users involved in the ***User Initiated Participation*** and in the ***User Interpretation***, i.e., the ***User*** in the ***User Initiated Participation*** may be different from the one involved in the ***User Interpretation***.

A ***User Observable State*** triggers ***User Interpretation*** and is presented by one or more ***Output Equipment***. Thus, ***Output Equipment*** supports ***User Interpretation***, i.e.:

A 8. $\forall$ *oe: Output Equipment, uos: User Observable State, ui: User Interpretation presents (oe, uos)* $\wedge$ *triggers (uos, ui)* $\rightarrow$ *supports (oe, ui)*

In John's case, the execution of programs triggered by the URL entered by him (***User Input Resulting State***) brings about the situation where the Expedia site is shown (***User Observable State***) in John's monitor (***Output Equipment***). John interprets the information showed in Expedia site (***User Interpretation***) and fills out the required fields to quote air tickets (***User Initiated Participation***).

In a ***Human-Computer Interaction***, both the ***Input*** and ***Output Equipment*** involved in the interaction should be part of the ***User Interface*** of the ***Interactive Computer System*** that participates in the interaction.

Since ***User Initiated Participation*** typically leads to some processing inside the system that results in an output to the user who can interpret it, we say that ***User Initiated Participation*** (indirectly) causes ***User Interpretation***. ***User Interpretation*** may cause the ***User*** to act again. Thus, ***User Interpretation*** may directly cause (in terms of UFO) ***User Initiated Participations***. In the example above, the act of John entering Expedia URL (***User Initiated Participation***) leads to programs execution that brings about the exhibition of Expedia site (***User Observable State***). John interprets the Expedia site (***User Interpretation***) and acts again, filling out the required fields (***User Initiated Participation***). Thus, the act of John entering Expedia URL indirect caused him to interpret Expedia site. This interpretation, in turn, directly caused him to fill out the required fields.

When a ***User Participation*** is intentional, it means that the ***User*** performed it considering some goal (***User Goal***). Interpretations performed by the user (***User Interpretation***s) may evaluate ***User Goal*** achievement. This is a case of goal-driven behavior, in which a cycle of actions (***User Participation***) are repeated until the user achieves his/her goal (***User Initiated Participation*** (indirectly) causes ***User Interpretation*** and ***User Interpretations*** (directly) causes ***User Initiated Participations***).

For example, when John types the Expedia URL (***User Initiated Participation***), he has the goal of accessing Expedia site (***User Goal***). When he sees that he has accessed the Expedia site (***User Interpretation***), he notices that he has achieved his goal in that particular participation. If we consider the whole scenario in John's case, his goal is to identify the air ticket with the lowest price. With this goal in mind, he interacts with the system. He performs several actions (e.g., types de Expedia URL, fills up the form, clicks bottom) (***User Initiated Participation***). After each action, he perceives (***User Interpretation***) that his goal was not achieved and keeps acting until the system shows flight options and he identifies the one with the lowest price, achieving his goal. In this scenario, John's input actions (***User Initiated Participation***) caused him to interpret the system outputs (***User Interpretation***). John's interpretations (***User Interpretation***), in turn, caused him to act again (***User Initiated Participation***), until he has achieved his goal.

The HC Interaction sub-ontology main purpose is to define what human-computer interaction is. In view of what was discussed, in summary, a ***Human-Computer Interaction*** is a *Complex Event* composed of

User Participation and ***Interactive Computer System Participation***. ***User Participation*** can involve both ***User Initiated Participations*** and ***User Interpretations***. ***Interactive Computer System Participation*** regards the set of *Program Executions* performed by the ***Interactive Computer System*** in the interaction.

6.4. HCIO Evaluation

To evaluate HCIO, we performed Ontology Verification & Validation (V&V) activities. Considering the guidelines proposed by SABiO Falbo (2014), HCIO was evaluated by using two evaluation approaches: *assessment by human approach* and *data-driven approach* Brank et al. (2005). In the first, we performed a verification activity by means of expert judgment, in which we checked if the concepts, relations and axioms defined in HCIO are able to answer the competency questions. Moreover, the ontology specification was peer-reviewed by two domain experts. In the second, since a reference ontology should be able to represent real-world situations, to validate HCIO, we instantiated its concepts and relations using data extracted from real cases. V&V activities were performed manually, considering the reference ontology.

As we explained in our methodological approach (see Section 4), to achieve the HCIO version shown in this paper, we performed three cycles involving development and evaluation. Each cycle resulted in a version of HCIO, which improved the previous one. After producing each version of HCIO, we evaluated it by performing V&V activities and submitting the ontology specification and the V&V results to the evaluation by domain experts. Based on the evaluation results, we improved the ontology and evaluated it again until we reached the current version. Some situations the domain experts pointed out as not properly covered by previous versions of HCIO include: a user can interact with the system without a goal in mind (e.g., the user can move his/her arm in a smart home and turn on the light without intending to do so); the same equipment can be used as input and output equipment at the same time (e.g., touch screens); more than one user can interact with the system at the same time (e.g., video games). Feedbacks like these helped us to improve the ontology. They also showed us that we needed to understand the different kinds of interaction (see Section 2.1) to develop an ontology able to represent different situations of the real-word. Next, we present some results of V&V activities related to the current version of HCIO.

6.4.1. Verification by Experts

For verifying HCIO, we started by manually checking if the concepts, relations and axioms defined in HCIO are able to answer its competency questions (CQs). This approach enabled us to check not only if the CQs were answered, but also whether there were irrelevant elements in the ontology, i.e., elements that do not contribute to answering any of the questions. Table 4 illustrates this verification process for HCIO, showing which elements of HCIO (concepts, relations, properties and axioms) answer the CQs. Concerning axioms, only the ones presented in this paper were included in the table. The table can also be used as a traceability tool, supporting ontology change management. Verification results showed that HCIO is able to answer the competency questions (i.e., the ontology addresses the established scope) and that it contains the sufficient and necessary elements to do so.

Table 4: HCIO verification against its competency questions

CQ Id	Description, Concepts and *Relations*	**Axioms**
CQ01	**What is an Interactive Computer System?** Interactive Computer System is *subtype of* Computer System that *has* User Interface and *includes* Loaded Interactive Software System Copy, which, in turn, is *subtype of* Loaded Software System Copy and is *materialization of* Interactive Software System	
CQ02	**What is an Interactive Software System?** Interactive Software System is *subtype of* Software System that *is constituted of* User Interface Loaded User Interface Program Copy is *subtype of* Loaded Program Copy that *handles* User Interface and is *materialization of* User Interface Program	A4
CQ03	**What is a complex interactive computer system?** Complex Interactive Computer System is *subtype of* Interactive Computer System that *is composed of* other Interactive Computer Systems	
CQ04	**What does make up the user interface of an interactive computer system?** Interactive Computer System *has* User Interface *constituted of* Output Equipment and Input Equipment, which are *subtype of* Hardware Equipment	A1, A2, A3
CQ05	**What is a user?** User is *subtype of* Person who performs (*participation of*) User Participation	
CQ06	**How can a user interact in human-computer interactions?** User Participation is the *participation of* a User User Initiated Participation and User Interpretation are *subtypes of* User Participation User Initiated Action is *subtype of* User Initiated Participation	
CQ07	**Considering intentionality, how can a user interact with an interactive computer system?** User Participation is the *participation of* a User Intentional User Participation (User Action) and Unintentional User Participation are *subtypes of* User Participation User Initiated Action is *subtype of* Intentional User Participation (User Action)	
CQ08	**Why does a user intentionally interact with an interactive computer system?** User Goal is the *propositional content of* User Intention that *inheres in* User Intentional User Participation (User Action) is *subtype of* User Participation *caused by* User Intention	A5
CQ09	**What does make up a complex user participation?** Atomic User Participation and Complex User Participation are *subtypes of* User Participation Complex User Participation *is composed of* other User Participation	A6
CQ10	**What is a human-computer interaction?** Human-Computer Interaction *is constituted of* User Participation and Interactive Computer System Participation	
CQ11	**Considering the human-computer interaction, how can a user participation cause another user participation?** User Initiated Participation *causes* User Interpretation that *directly causes* User Initiated Participation	
CQ12	**How is a user input processed by an interactive computer system?** User Initiated Participation *brings about* User Input Resulting State that *triggers* Program Copy Execution	
CQ13	**How does a user receive an output from an interactive computer system?** Output Equipment *presents* User Observable State	
CQ14	**How is a user input processed by an interactive computer system and how is the corresponding output presented to him/her?** User Initiated Participation *is done using* Input Equipment and *brings about* User Input Resulting State that *triggers* Program Copy Execution. Thus, User Initiated Participation *directly causes* Program Copy Execution	A7, A8

