Domain and Semantic Modeling in the Context of Interactive Systems Development: User and Device Cases

Saulo Silva(✉) and Orlando Belo

ALGORITMI R&D Centre/LASI, University of Minho, Braga, Portugal
saulo.silva@ifg.edu.br, obelo@di.uminho.pt

Abstract. The development of interactive systems requires tools and knowledge from a number of domains. The combination of Software Engineering with Human Factors aspects assists designers and engineers in designing error free, safe systems for humans' operators. This paper explores the use of multidimensional knowledge representation as means to narrow the gap between design and implementation of interactive systems. Two models are discussed for delivering heavyweight ontologies, by the means of Ontology–Driven Development. This might enable the development of software artifacts, with potential of improving system specification, information sharing and reusability. The theoretical approach is discussed; a literature revision in the topic is included, along with taxonomy enabling the description of user and device's characteristics. Finally, the ontological development is presented and as well as analyzed.

Keywords: Ontology-based Development · Requirements Analysis · User Interface · Interactive Systems

1 Introduction

When Information Systems (IS) are interactive, that is, operated by humans, they must include a fundamental component, namely the User Interface (UI), which is defined as the part of the system that humans come into contact physically, perceptually and conceptually [1]. Usually, user interfaces that provide graphical elements for controlling the system – for instance, buttons, screens, text boxes, and sliders, among others – are recognized as Graphical User Interfaces (GUI). User Interfaces seek providing system representation for human operators, exposing the control logic behind the system under control. Humans make use of the user interface for carrying out her or his goals with the computational system. Hence, problems in user interfaces have the potential of compromising such goal achievement. For computer systems with critical functions, such as medical devices, energy or aeronautical systems, referred to as interactive critical systems or High Assurance (HA) systems, problems in user interface design may cause losses of different nature, such as financial, environmental or social, which can be characterized as accidents [2].

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2023
M. Papadaki et al. (Eds.): EMCIS 2022, LNBIP 464, pp. 497–514, 2023.
https://doi.org/10.1007/978-3-031-30694-5_36

Developing efficient user interfaces for IS requires knowledge from different domains, such as Software Engineering (SE), Ergonomics, Psychology, or Human Factors, for delivering error-free, trustworthy and reliable interfaces. User interface experts use information from those domains for perceiving the most relevant aspects for an ideal system use, providing the best solutions in terms of computer controls suitable for human expectations, and consequently averting gaps that could induce to interaction errors. In this context, modeling can play an important role in assisting designers during user interface development. Model–Driven Development (MDD) is a software development paradigm that makes use of information present in models for reasoning about problems and solutions regarding aspects being implemented in software artifacts. Hence, user and device modeling, for instance, might provide important insights about the description of a potential system user and the computational system in use, respectively.

Another approach for information organization relies on knowledge representation or on ontologies, for representing explicit specifications of conceptualization [3]. For [4], ontologies might have different roles in Software Engineering field. They may be a conceptual basis for software components specification and development. Ontology–Driven Development (ODD) is a software engineering vision considering the need of semantic constraints for supporting development processes, "through which software becomes a direct projection of a semantic definition" [5]. Those constraints are valuable assets for requirement analysis activities, with the potential of reducing costs and assuring the ontological adequacy of the user interface of an information system.

Despite the specificities of ODD and MDD approaches, ontologies may be seen as models. ODD might be considered as a particular type of MDD and might assist in describing problem domain at development time, with potential of reducing the cost of conceptual analysis or assisting in the constraints/requirements phase by providing relations between knowledge elements extracted from application domains. By providing human factors knowledge in the context of user interface development processes, this paper presents a work that seeks narrowing the gap between design and implementation of user interfaces. Our strategy is discussing the necessity and usefulness of human and device knowledge in user interfaces development, describing a multidimensional knowledge representation model related with (abstracts) human and device models, and presenting a human device ontology based on multidimensional models. The remaining part of this paper is organized as follows: Sect. 2 presents the background related with ontologies and knowledge representation. Section 3, approach ontology development, reveals the human device ontology we developed, and discusses how Interactive Systems development can benefit from ontologies. Section 4 highlights results jointly with an ontological analysis; and finally, Sect. 5, presents some remarks and conclusions, and a few research lines for future work.