Table 4 – *Continued from previous page*

CQ Id	Description, Concepts and *Relations*	Axioms
	Program Copy Execution *brings about* User Observable State that is *subtype of* Computing Resulting State. User Observable State *triggers* User Interpretation that *interprets* User Observable State. Thus, Program Copy Execution *directly causes* User Interpretation. Output Equipment *presents* User Observable State. Thus, Output Equipment *supports* User Interpretation	
CQ15	**How does a user evaluate if his/her goal was achieved in a human-computer interaction?** User Interpretation *may evaluate achievement of* User Goal	

6.4.2. Validation

Concerning ontology validation, the ontology should be able to properly represent real-world situations Falbo (2014). Based on that, we instantiated the ontology using data extracted from the John and Rino's cases. When describing HCIO conceptual models, we included some instances from these cases as examples. Next, in Table 5, we present a summary containing some instances extracted from Rino's case (Figure 5). The complete instantiation of both cases and the HCIO dictionary of terms are available at Costa et al. (2020a). Although in this paper we instantiate HCIO considering only the two cases described in Section 5, during validation we also considered several other cases (e.g., gesture and haptic interaction, interaction with a smart house, physiological interaction, interaction with a collaborative system) to ensure that HCIO is able to represent them. The successful instantiation of HCIO with data coming from real cases gave us indications of the appropriateness of the proposed ontology as a reference model.

Table 5: HCIO Instantiation - Rino's case

Interactive Computer System Sub-ontology	
Interactive Computer System	Rino's Apple Watch
Computer Machine	Apple Watch computer
User Interface	The whole Apple watch features like the crown, the touch screen, the microphone, the sensors, the speaker, the taptic engine
Input Equipment	Apple watch touch screen, microphone, crown, sensors
Output Equipment	Apple watch touch screen, speaker, taptic engine
Interactive Software System	IOS, Siri, Activity
Loaded Interactive Software System Copy	Copies of the IOS, Siri and Activity loaded in Rino's Apple Watch
User Interface Program	Programs constituting IOS, Siri and Activity that handle User Interface elements
Loaded User Interface Program Copy	Copies of the programs constituting IOS, Siri and Activity that handle User Interface elements loaded in Rino's Apple Watch
User Sub-ontology	
User	Rino
User Intention	Monitor performance
User Goal	Monitoring heart rate, time, distance and velocity in a run
User Participation (UP)	UP1 = Rino presses the Apple watch crown (Fig. 5, 2nd picture) UP2 = Rino sees a message from Siri on the watch screen (Fig. 5, 3rd picture) UP3 = Rino says "Start outdoor run" (Fig. 5, 4th picture) UP4 = Rino sees the Activity app opened (Fig. 5, 5th picture) UP5 = Rino touches the [...] widget (Fig. 5, 5th picture) UP6 = Rino sees a new screen from the Activity app on the watch (Fig. 5, 6th picture) UP7 = Rino touches the time widget (Fig. 5, 6th picture) UP8 = Rino sees a new screen from the Activity app on the watch (Fig. 5, 7th picture) UP9 = Rino touches the [+] widget 30 times (Fig. 5, 7th picture) UP10 = Rino touches the start widget (Fig. 5, 8th picture) UP11 = Rino sees the countdown on the screen (Fig. 5, 9th picture) UP12 = Rino turns his wrist to activate the watch screen to check his heart rate and performance (Fig. 5, 10th picture) UP13 = Rino sees that his heart rate is 169 BPM, distance 0,91 MI and pace 4'66"/MI (Fig. 5, 10th picture) UP14 = Rino feels the watch vibrate (Fig. 5, 11th picture) UP15 = Rino turns his wrist to activate the watch screen (Fig. 5, 11th picture) UP16 = Rino sees the message informing that he has reached half the way (Fig. 5, 11th picture) UP17 = Rino feels the watch vibrate (Fig. 5, 12th picture) UP18 = Rino turns his wrist to activate the watch screen (Fig. 5, 12th picture) UP19 = Rino sees the message informing that he has reached his goal (Fig. 5, 12th picture)
User Initiated Action	UP1, UP3, UP5, UP7, UP9, UP10, UP12, UP15, UP18
User Interpretation	UP2, UP4, UP6, UP8, UP11, UP13, UP14, UP16, UP17, UP19
Complex User Participation	UP9 (considering each touch to reach 30 minutes as an Atomic User Participation)
Unintentional User Participation	UP14, UP17
HC Interaction Sub-ontology	
For simplification reasons, in this sub-ontology we *instantiate* only *one* human-computer interaction. *The others involved in the Rino's case are similar to the one presented in the following*	
Interactive Computer System Participation (ICSP)	ICSP1 = The set of program executions and other events involving the Apple watch's computer system when interacting with Rino in the context of UP1 + UP2
Human-Computer Interaction	The interaction constituted of the Complex User Participation UP1 + UP2 and the Interactive Computer System Participation ICSP1
User Input Resulting State (UIRS)	UIRS1 = The situation achieved as a result of performing UP1 (i.e., the Apple watch crown pressed)
User Observable State (UOS)	UOS1 = The situation observable in the Apple watch screen in UP2 (i.e., the Siri message shown in the screen)
Program Copy Execution	The set of executions of program copies constituting the IOS, Siri and Activity that are loaded in the Rino's Apple watch, which led from UIRS1 to UOS1

After HCIO evaluation, we implemented HCIO's operational version using the open-source editor Protégé 5.5.0[4], which supports the construction of OWL[5] models, and its in-built reasoned HermiT 1.4.3. HCIO machine-readable version is available at `http://bit.ly/hcioOWL`. To provide a graphical visualization of the operational ontology, we used OWLGrEd[6] (UML style graphical editor for OWL) to create a web-based visualization, which is available at `http://bit.ly/hcioGrEd`. It is also available as RDF[7] at `https://bit.ly/hcioRDF` and Turtle[8] (textual syntax for RDF) at `http://bit.ly/hcioTurtle`.

[4]Available for download at `https://protege.stanford.edu/products.php#desktop-protege`
[5]W3C Web ontology Language, `https://www.w3.org/OWL/`
[6]Graphical Ontology Editor online visualization, `http://owlgred.lumii.lv/online_visualization`
[7]Resource Description Framework (RDF) - standard model for data interchange on the Web, `https://www.w3.org/RDF/`
[8]W3C Terse RDF Triple Language (Turtle) - a RDF graph in a compact textual form, `https://www.w3.org/TR/turtle/`

7. Discussion

In this section, we make some discussions about HCIO conceptualization.

Concerning different interaction types (see Section 2.1), HCIO is able to represent both *explicit* and *implicit* interaction (what we call respectively as intentional and unintentional). For example, in John's case, when he performs intentional actions driven by his goal of quoting flights price, we have explicit interaction. We have implicit interaction when Rino unintentionally inputs data from his pulse rate. HCIO also is able to represent *goal* and *data* or *event-driven behavior.* As previously discussed, in John and Rino cases, they interact with the computer system driven by the goals they want to achieve, thus, we have goal-driven behavior. However, when Rino perceives the watch vibrate, we have event-driven behavior.

To represent different types of user actions, we decided to use User Participation as a general term based on UFO. From this umbrella concept, there are subtypes of user participation that allow representing user intentional or unintentional actions (related to user input) and interpretation (related to system output). Regarding unintentional participation, it represents unintentional user actions mostly because it has no associated goal. It also represents unconscious and uncontrolled actions related to human vital activities such as physiological functions, that are indeed unintentional.

Since HCIO is devoted to human-computer interaction, it does not represent interaction between humans and non-computer systems (e.g., a typewriter or a shower). HCIO allows representing traditional interactive computers systems, computer household appliances (e.g., a coffee machine that has software and hardware able to run programs) and even more complex systems such as IoT (Internet of Things), among others. As for actions, all user actions considered in HCIO are actions the user performs to interact with the computer system. Thus, actions that do not involve interaction with a computer system (e.g., if the user drinks water when typing a URL in his/her computer) are not covered.

Although in this paper we have explored examples of HCI involving a single user, HCIO allows representing situations common in collaborative systems, where different users interact with the same interactive computer system at the same time. For these cases, the ontology represents the interaction from the point of view of each individual user. The common element in this collaborative interaction is the interactive computer system. For example, an interactive computer system allows a certain user to initiate an action (e.g., a user types a text in a shared document) and another user to interpret the system response to that action (e.g., another user sees the document and interprets the text added by the first user).

In the context of an interaction, HCIO allows representing situations in which there is a disruption to the communication process due to failure or error in the computer system. This may occur when a user input (User Input Resulting State) does not lead to the execution of programs or the execution of programs does not lead to new system output (User Observable State). For example, the user clicks a button, but nothing happens because there is no program associated with that button (an implementation error) or due to a failure in the corresponding program. This can be particularly useful to represent interactions with prototypes and support prototype evaluation. In HCIO, User Observable State is related to what Nielsen Nielsen (1993) calls user feedback. User feedback refers to a basic characteristic of usable interfaces in which the system should continuously inform the user about what it is doing Nielsen (1993). When the system fails (e.g., because it was not properly designed or due to hardware malfunction), it may not provide new feedback for the user. For example, if after the user provides an input to the system it behaves as if nothing had happened, the absence of change (e.g., because the program execution failed) is itself a User Observable State, which will be interpreted by the user, who will conclude that something went wrong. HCIO does not represent what went wrong (e.g., if the failure was caused by a design problem – the program was incorrect – or a hardware problem). It does not explicitly address the problems because we consider that this issue (hardware and software malfunction) is more related to Software Engineering and, thus, should not be treated as a core aspect of the HCI phenomenon.

Being a core reference ontology, HCIO aims to describe core aspects of the HCI phenomenon. Thus, particularities of specific types of interaction (e.g., gesture or haptic interaction) are not addressed. Therefore, forms of input control (e.g., by hand, by blinking eye, brain activity, by moving arms, whole-body), input (e.g., direct manipulation, pointing, mouse) and output (e.g., visual, graph, screen) modalities, types of user interface (e.g., haptic-tactile interface, brain-computer interface), types of interactive computer systems (e.g., intelligent house, physiological computing, adaptive systems) are outside of HCIO scope. Ontologies focusing on particular types of interactions and involved elements should be defined by reusing/specializing HCIO concepts. For example, if one wants to create a brain-computer interface ontology and needs to represent brain activities, he/she can extend the User concept (i.e., create new concepts related to it) to address user body and brain (and even brain areas), and can extend the User Initiated Participation or User Initiated Action (intentional brain activity) concepts to represent brainwave and relate it to brain areas in order to represent which area of the brain is responsible for inputs that generate different behaviors in the system.

As a core ontology, HCIO should allow representing situations under the perspective of the different interaction paradigms presented in Section 2.1. On one hand, HCIO is general enough to address the different paradigms at a higher abstraction level. On the other hand, by specializing HCIO concepts, it is possible to develop specific domain ontologies focusing on each paradigm. For example, by specializing HCIO, we have developed an ontology addressing the HCI phenomenon under the Cognitive Science perspective Costa et al. (2020b).

8. Using HCIO

In this section, we discuss some uses of HCIO. First, we describe how HCIO has been used as core ontology of an HCI Ontology Network (HCI-ON Costa et al. (2020)). Second, we discuss how we have explored HCIO together with other ontologies of HCI-ON to (i) support the evaluation of user experience in an immersive application and (ii) aid knowledge sharing in HCI design. Finally, we discuss some envisioned applications to HCIO.

By analyzing the HCI ontologies we found in the literature (see Section 3), we noticed a tendency to develop isolated ontologies with very specific purposes, to be used in specific applications and contexts. If there are many available ontologies, it may be hard to identify which ones are needed to solve a particular problem and integrate them to produce the necessary ontology. As a result, many times, one chooses to develop a new ontology from scratch, creating yet another ontology, even when there are others able to solve the problem of interest. Reusing existing ontologies contributes to improve the quality of the resulting ontology and increase the productivity of the ontology development process Poveda-Villalón et al. (2010).

In the last decades, two of the authors of this paper have faced similar problems in the Software Engineering domain. As a solution, they have worked on Software Engineering ontology networks and obtained several benefits from this strategy Borges Ruy et al. (2016). An ontology network (ON) is a collection of ontologies, included in such a network, related together, through a variety of relationships, such as alignment and dependency, sharing concepts and relations with other ontologies. ONs enable to establish a comprehensive conceptualization that provides a common understanding of the domain of interest and can be used as a reference to solve semantic interoperability and knowledge problems related to the conceptualization as a whole or 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.

Considering the advantages of using ONs, we have worked on an HCI ontology network called HCI-ON Costa et al. (2020). Figure 10 presents an overview of HCI-ON architecture. We adopted a three-layered architecture. At the top, we have the *foundational layer*, where we use the Unified Foundational Ontology (UFO) Guizzardi (2005) to provide the general ground knowledge for classifying concepts and relations in the ON. At the *core layer*, core ontologies are used to represent the general domain knowledge, being the basis for the domain networked ontologies. HCIO lies at this layer, providing the core conceptualization to all networked domain ontologies. At the bottom, in the *domain layer*, domain ontologies appear, describing more specific knowledge. This layer is divided into two sets of ontologies: *well-founded domain ontologies*, which represent important and more general aspects of the HCI domain and are developed based on the foundational and core ontologies; and *aligned ontologies*, which are existing ontologies (e.g., ontologies we found in the literature) plugged into the network, meaning that they are kept as they are, not affecting applications using those ontologies. In the figure, arrows indicate dependency relationships, meaning that an ontology reuses concepts from another. Dotted arrows indicate alignment relationships. For visualization reasons, the color used in the border of each aligned ontology is also used to identify its alignment relationships.