2 Related Works

According with [6], a (abstract) human model can be considered as a set of human parameters, represented by variables, which can be employed for describing users of a product. They differentiate a user model declaration from a user model instantiation, whereas the first relates with the establishment of user variables, and the second relates

with the creation of a user profile. The model is described using a machine and human-readable format. User models are relevant for a number of disciplines. They provide human factors information about users that can be considered during design project. Due to this, a number of standards have been created to provide user-modeling guidelines.

A different user-modeling paradigm is based upon context-aware information. In [7] we find an example of that approach. Authors propose a learning mechanism using historical context-aware information gathered from user for reflecting user models. The parameters considered in such models seek assisting systems in dealing with inter-user variance. According with [8], it refers to the variance of user parameters that must be recognized in system specification, which requires that the design cope with different types of users in the system's users. The intra-user variance refers to monitoring internal user knowledge that is necessary for performing the task. User knowledge is especially relevant in the design of interactive systems. It supports different types of analysis that have the potential of preventing unwanted user interface interactions.

Device models are also relevant in designing interactive systems, since it demands rigorous design of user interface structure and behavior for avoiding hazards. Device information can provide valuable understanding of user interface parameters, which can be considered in the analysis. This is especially relevant for interactive systems with critical functions, such as critical interactive systems, or high assurance systems.

Device modeling allows investigating user interface structural and behavioral parameters that are presented as actions that users are allowed to perform for achieving their goals and keeping performance. The interface structure is useful for reasoning about the user interface adequate working, in terms of structure's individual components (for instance, buttons, displays). It is motivated by the need of reasoning about the quality of this structure [9]. For instance, device based user interface individual components, such as buttons, when pressed, causes the activation or deactivation of internal system processes, and displays, which are employed for monitoring changes in internal variables of systems. On the other hand, user interface behavior is useful for reasoning about the adequate user interface working behavior, resulting in the individual components coordination of such systems, being motivated by the need for reasoning about the quality of this behavior – sets of user interface components, such as panels, which are a collection of components like displays and buttons that have different working processes intending the control of different output signals.

The relevance of user and system modeling in the development of interactive systems is undeniable. It has the potential of highlighting crucial and critical user and device aspects that might contribute for positive outcomes such as user satisfaction, improving efficiency and productivity during the interaction process, along with decreasing costs. For achieving that, a crucial phase in this process is the analysis of the models, i.e., providing assurance that it complies with the system specification. Semantic modeling and analysis can play an important role in this task, as models with semantic constraints have the potential in assisting designers and system engineers during the requirements phase of interaction system development. Knowledge Engineering is a discipline closely related with SE that involves methods for knowledge representation [10]. According with [11], agreeing on a single definition of Knowledge is difficult due to its level abstractness. While their work provides a range of different definitions, in the current work, we adopt

the definition given by [12], which consider knowledge as interrelations between entities and relationships, highlighting that "relationships between entities provide meaning, at the same time that entities derive their meaning from their relationships". Knowledge representation is then a research field that makes use of modeling activity for creating knowledge bases. In its turn, knowledge bases are useful for representing conceptualizations that are assumed existing in a certain domain of interest, along with the relations between them. Those conceptualizations and their respective relations represent commitments about that domain and might be used as a reusable knowledge repository. The level of commitments depends on the technology for representing knowledge. From the knowledge viewpoint, [13] defines ontology as the most useful organization of knowledge about a particular domain, evolving from the need of satisfying formal postulates of signification. This is achieved by the use of object-attribute-value (O-A-V) triplets, which in ontology theory represents facts about objects and their attributes [14].