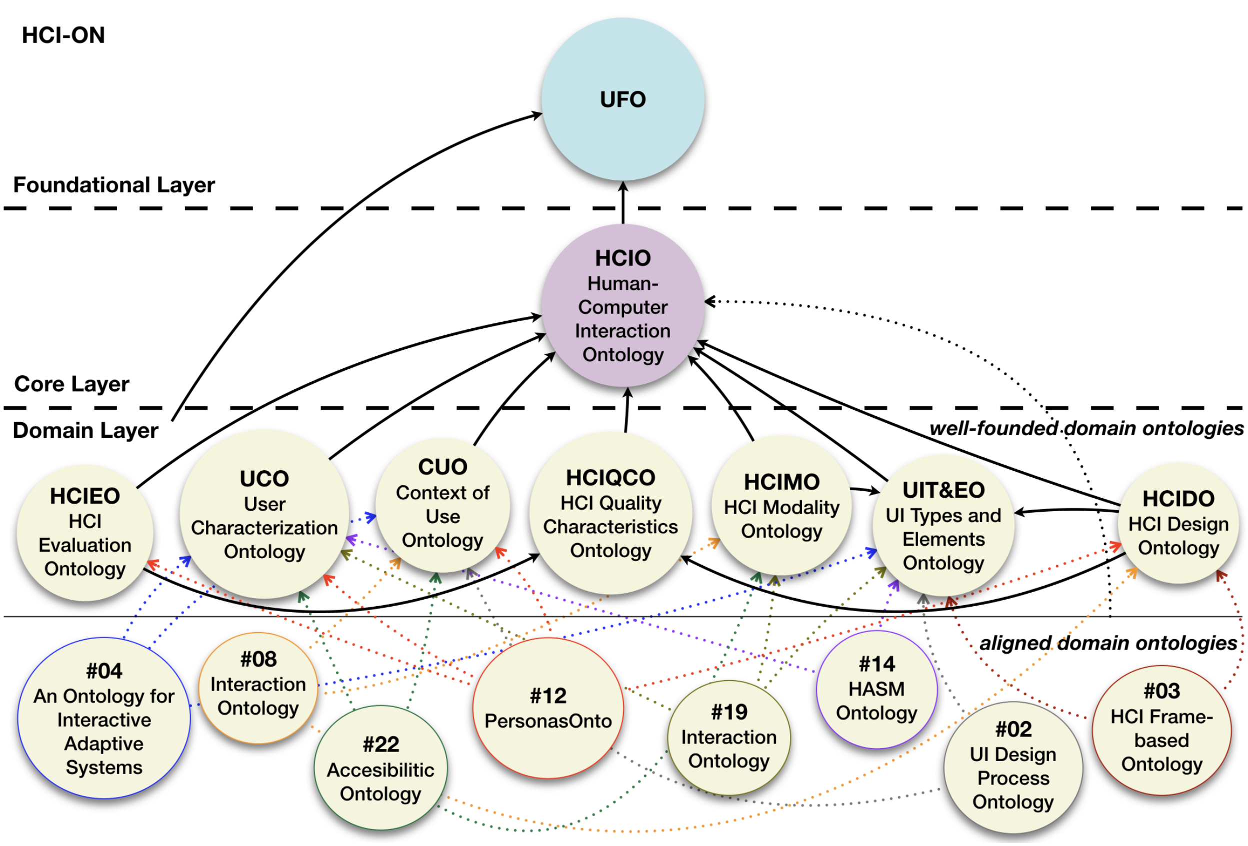

Figure 10: HCI ontologies as an Ontology Network

As it can be noticed in Figure 10, HCIO concepts (e.g., *User*, *Interactive Computer System* and *Human-Computer Interaction*) are reused by all well-founded domain ontologies. Moreover, HCIO concepts enable alignment between ontologies (represented in the figure by the single dotted line between the domain layer subdivision and HCIO). Hence, by providing the HCI core concepts, HCIO is essential to the ON growth. For example, in the well-founded domain ontologies layer, by extending the HCIO's *User Interface* concept, we are able to address different user interface types and elements as well as some aspects of interaction modality. Thus, we expanded the network by addressing these aspects in the UI Types and Elements Ontology (UIT&EO) and HCI Modality Ontology (HCIMO), respectively. With respect to alignment, HCIO conceptualization helps identify mappings between concepts of different ontologies and plug the ontologies into the network. As a result, the ON conceptualization gets more comprehensive by including the aligned ontology conceptualization, while the aligned ontology can benefit from the ON conceptualization as a whole. To exemplify how HCIO concepts have been reused to develop well-founded domain ontologies, Figure 11 shows a fragment of the HCI Design Ontology (HCIDO) Costa et al. (2020) where the HCIO concepts *User Interface Program*, *Interactive Software System* and *Interactive Computer System* are reused. Further information about the HCIDO can be found in Costa et al. (2020).

To illustrate the use of HCIO to align ontologies to the network, as well as to help integrate existing ontologies, Table 6 shows semantic mappings between concepts from the aligned ontologies shown in Figure 10 and HCIO.

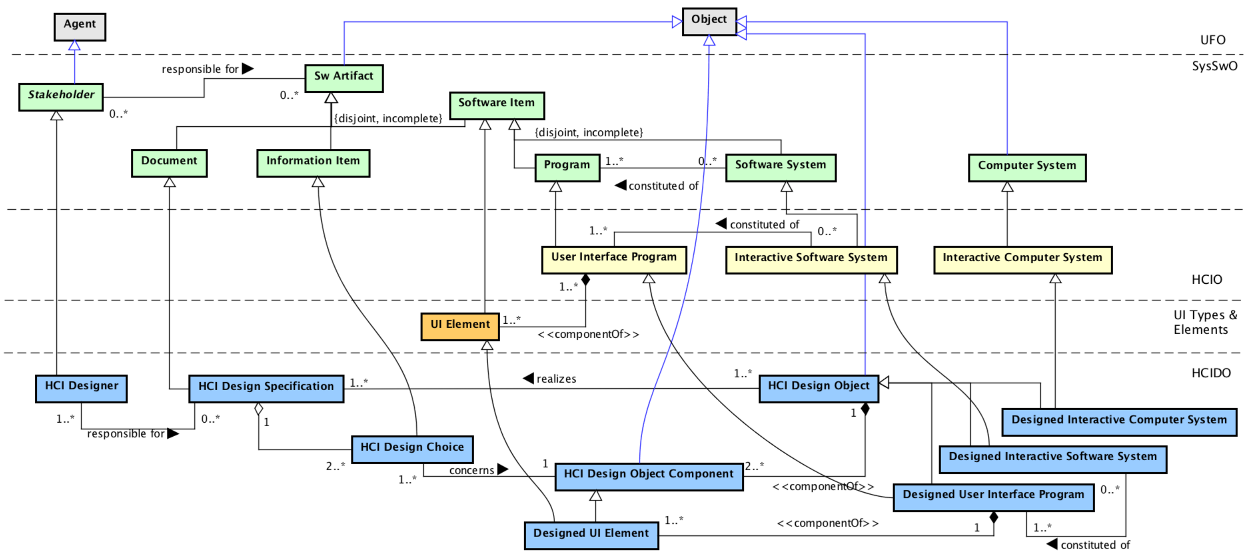

Figure 11: Fragment of HCI Design Ontology (HCIDO)Costa et al. (2020), showing HCIO concepts being reused

By organizing ontologies in an ON, when ontologies are needed in scenarios spanning different HCI subdomains, instead of spending effort to integrate several ontologies, one can just extract the ON portion to be used. As an example, we have used an HCI-ON extract composed of HCIO and other two ontologies to develop an ontology-based solution to evaluate user experience in an immersive mobile application.

Table 6: Aligning ontologies using HCIO

HCIO	**#02**	**#03**	**#04**	**#08**	**#12**	**#14**	**#19**	**#22**
User	Agent	Target User		User	Person	User	Craftswoman	User
Human-Computer Interaction	Interaction, Dialogue		Interaction					
Interactive Computer System	Interaction Object			Device	Product			
Interactive Software System				Application				
Input Equipment				Input Component			Input-medium	
Output Equipment				Output Component			Output-medium	
User Goal					Goal			
User Participation	Action							Activity Participation
User Initiated Participation							Input-Modality	

We have worked with the Usability and Software Engineering Research Group (USES Research Group[9]), which reported the need to evaluate the user behavior (particularly user experience) in his/her interaction with a mobile entertainment application for large events (Amazonas et al. (2020, 2019); Marques et al. (2020)). The application, called Compomus, is an immersive technology used by many people and its goal is to create a sense of immersion for the user by transforming the audience's role from a mere spectator to an active element of the show Amazonas et al. (2020, 2019); Marques et al. (2020). The user experience is measured by means of its engagement in the immersive interaction.

For evaluating the user behavior, it is necessary to collect data during the user interaction with the mobile application, use collected data to calculate user experience metrics and analyze them. Since interaction data regards many users and should be collected without interrupting the user experience, it is not feasible to collect and analyze data manually. Thus, an automated solution is needed. The solution consists of using ontologies to reason and as a basis of a system that collects and stores data, as well as analyses data and presents consolidated information about the user experience. The ontology is used both as conceptual model and operational ontology. The operational ontology, implemented in OWL[10], is used to capture interaction data recorded in logs. Data log is automatic read, loaded in OWL and stored in a triplestore built considering the ontology conceptual model. The operational ontology is then used to reason and provide information about user experience through a set of metrics such as interactivity and user interaction Marques et al.

[9] `http://uses.icomp.ufam.edu.br/index.php/en/`

[10] The current version of the HCI-ON extract used at UXON is available in RDF at `https://bit.ly/uxonRDF` and in Turtle at `https://bit.ly/UXONttl`

(2020). In this application, the use of HCIO enables to automatically measure and evaluate user experience.

The portion of the HCI domain involved in this scenario includes aspects related to HCI phenomenon (e.g., *User*, *Human-Computer Interaction*, *User Interface*, *User Action*, *User Participation*), user characterization (e.g., *User Profile*) and HCI evaluation (e.g., *HCI Evaluation*, *HCI Quality Characteristic*). Therefore, we extracted the ON fragment composed of HCI Ontology (HCIO), User Characterization Ontology (UCO), HCI Quality Characteristics Ontology (HCIQCO) and HCI Evaluation Ontology (HCIEO) fragments (which, as part of the ON, were already integrated) and used it in our ontology-based solution. To address the needed user experience metrics, we added some concepts to HCIQCO (e.g., we added interactivity and user interaction as specializations of *HCI Quality Characteristic*). Figure 12 illustrates the ontology-based solution in which HCIO is used to capture the user actions and support to evaluate user experience.

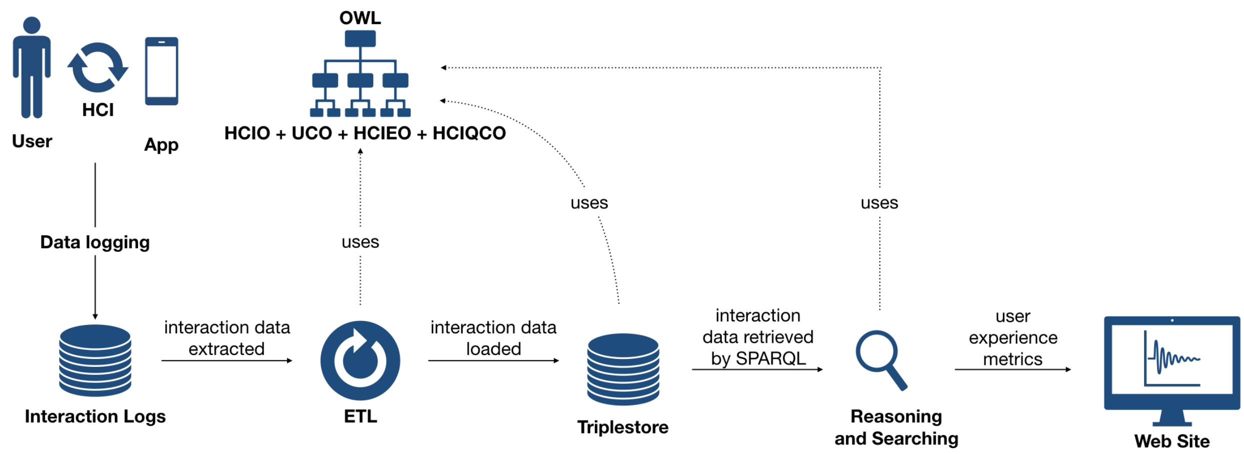

Figure 12: Ontology-based solution using HCIO

We have used another HCI-ON extract (composed of concepts from HCIO, HCIDO and UIT&EO) to support knowledge sharing in HCI design. The ontologies were used to build the conceptual model of a knowledge management tool called Knowledge Supporting Tool for Human-Computer Interaction Design (KTID) Castro et al. (2021). In KTID, knowledge related to HCI design decisions is recorded associated with HCIO concepts (e.g., *Interactive Software System*, *User Interface*) and can be retrieved by HCI designers, helping them to build suitable UI and keeping a record of the rationale behind the decisions made. As a result, designers can learn from previous experiences of other designers or even reuse solutions made available in the tool. Programmers, in turn, can access the designed UI and knowledge about the design choices to have a better understanding of what he/she will develop, reducing communication problems and anticipating the identification of potential problems. In this application, HCIO concepts are used to represent the interactive system and its components as well as user actions over the user interface, describing the action flows (interaction) designed to be followed by the user.

In addition to the use cases described above, next, we discuss other possible applications of HCIO. HCIO can be applied to harmonize standards. Different standards often adopt different terminologies and, sometimes, it is necessary to use different standards in a combined way. That means, it is necessary to harmonize different terminologies/conceptualizations so that different standards can work together and knowledge workers can better understand each other. Consider the three different definitions for "user interface" provided by ISO (2014, 2008, 2018, 2012) and presented in Section 1. As discussed, in those definitions it is not clear if interactive system, computer system and system have the same meaning. As a consequence, the definition of user interface is also not clear: does any system have user interface? By using HCIO as a reference conceptualization, it is possible to conclude that in those definitions interactive system, computer system and system have the same meaning, which is semantically equivalent to the Interactive Computer System concept in HCIO, because, according to HCIO, only interactive computer systems have user interface. Once the semantic conflict is solved, one can benefit from HCIO conceptualization by learning, for example, that in a human-computer interaction the user actions can be intentional or not.

When associated with the User Characterization Ontology and UI Types & Elements Ontology, HCIO can be used to support automatic adaptation of user interfaces. User actions (addressed in HCIO) can be captured and associated to user profiles (addressed in the User Characterization Ontology), helping identify how users with a certain profile behave when interacting with the system and, thus, adapting the user interface elements (addressed in the UI Types and Elements Ontology) according to the user actions.

9. Conclusions

This paper presented HCIO, a core reference ontology about the human-computer interaction phenomenon. HCIO scope was defined by means of competency questions as suggested by the adopted ontology engineering method. HCIO is modularized in three sub-ontologies explaining: (i) what an interactive computer system is, its components and its user interface; (ii) user actions taken in the course of an HCI and the user motivation to start an interaction; and (iii) how an HCI happens.

Considering that HCI is a complex domain, we argue that in order to provide a comprehensive and consistent conceptualization for representing HCI body of knowledge, ontologies covering different HCI aspects should be integrated. However, to produce a large monolithic HCI ontology is unfeasible. Thus, we advocate that ontologies should be organized in an ontology network Suárez-Figueroa et al. (2012). This way, knowledge is better structured and ontologies reuse concepts one from another, keeping consistency in shared concepts and decreasing overlap problems. In such an ontology network, HCI core concepts are particularly important, since they provide the basic notion of the HCI phenomenon that should be shared with ontologies describing specific HCI aspects.

In this sense, we have worked on the Human-Computer Interaction Ontology Network (HCI-ON) Costa et al. (2020). By describing core concepts of the HCI phenomenon, HCIO is at the heart of the HCI domain and thus it is the core ontology of HCI-ON. HCIO is essential to the ontology network growth (and, thus, knowledge growth), being the basis for adding new and existing ontologies to the network. We have proposed an initial version of HCI-ON in Costa et al. (2020).

Regarding the limitations of this work, we highlight HCIO evaluation. HCIO was evaluated mainly by the authors, which may result in biased results. To minimize this threat, we used real cases (the interactions described in the cases presented in Section 5 were executed in practice) to instantiate the ontology.

We have made HCIO available as a reference ontology (conceptual models, descriptions and axioms described in First Order Logic) and as an operational ontology (OWL and graphical visualization) aiming to provide a complete view of HCIO and allow users to use the view they are more interested in. In this paper, we highlighted the main cases where HCIO can be reused: as a basis to develop specific domain ontologies and to align or integrate existing ontologies. HCIO has been used (by us and also by other researchers) in the development of domain networked ontologies (e.g., HCI Design Ontology (HCIDO), HCI Evaluation ontology (HCIEO), User Characterization Ontology (UCO), HCI Quality Characteristics Ontology (HCIQCO), User Characterization Ontology (CUO) and UI Types & Elements Ontology (UIT&EO)) and has also served to align and integrate ontologies (as presented in Section 8).

HCIO is a central part of HCI-ON, but it is important to notice that it can also be reused by other ontologies and ontology networks through dependency or alignment relationships (e.g., ontologies addressing the software domain or domains that intersect with the HCI area). We have defined mechanisms (procedures) on how to reuse HCI-ON ontologies to develop new and align or integrate existing ontologies. These procedures can also be applied to reuse HCIO. The initial ideas of such procedures were discussed in Costa et al. (2020).

Concerning HCIO applications, we used HCIO together with other HCI-ON ontologies to develop two computational solutions that support HCI-related activities. By combining HCIO concepts with HCI evaluation concepts (from HCIEO), we have automatized the collection, storage and analysis of data about user experience, enabling to get conclusions about user experience when using Compomus (an immersive application) Amazonas et al. (2020, 2019); Marques et al. (2020). This work, in partnership with a group of usability specialists, is in the final stage. Some tests have already been done and the feedback we have gotten so far is encouraging: the task that UX/UI experts previously took days is now automated and presents information in a matter of seconds. The provided information has helped experts to identify possible improvements in the Compomus user interface to promote a higher quality UX for Compomus users.

By exploring HCIO with other HCI-ON ontologies, we have also supported knowledge sharing in HCI design. The developed tool has been perceived as particularly useful for less experienced HCI designers to learn from other designers' experience Castro et al. (2021).

We have experienced the benefits of reusing HCIO, both as a conceptual model and as an operational ontology. In the former, the ontology provides core knowledge to be reused to understand HCI central aspects (e.g., for communication and learning purposes), to support structuring knowledge basis and to develop (e.g., through specialization) more specific domain ontologies. In the latter, the ontology allowed automatizing HCI-related tasks, such as user experience measurement and evaluation.

Currently, we are exploring HCIO together with the User Characterization Ontology and the UI Types & Elements Ontology to develop an adaptive user interface system. In all the aforementioned initiatives, HCIO has been considered suitable and useful. In order to improve HCIO evaluation, we intend to apply HCIO in other cases and use it in practice to support HCI teaching and learning.

Acknowledgement

This research is supported by the Brazilian Research Funding Agency CNPq (grant 314174/2020-6).

References

J. M. Carroll, Human Computer Interaction (HCI), in: M. Soegaard, R. F. Dam (Eds.), The Encyclopedia of Human-Computer Interaction, 2nd ed., The Interaction Design Foundation, Aarhus, Denmark, 2014, pp. 21–61.

A. G. Sutcliffe, Requirements Engineering from an HCI Perspective, in: M. Soegaard, R. F. Dam (Eds.), The Encyclopedia of Human-Computer Interaction, 2 ed., The Interaction Design Foundation, Aarhus, Denmark, 2014, pp. 707–760.

ISO, International Standard - Systems and software engineering–Vocabulary, Standard ISO/IEC/IEEE 24765:2017, International Organization for Standardization, 2017. URL: https://pascal.computer.org/. doi:10.1109/IEEESTD.2017.8016712.

ISO, Systems and software engineering–Systems and software product Quality Requirements and Evaluation (SQuaRE) Common Industry Format (CIF) for usability: Context of use description, Standard ISO/IEC 25063:2014, International Organization for Standardization, 2014. URL: https://www.iso.org/standard/35789.html.