Ontologies might be described according with the level of expressiveness and granularity, as graphically illustrated in Fig. 1 [15][16]. Top-level ontologies refer to general and high-level (abstracts) concepts like space, time, and object, among others and provide higher expressiveness, at the cost of lower granularity. Examples can be found in the works of [17] and [18]. Domain ontologies represent concepts that belong to a generic domain or part of the world, such as politics, biology or computer science [19]. Task ontologies describe knowledge about a task or activity, such as initiating a system procedure, ending a system mode selection, or engaging a specific operation mode. Example of task ontology can be found in the work of [20]. Application ontologies, which provide higher granularity, are given as specializations of both domain and task ontologies for a specific case application [21]. Apart from the potential of reusability at development time for reducing the cost of conceptual analysis, ontologies might also offer another advantage in the context of the software development life cycle, as observed in other types of formal specifications, namely allowing designers and engineers to devote time in conceptual modeling (planning) phase prior to development phase. This has the potential of preventing unnecessary overload on software development life cycle implementation and maintenance phases. For the reader interested in covering a wider list of ontologies applications in the context of the Software Engineering life cycle, the works of [10] and [22] provide a categorization of ontologies as well as enumerate other approaches for using ontologies in the context of SE.

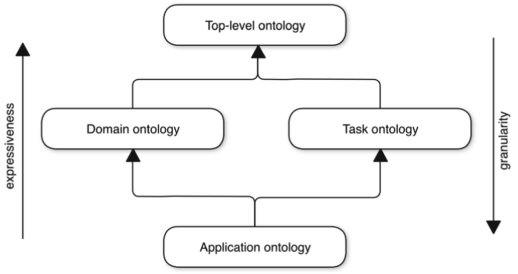

Fig. 1. Top-level, domain, task and application ontology in relation with granularity and expressiveness (Source: [15] and [16]).

The literature covers multiple strategies on Ontology development, according with [23] and [24]. It is important highlighting their main differences, as well agreeing on at least one. Ontology development approaches differs on their formalism level. Some approaches base their methodology on elaborating "tasks" or "principles" for ontology development (for instance, the methodology proposed by [25]). Other strategies focus on narrowing the distance from ontology development to other engineering disciplines, incorporating a defined life cycle for it (for instance, the work of [26] in establishing well-defined and iterative phases for ontology development). Following the same principle, the work of [27] comprises the creation of the *Methontology framework* intending standardizing the whole ontology development life cycle, transforming the process in engineering development.

[5] states that more importantly is agreeing on iterative and structured process, which expresses an ontology life cycle, with well-defined processes for each activity for building and maintaining the ontology, such as:

- Domain determination – the first important task in ontology development is considering the domain whose syntactic and semantic knowledge are modeled into a knowledge model.
- Terms enumeration – important expressions, which describe the domain or scope and can contain underlying concepts terms, properties related with those terms, and others, are enumerated [25].
- Definition of class and hierarchy – by examining the objects in previously defined concepts, intends developing the first idea of a class and its subclasses (i.e., a hierarchy). The approaches might be top-down (classes are first defined and then specialized in its subclasses), bottom-up (most relevant subclasses are enumerated and then grouped in different classes) or a combination of both with no loss for the result.
- Class properties definition – also referred to as slots, relates with definition of the internal structure of the class. All subsequent classes inherit the slots of their super classes.
- Domain and range of properties – define the requirements for the information slots hold, i.e., its allowed values (also referred to as facets definition). A facet describes values characteristics such as type, domain and cardinality and therefore has the potential of enforcing constraints for slots-values. For instance, slots-value *type* describes what types of values are allowed in the slot. Accepted types range from *String, Number, Boolean, Enumerated* or *Instance*. Slots-value *domain* refers to the classes describing the slot, which is hence referred to as the range of a slot. Slot-value *cardinality* is a property that refers to the quantity of values assigned to a slot. Some slot values require a single cardinality for correctly describing certain aspects of the reality (e.g., an animal has one body). Multiple-cardinality describes any number of values (e.g., the number of species of the animal kingdom).
- Properties instances – finally individual instance(s) are created by i) choosing a class, ii) creating individual instance of that class, and iii) filling all the defined slots of that class [25].
- Relations Definition – the main goal of identifying and characterizing relations (also referred to as *connections* or *associations*) is uncovering associations expressed between the characteristics and groups. [28] explains this identification as related