ISO, Systems and software engineering–requirements for designers and developers of user documentation, Standard ISO/IEC 26514:2008, International Organization for Standardization, 2008. URL: https://www.iso.org/standard/43073.html.

ISO, Systems and software engineering–Requirements for acquirers and suppliers of information for users, Standard ISO/IEC/IEEE 26512:2018, International Organization for Standardization, 2018. URL: https://www.iso.org/standard/72088.html.

ISO, Information technology – Object Management Group Architecture-Driven Modernization (ADM) – Knowledge Discovery Meta-Model (KDM), Standard ISO/IEC 19506:2012, International Organization for Standardization, 2012. URL: https://www.iso.org/standard/32625.html.

C. Pardo, F. J. Pino, F. Garcia, M. T. Baldassarre, M. Piattini, From chaos to the systematic harmonization of multiple reference models: A harmonization framework applied in two case studies, Journal of Systems and Software 86 (2013) 125–143. doi:10.1016/j.jss.2012.07.072.

H. Liyanage, P. Krause, S. De Lusignan, Using ontologies to improve semantic interoperability in health data, Journal of Innovation in Health Informatics 22 (2015) 309–315. doi:10.14236/jhi.v22i2.159.

A. Sene, B. Kamsu-Foguem, P. Rumeau, Data mining for decision support with uncertainty on the airplane, Data & Knowledge Engineering 117 (2018) 18–36. doi:10.1016/j.datak.2018.06.002.

H. Yago, J. Clemente, D. Rodriguez, P. Fernandez-de Cordoba, ON-SMMILE: Ontology Network-based Student Model for MultIple Learning Environments, Data & Knowledge Engineering 115 (2018) 48–67. doi:10.1016/j.datak.2018.02.002.

C. M. d. O. Rodrigues, C. Bezerra, F. Freitas, I. Oliveira, Handling Crimes of Omission by reconciling a criminal core ontology with UFO, Applied Ontology 15 (2020) 7–39. doi:10.3233/AO-200223.

B. Andersson, M. Bergholtz, A. Edirisuriya, T. Ilayperuma, P. Johannesson, J. Gordijn, B. Grégoire, M. Schmitt, E. Dubois, S. Abels, A. Hahn, B. Wangler, H. Weigand, Towards a Reference Ontology for Business Models, in: D. W. Embley, A. Olivé, S. Ram (Eds.), Proceedings of the 25th International Conference on Conceptual Modeling, Springer Berlin Heidelberg, Berlin, Heidelberg, 2006, pp. 482–496. doi:10.1007/11901181_36.

C. Rosse, J. L. Mejino, A reference ontology for biomedical informatics: the Foundational Model of Anatomy, Journal of Biomedical Informatics 36 (2003) 478–500. doi:10.1016/j.jbi.2003.11.007.

E. C. Bastos, M. P. Barcellos, R. d. A. Falbo, Using Semantic Documentation to Support Software Project Management, Journal on Data Semantics 7 (2018) 107–132. doi:10.1007/s13740-018-0089-z.

L. D. Renault, M. P. Barcellos, R. de Almeida Falbo, Using an Ontology-based Approach for Integrating Applications to support Software Processes, in: Proceedings of the 17th Brazilian Symposium on Software Quality, SBQS, ACM, New York, NY, USA, 2018, pp. 220–229. doi:10.1145/3275245.3275269.

J. C. Nardi, R. d. A. Falbo, J. P. A. Almeida, G. Guizzardi, L. F. Pires, M. J. van Sinderen, N. Guarino, C. M. Fonseca, A commitment-based reference ontology for services, Information Systems 54 (2015) 263–288. doi:10.1016/j.is.2015.01.012.

M. Barcellos, R. Falbo, V. G. V. Frauches, Towards a measurement ontology pattern language, in: Proceedings of the 1st Joint Workshop ONTO.COM / ODISE on Ontologies in Conceptual Modeling and Information Systems Engineering, volume 1301, 2014, p. 14.

D. Magro, A. Goy, A core reference ontology for the customer relationship domain, Applied Ontology 7 (2012) 1–48. doi:10.3233/AO-2012-0102.

L. F. Sikos, VidOnt: a core reference ontology for reasoning over video scenes, Journal of Information and Telecommunication 2 (2018) 192–204. doi:10.1080/24751839.2018.1437696.

G. Guizzardi, G. Wagner, J. P. A. Almeida, R. S. Guizzardi, Towards ontological foundations for conceptual modeling: The unified foundational ontology (UFO) story, Applied Ontology (2015). doi:10.3233/AO-150157.

A. C. d. O. Bringuente, R. d. A. Falbo, G. Guizzardi, Using a foundational ontology for reengineering a software process ontology, Journal of Information and Data Management 2 (2011) 511–526.

B. B. Duarte, A. L. de Castro Leal, R. de Almeida Falbo, G. Guizzardi, R. S. S. Guizzardi, V. E. S. Souza, Ontological foundations for software requirements with a focus on requirements at runtime, Applied Ontology 13 (2018) 73–105. doi:10.3233/AO-180197.

S. D. Costa, M. P. Barcellos, R. d. A. Falbo, M. V. H. B. Castro, Towards an Ontology Network on Human-Computer Interaction, in: Proceedings of the Conceptual Modeling, Springer International Publishing, Cham, 2020, pp. 331–341. doi:10.1007/978-3-030-62522-1_24.

J. Preece, H. Sharp, Y. Rogers, Interaction Design: Beyond Human-Computer Interaction, 4th ed., John Wiley & Sons, 2015.

A. Dix, J. E. Finlay, G. D. Abowd, R. Beale, Human-Computer Interaction, 3rd ed., Pearson Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 2004.

D. Benyon, Designing interactive systems A comprehensive guide to HCI and interaction design, 2nd ed., Pearson Education, 2010.

D. A. Norman, The Design of Everyday Things: Revised and Expanded Edition, 2nd ed., Basic Books, 2013.

A. Schmidt, Implicit human computer interaction through context, Personal Technologies 4 (2000) 191–199. doi:10.1007/BF01324126.

A. Schmidt, Interactive Context-Aware Systems Interacting with Ambient Intelligence, Ambient intelligence (2005) 159–178.

T. R. Gruber, Toward principles for the design of ontologies used for knowledge sharing?, International Journal of Human-Computer Studies 43 (1995) 907–928. doi:10.1006/ijhc.1995.1081.

A. Scherp, C. Saathoff, T. Franz, S. Staab, Designing core ontologies, Applied Ontology 6 (2011) 177–221. doi:10.3233/AO-2011-0096.

R. d. A. Falbo, M. P. Barcellos, J. C. Nardi, G. Guizzardi, Organizing Ontology Design Patterns as Ontology Pattern Languages, in: Proceedings of The Semantic Web: Semantics and Big Data, Springer Berlin Heidelberg, Berlin, Heidelberg, 2013, pp. 61–75. doi:10.1007/978-3-642-38288-8_5.

G. Guizzardi, On Ontology, Ontologies, Conceptualizations, Modeling Languages, and (Meta)Models, in: Proceedings of the Frontiers in Artificial Intelligence and Applications, Databases and Information Systems IV, IOS Press, Amsterdam, The Netherlands, The Netherlands, 2007, p. 18–39. doi:dl.acm.org/doi/10.5555/1565421.1565425.

R. Studer, V. Benjamins, D. Fensel, Knowledge engineering: Principles and methods, Data & Knowledge Engineering 25 (1998) 161–197. doi:10.1016/S0169-023X(97)00056-6.

N. Guarino, Formal Ontology in Information Systems: Proceedings of the 1st International Conference June 6-8, 1998, Trento, Italy, 1st ed., IOS Press, Amsterdam, The Netherlands, The Netherlands, 1998.

G. Guizzardi, Ontological foundations for structural conceptual models, Ph.D. thesis, University of Twente, 2005.

G. Guizzardi, C. M. Fonseca, A. B. Benevides, J. P. A. Almeida, D. Porello, T. P. Sales, Endurant Types in Ontology-Driven Conceptual Modeling: Towards OntoUML 2.0, in: Proceedings of the Conceptual Modeling, Springer, Cham, Xi'an, China, 2018, pp. 136–150. doi:10.1007/978-3-030-00847-5_12.

G. Guizzardi, R. Falbo, R. S. S. Guizzardi, 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), 2008, pp. 127–140.

G. Guizzardi, G. Wagner, R. d. A. Falbo, R. S. S. Guizzardi, J. P. A. Almeida, Towards Ontological Foundations for the Conceptual Modeling of Events, in: Proceedings of the Conceptual Modeling, Springer Berlin Heidelberg, Berlin, Heidelberg, 2013, pp. 327–341. doi:10.1007/978-3-642-41924-9_27.