with uncovering associations between the characteristics and groups, while characterization follows the identifications and characterization of relations in a way that they could allow the relational knowledge being utilized in a near future for making useful inferences. For instance, the relation *is_part_of*, holds an association between two entities such as *software_button* and *user_interface*, in which it is built in.
- Data properties – its goal is twofold. They are used for connecting classes with specific data values, supporting the parameters that compose the models. Data properties are also used for restricting classes named *Defined Classes* under necessary and sufficient conditions.

We can find other examples of works in the literature related with ontologies in the context of SE life cycle, such as the work of [28] who proposed the IDEF5 ontology development method for promoting knowledge sharing and terminology standardization. [29] proposed the Definitions for User Experience Experimental Terms, a collection of terms interconnected into semantic network and followed by related axioms created in the context of Suggested Upper Merged Ontology (SUMO) ontology framework. [30] described GUMO, a user model ontology formalized in OWL language, capturing user important dimensions for use in user-adaptive systems, such as *Mental State, Personality* and *Contact Information.* Their ontology is revisited later, in the context of the semantic web ontology language [31]. [32], which described ontologies classifications and proposed taxonomy for assisting Software Engineering and Technology (SET) experts for understanding the linking between ontologies with some software development aspects. Other author, [33], presented a conceptualization for user interface development through ontological modeling, which provided descriptions of the user interface itself for a given architecture, for instance, in the description of components, relationships between components, among others. [34] presented a formal user interface representation describing both interaction and components aspects, while also discussed how the formal representation of user interface might benefit its development. In the context of Knowledge Management (KM), [35] presented OntobUMf, an ontology-based Framework for modeling user behavior. She proposed a model for user behavior and implemented a classifier for ranking users based on their knowledge level. [36] proposed a semantic framework in the context of the human system integration (HSI) research field. The main goal is extending the overall modeling capabilities in human-machine systems for improving human representation inside the overall system view. The work of [21] presented a particular implementation of user interface ontology tailored for the software development in avionics domain, where it might play a role as documentation resource. The approach made use of requirements ontology for supporting the creation of software test cases. Finally, [37] provided example of ontology creation for web user interfaces since the specification to the functional fragments.

3 The Design Approach

In this paper we propose cataloguing information on User and System domains for formalizing software models. This information includes the characteristics of the models, along with its descriptors (Table 1). The goal of the taxonomy is assisting in answering the following questions:

- What human and device characteristics are important considering for user interface development?
- What are the literature references that support these human factors and device characteristics?

For assisting this task, the taxonomy presents a set of ten (10) descriptors, such as name, description, unit of measurement, value space, taxonomy group, data type, how to measure/detect, reference/source, relations and comment (Table 1).

Table 1. Index of descriptors in the taxonomy.

Descriptor name	Descriptor summary
Name	A characteristic must present unique information that describes it
Description	Additional information for detailing the characteristic
Unit of measurement	The unit of measurement of the characteristic
Value space	The value space of the characteristic (*nominal, ordinal, interval, ratio, absolute*)
Taxonomy group	Which category the characteristic belongs to
Data type	Which kind of data the characteristic presents (*string, enumeration, list/vector, real, integer*)
How to measure/detect	How the characteristic information is measured or detected (type of measurement method)
Reference/source	Which are the references/standards that supports the characteristic
Relations	If the characteristics is related with any other
Comment	Further explanations/observations from the given characteristic

Table 2 presents a summary for user domain composed by a set of twenty-five (25) descriptors (for instance, related with physical body conditions), along with its seven (7) groups (encapsulating demographic, anthropometric, interaction related states, hearing, visual, motor and knowledge/experience data).

An excerpt (first level) composed by six (6) device descriptors, alongside its three (3) groups is presented (Table 3). The complete work extends up to five other levels providing classification for components, devices and systems.

The next tasks in ontology creation are supported by the taxonomy, whose main role is ensuring required information for the ontology development tasks.

- *Domain determination* – is performed by considering the need to know *Device* and *User* aspects. The literature of human-machine interaction provides a number of measurements for estimating the quality of that interaction. In this context, one of the most widely accepted measurements considered is provided by the Ergonomics of human-system *interaction's* chapter of the International Organization for Standardization

Table 2. User characteristics and groups.

Group name	Characteristic name
Demographic data	Age
	Gender
Anthropometric data	Weight
	Stature
Interaction related states	Perceived stress
	Stress and reaction time
	Perceived Fatigue
	Perceived Attitude towards computer
	Motivation
Hearing parameters	Hearing
	Hearing @ 500Hz
	Hearing @ 1kHz
	Hearing @ 2kHz
	Hearing @ 4kHz
	Background Noise
Visual parameters	Visual acuity (and sensitivity)
	Color perception
	Field of vision
Motor parameters	Contact grip
	Finger precision
	Hand precision
	Arm precision
	Pinch grip
	Clench grip
Knowledge and Experience	Semantic User's Knowledge and Experience

(ISO), which is standard ISO 9241–11. In this particular case the construct of *Usability* considers *User* and *Equipment* component, among others, as central for evaluating man-machine interaction aspects [38].

- *Terms enumeration* – important terms are enumerated for describing the domain or scope based on elements from the main taxonomy structure, i.e., taxonomy group names and taxonomy characteristics names.
- *Class and hierarchy definition* – a top-down approach might provide the first idea of a class and its subclasses (i.e., hierarchies), given by examining the objects in the previously defined concepts from domain or scope enumeration.

Table 3. Excerpt of Device characteristics and groups.

Group name	*Characteristic name*
Component Type	Input
	Output
Device Type	Generic Device
	Specialised Device
System Type	Application System
	Operational System

- *Class properties definition* – definition of class properties (also referred to as slots) is given by examining the taxonomy-defined hierarchy.
- *Domain and range of properties* – the descriptors *Unit of* measurement*, Value space* and *Data type* of the taxonomy assist in defining facets.
- *Properties instances – ultimately, individual instance(s) of class hierarchy and characteristics might be created.*
- *Definition of relations* –based on the analysis of user and device models previously presented, the authors propose an extension of the relations framework provided by [39]. A list of 33 tailored relations along with their inversion relation names, are presented (Table 4) in the context of the human and device ontology.

Table 4. Types and names of relations.

Relation type	*Relation name*	*Inverse relation name*
Compositional		
	Component of	Has component
	Has component	Component of
	Element of	Has element
	Has element	Element of
	Member of	Type
	Has member	Member of
	Part of	Has part
	Has part	Part of
	Gender of	Has gender
Spatial		
	Interacts with	Operator

(*continued*)

Table 4. (*continued*)

Relation type	*Relation name*	*Inverse relation name*
	Push	Is pushed
	Pull	Is pulled
	Press	Is Pressed
	Turn	Is Turned
Role		
	Instrument	Is resource
	Operator	Interact with
	Resource	Is instrument
	Input	Output
	Output	Input
General		
	Describes	Represents
	Represents	Describes
	Type	Member of
	Has attribute	Attribute of
	Has Gender	Gender of
Dependency		
	Depends on	Affected by
	Presumption for	Affected by
Influence		
	Influence on	Interact with
	Is opposing	Is supporting
	Is supporting	Is opposing
	Affect	Depends on
Rate		
	Age of	Has age
	Measure of	Has measure
	Rate of	Has rate

The set of relation types presented, albeit useful for reasoning about the main characteristics of the proposed ontology, when depleting its capability of capturing the totality of properties and relations from existing models, can be extended and specialized for supporting additional and more representative semantics. Ontologies are on-going work, i.e., they require evolution with updated entities, properties and relations, and so are these

relations classification. Additionally, we identified 132 mappings between relations and their source and destinations classes.