S. Jalali, C. Wohlin, Systematic Literature Studies: Database Searches vs. Backward Snowballing, in: Proceedings of the ACM-IEEE International Symposium on Empirical Software Engineering and Measurement, ESEM '12, ACM, New York, NY, USA, 2012, pp. 29–38. doi:10.1145/2372251.2372257.

B. A. Kitchenham, S. Charters, Guidelines for performing Systematic Literature Reviews in Software Engineering, Technical Report, Software Engineering Group School of Computer Science and Mathematics Keele University and Department of Computer Science University of Durham, 2007.

S. D. Costa, M. P. Barcellos, R. d. A. Falbo, Ontologies in human–computer interaction: A systematic literature review, Applied Ontology 16 (2021) 421–452. doi:10.3233/AO-210255.

D. A. Norman, Cognitive engineering, in: User Centered System Design: New Perspectives on Human-Computer Interaction, 1 ed., L. Erlbaum Associates Inc., Hillsdale, NJ, USA, 1986, pp. 31–61.

G. Storrs, A conceptualisation of multiparty interaction, Interacting with Computers 6 (1994) 173–189. doi:10.1016/0953-5438(94)90023-X.

P. R. Suàrez, B. L. Jùnior, M. A. de Barros, Applying knowledge management in UI design process, in: Proceedings of the 3rd annual conference on Task models and diagrams - TAMODIA '04, ACM Press, NY, NY, USA, 2004, pp. 113–120. doi:10.1145/1045446.1045468.

M. Bakaev, T. Avdeenko, Ontology to Support Web Design Activities in E-Commerce Software Development Process, in: Proceedings of the IASTED International Conference on Automation, Control, and Information Technology - Information and Communication Technology, ACIT-ICT 2010, ACTA Press, Novosibirsk, Russia, 2010, pp. 241–248. doi:10.2316/p.2010.691-075.

M. Bakaev, T. Avdeenko, User Interface Design Guidelines Arrangement in a Recommender System with Frame Ontology, in: Proceedings of the Database Systems for Advanced Applications. DASFAA 2012. Lecture Notes in Computer Science, vol 7240, Springer Berlin Heidelberg, Berlin, Heidelberg, 2012, pp. 311–322. doi:10.1007/978-3-642-29023-7_31.

M. Bezold, W. Minker, A Framework for Adapting Interactive Systems to User Behavior, Journal of Ambient Intelligence and Smart Environments 2 (2010) 369–387. doi:10.3233/AIS-2010-0081.

I. Celino, F. Corcoglioniti, Towards the Formalization of Interaction Semantics, in: Proceedings of the 6th International Conference on Semantic Systems, I-SEMANTICS '10, ACM, Graz, Austria, 2010, pp. 10:1—-10:8. doi:10.1145/1839707.1839719.

A. Martín, G. Rossi, A. Cechich, S. Gordillo, Engineering Accessible Web Applications. An Aspect-Oriented Approach, World Wide Web 13 (2010) 419–440. doi:10.1007/s11280-010-0091-3.

S. K. Shahzad, M. Granitzer, D. Helic, Ontological model driven GUI development: User interface ontology approach, in: Proceedings of the 6th International Conference on Computer Sciences and Convergence Information Technology, ICCIT 2011, IEEE, Seogwipo, South Korea, 2011, pp. 214–218.

T. Tourwé, E. Tsipokova, N. González-Deleito, A. Hristoskova, Ontology-driven elicitation of multimodal user interface design recommendations, in: Proceedings of the BNAIC : Belgian/Netherlands Artificial Intelligence Conference, Ghent, Belgium, 2011, p. 8. doi:1854/LU-2012408.

C.-M. Chera, W.-T. Tsai, R.-D. Vatavu, Gesture ontology for informing Service-oriented Architecture, in: Proceedings of 2012 IEEE International Symposium on Intelligent Control, IEEE, Dubrovnik, Croatia, 2012, pp. 1184–1189. doi:10.1109/ISIC.2012.6398257.

D. Devaurs, A. Rath, S. Lindstaedt, Exploiting the user interaction context for automatic task detection, Applied Artificial Intelligence 26 (2012) 58–80. doi:10.1080/08839514.2012.629522.

A. Haller, T. Groza, F. Rosenberg, Interacting with Linked Data via Semantically Annotated Widgets, in: Proceedings of The Semantic Web, . JIST 2011. Lecture Notes in Computer Science, vol 7185, Springer Berlin Heidelberg, Berlin, Heidelberg, 2012, pp. 300–317. doi:10.1007/978-3-642-29923-0_20.

S. Negru, S. Buraga, A Knowledge-Based Approach to the User-Centered Design Process, in: Proceedings of the Knowledge Discovery, Knowledge Engineering and Knowledge Management. IC3K 2012. Communications in Computer and Information Science, vol 415., Springer Berlin Heidelberg, Berlin, Heidelberg, 2013, pp. 165–178. doi:10.1007/978-3-642-54105-6_11.

S. Negru, S. C. Buraga, Persona Modeling Process: From Microdata-based Templates to Specific Web Ontologies., in: Proceedings of the International Conference on Knowledge Engineering and Ontology Development - Volume 1: KEOD, (IC3K 2012), SciTePress, Barcelona, Spain, 2012, pp. 34–42. doi:10.5220/0004127000340042.

G. Salma, e. M. Marouane, Operating an application for modeling persona by using ontologies, Journal of Theoretical & Applied Information Technology 88 (2016) 57–64.

G. Salma, M. E. K. K. Eddine, M. C. B. Sabin, Use of ontologies in modeling persona, in: Proceedings of the 2012 IEEE International Conference on Complex Systems (ICCS), IEEE, Agadir, Morocco, 2012, pp. 1–9. doi:10.1109/ICoCS.2012.6458521.

E. Myrgioti, N. Bassiliades, A. Miliou, Bridging the HASM: An OWL ontology for modeling the information pathways in haptic interfaces software, Expert Systems with Applications 40 (2013) 1358–1371. doi:10.1016/j.eswa.2012.08.053.

E. M. Zamzami, E. K. Budiardjo, H. Suhartanto, Requirements recovery using ontology model for capturing end-to-end interaction of proven application software, International Journal of Software Engineering and its Applications 7 (2013) 425–434. doi:10.14257/ijseia.2013.7.6.36.

Y. Elyusufi, H. Seghiouer, M. A. Alimam, Building profiles based on ontology for recommendation custom interfaces, in: Proceedings of the 2014 International Conference on Multimedia Computing and Systems (ICMCS), IEEE, Marrakech, Morocco, 2014, pp. 558–562. doi:10.1109/ICMCS.2014.6911166.

N. Mezhoudi, J. Vanderdonckt, A user's feedback ontology for context-aware interaction, in: Proceedings of the 2015 2nd World Symposium on Web Applications and Networking (WSWAN), IEEE, Sousse, Tunisia, 2015, pp. 1–7. doi:10.1109/WSWAN.2015.7210331.

M. Bakaev, M. Gaedke, Application of evolutionary algorithms in interaction design: From requirements and ontology to optimized web interface, in: Proceedings of the 2016 IEEE NW Russia Young Researchers in Electrical and Electronic Engineering Conference (EIConRusNW), IEEE, St. Petersburg, Russia, 2016, pp. 129–134. doi:10.1109/EIConRusNW.2016.7448138.

F.-Z. Lebib, H. Mellah, L. Mohand-Oussaid, Ontological Interaction Modeling and Semantic Rule-based Reasoning for User Interface Adaptation, in: Proceedings of the 12th International Conference on Web Information Systems and Technologies - Volume 1: SRIS, (WEBIST 2016), INSTICC, SciTePress, Rome, Italy, 2016, pp. 347–354. doi:10.5220/0005854303470354.

M. Kultsova, A. Potseluico, I. Zhukova, A. Skorikov, R. Romanenko, A Two-Phase Method of User Interface Adaptation for People with Special Needs, in: Proceedings of the Conference on Creativity in Intelligent Technologies and Data Science. CIT&DS 2017. Communications in Computer and Information Science, vol 754., Springer, Cham, Volgograd; Russia, 2017, pp. 805–821. doi:10.1007/978-3-319-65551-2_58.

T. Robal, J. Marenkov, A. Kalja, Ontology Design for Automatic Evaluation of Web User Interface Usability, in: Proceedings of the 2017 Portland International Conference on Management of Engineering and Technology (PICMET), IEEE, Portland, OR, USA, 2017, pp. 1–8. doi:10.23919/PICMET.2017.8125425.

B. Romero Mariño, M. J. Rodríguez Fórtiz, M. Hurtado, H. M. Haddad, Accessibility and Activity-Centered Design for ICT Users: ACCESIBILITIC Ontology, IEEE Access 6 (2018) 60655–60665. doi:10.1109/ACCESS.2018.2875869.

M. D'Aquin, A. Gangemi, Is there beauty in ontologies?, Applied Ontology 6 (2011) 165–175. doi:10.3233/AO-2011-0093.

D. Vrandečić, Ontology Evaluation, Springer Berlin Heidelberg, Berlin, Heidelberg, 2009, pp. 293–313. doi:10.1007/978-3-540-92673-3_13.

J. Brank, M. Grobelnik, D. Mladenić, A survey of ontology evaluation techniques, in: Proceedings of the Conference on Data Mining and Data Warehouses (SiKDD 2005), Ljubljana, Slovenia, 2005, pp. 166–170.

R. d. A. Falbo, SABiO: Systematic approach for building ontologies, in: Proceedings of the CEUR Workshop, Rio de Janeiro, Brazil, 2014, p. 14.

A. R. Hevner, A Three Cycle View of Design Science Research, Scandinavian Journal of Information Systems 19 (2007) 87–92. URL: https://aisel.aisnet.org/sjis/vol19/iss2/4.

F. Borges Ruy, R. de Almeida Falbo, M. Perini Barcellos, S. Dornelas Costa, G. Guizzardi, SEON: A Software Engineering Ontology Network, in: Proceedings of the Knowledge Engineering and Knowledge Management, Springer International Publishing, Cham, 2016, pp. 527–542. doi:10.1007/978-3-319-49004-5_34.

C. Feilmayr, W. Wöß, An analysis of ontologies and their success factors for application to business, Data & Knowledge Engineering 101 (2016) 1–23. doi:10.1016/j.datak.2015.11.003.

ISO, Ergonomics of human-system interaction — Part 210: Human-centred design for interactive systems, Standard ISO 9241-210:2019, International Organization for Standardization, 2019. URL: https://www.iso.org/standard/77520.html.

C. S. De Souza, The Semiotic Engineering of Human-Computer (Acting with Technology), The MIT Press, Cambridge, Massachussetts, 2005. doi:10.1017/CBO9781107415324.004.

G. D. Abowd, R. Beale, Users, systems and interfaces: A unifying framework for interaction, in: Proceedings of the Sixth Conference of the British Computer Society Human Computer Interaction Specialist Groupe - People and Computers VI, Cambridge University Press, 1991, pp. 73–87.

T. Hewett, R. Baecker, S. Card, T. Carey, J. Gasen, M. Mantei, G. Perlman, G. Strong, W. Verplank, ACM SIGCHI Curricula for Human-Computer Interaction, Association for Computing Machinery, New York, NY, USA, 1992. doi:10.1145/2594128.

L. R. Krol, S. Freytag, M. Fleck, K. Gramann, T. O. Zander, A task-independent workload classifier for neuroadaptive technology: Preliminary data, in: Proceedings of the 2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC), 2016, pp. 3171–3174.

S. H. Fairclough, Fundamentals of Physiological Computing, Interact. Comput. 21 (2009) 133–145. doi:10.1016/j.intcom.2008.10.011.

D. A. Norman, The Design of Future Things, reprint ed., Basic Books, New York, NY, USA, 2009.

S. H. Fairclough, K. e. Gilleade, Advances in Physiological Computing, Human–Computer Interaction Series, 1 ed., Springer-Verlag London, 2014.

T. O. Zander, L. R. Krol, K. Gramann, Chapter 85 - Towards Neuroadaptive Technology: Implicitly Controlling a Cursor Though a Passive Brain–Computer Interface, in: H. Ayaz, F. Dehais (Eds.), Neuroergonomics, Academic Press, 2018, pp. 301–302. doi:10.1016/B978-0-12-811926-6.00085-3.

C. Belkhiria, V. Peysakhovich, Electro-Encephalography and Electro-Oculography in Aeronautics: A Review Over the Last Decade (2010–2020), Frontiers in Neuroergonomics 1 (2020) 3. doi:10.3389/fnrgo.2020.606719.

T. O. Zander, C. Kothe, S. Jatzev, M. Gaertner, Enhancing Human-Computer Interaction with Input from Active and Passive Brain-Computer Interfaces, Springer London, London, 2010, pp. 181–199. doi:10.1007/978-1-84996-272-8_11.

T. R. Clites, M. J. Carty, J. B. Ullauri, M. E. Carney, L. M. Mooney, J.-F. Duval, S. S. Srinivasan, H. M. Herr, Proprioception from a neurally controlled lower-extremity prosthesis, Science Translational Medicine 10 (2018) eaap8373. doi:10.1126/scitranslmed.aap8373.

T. O. Zander, C. Kothe, Towards passive brain–computer interfaces: applying brain–computer interface technology to human–machine systems in general, Journal of Neural Engineering 8 (2011) 025005. doi:10.1088/1741-2560/8/2/025005.

J. Nielsen, Usability Engineering, 1st ed., Morgan Kaufmann, San Francisco, USA, 1993.

R. Oliveira, P. Palanque, B. Weyers, J. Bowen, A. Dix, State of the Art on Formal Methods for Interactive Systems, in: The Handbook of Formal Methods in Human-Computer Interaction, Springer International Publishing, Cham, 2017, pp. 3–55. doi:10.1007/978-3-319-51838-1_1.

Y. Rogers, H. Sharp, J. Preece, Interaction Design: Beyond Human-Computer Interaction, Interaction Design: Beyond Human-computer Interaction, 3rd ed., John Wiley & Sons, Chichester, United Kingdom, 2011.

D. Saffer, Designing for Interaction: Creating Innovative Applications and Devices, 2nd ed., New Riders, 2010.

D. E. Callan, C. Terzibas, D. B. Cassel, M.-a. Sato, R. Parasuraman, The Brain Is Faster than the Hand in Split-Second Intentions to Respond to an Impending Hazard: A Simulation of Neuroadaptive Automation to Speed Recovery to Perturbation in Flight Attitude, Frontiers in Human Neuroscience 10 (2016) 187. doi:10.3389/fnhum.2016.00187.

S. D. Costa, M. P. Barcellos, R. d. A. Falbo, HCIO complete instantiation & dictionary of terms, Available at https://bit.ly/HCIO_instantiation, 2020a.

S. D. Costa, M. P. Barcellos, R. d. A. Falbo, Human-Computer Interaction Cognitive Engineering Ontology (HCICEO) - specification & instantiation & dictionary of terms, Available at http://bit.ly/_HCICEO (in Portuguese only), 2020b.

M. Poveda-Villalón, M. C. Suárez-Figueroa, A. Gómez-Pérez, Reusing Ontology Design Patterns in a Context Ontology Network, in: Proceedings of the 2nd International Conference on Ontology Patterns - Volume 671, WOP'10, CEUR-WS.org, Aachen, DEU, 2010, pp. 35–52.

M. Amazonas, T. Castro, J. G. Kienen, R. Freitas, B. Gadelha, Composing aleatoric music through interaction, Per Musi (2020) 1–35. doi:10.35699/2317-6377.2020.26077.

M. Amazonas, T. Castro, R. De Freitas, B. Gadelha, Composing through Interaction: a framework for collaborative music composition based on human interaction on public spaces, in: Proceedings of Brazilian Symposium on Musical Computing (SBCM 2019), Sociedade Brasileira de Computação - SBC, 2019, pp. 42–49. doi:10.5753/sbcm.2019.10421.

L. Marques, M. Amazonas, T. Castro, W. Assuncao, L. Zaina, B. Gadelha, T. Conte, UX-Trek: A post-interaction journey from immersive experience logs, in: Proceedings of the XIX Brazilian Symposium on Human Factors in Computing Systems (IHC'20), ACM, Diamantina, Brazil, 2020, p. 6. doi:https://doi.org/10.1145/3424953.3426547.

M. V. H. B. Castro, M. P. Barcellos, R. d. A. Falbo, S. D. Costa, Using Ontologies to aid Knowledge Sharing in HCI Design, in: Proceedings of the XX Brazilian Symposium on Human Factors in Computing Systems, ACM, New York, NY, USA, 2021, pp. 1–7. doi:10.1145/3472301.3484327.

M. C. Suárez-Figueroa, A. Gómez-Pérez, E. Motta, A. Gangemi, Ontology Engineering in a Networked World, Springer Berlin Heidelberg, Berlin, Heidelberg, 2012. doi:10.1007/978-3-642-24794-1.