The goals of data properties are twofold. They are used for connecting classes to specific data values, supporting the parameters that compose the models, and also restrict classes named Defined Classes under necessary and sufficient conditions. We identified 25 data properties in the context of the current ontology (Table 5).

Table 5. Data property in the *User System Ontology*.

Data Properties
hasClenchGripValue
hasPinchGripValue
hasArmPrecisionValue
hasHandPrecisionValue
hasFingerPrecisionValue
hasContactGripValue
hasColourPerceptionValue
hasFieldOfVisionValue
hasVisualAcuityValue
hasHearingValue
hasBackgroundNoiseValue
hasHearing500HzValue
hasHearing4kHzValue
hasHearing2kHzValue
hasHearing1kHzValue
hasMotivationValue
hasPerceivedAttitudeTowardsComputerValue
hasPerceivedFatigueValue
hasStressAndReactionTimeValue
hasPerceivedStressValue
hasStatureInCM
hasGenderFM
hasWeightInKg
hasAgeInYears
hasSemanticKnowledgeAndExperienceValue

Next, a domain model is provided for representing the modeled objects (conceptual classes), as depicted in Fig. 2. Domain modeling organizes knowledge around the basic concepts that is investigated. The model is illustrated with a class diagram, which use classes and interfaces for capturing details about the entities that make up your system and the static relationships between them [40]. The reader will notice that the Device portion is condensed for space constraints.

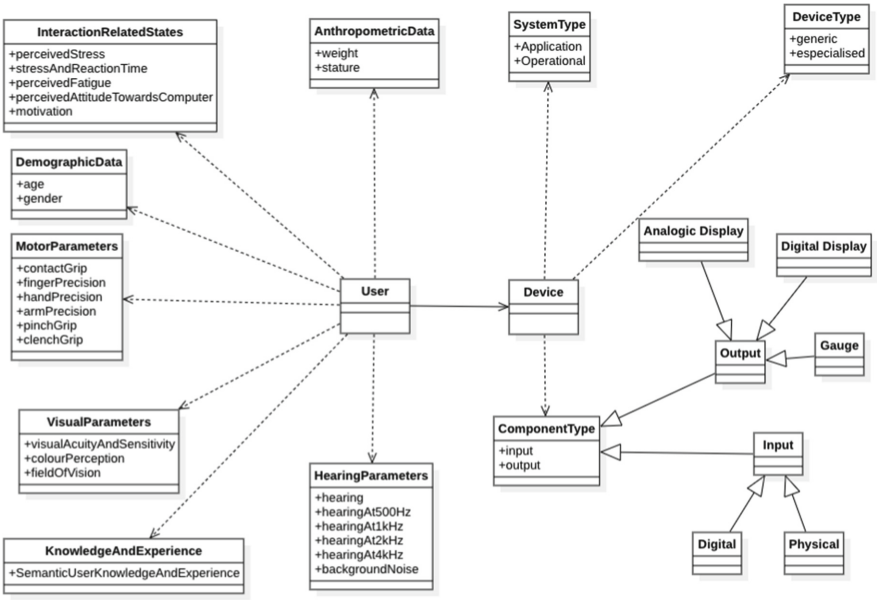

Fig. 2. Domain model of user and device information system.

The modeling is also supported by the conceptual model for the User portion of the Ontology, which is depicted in Fig. 3.

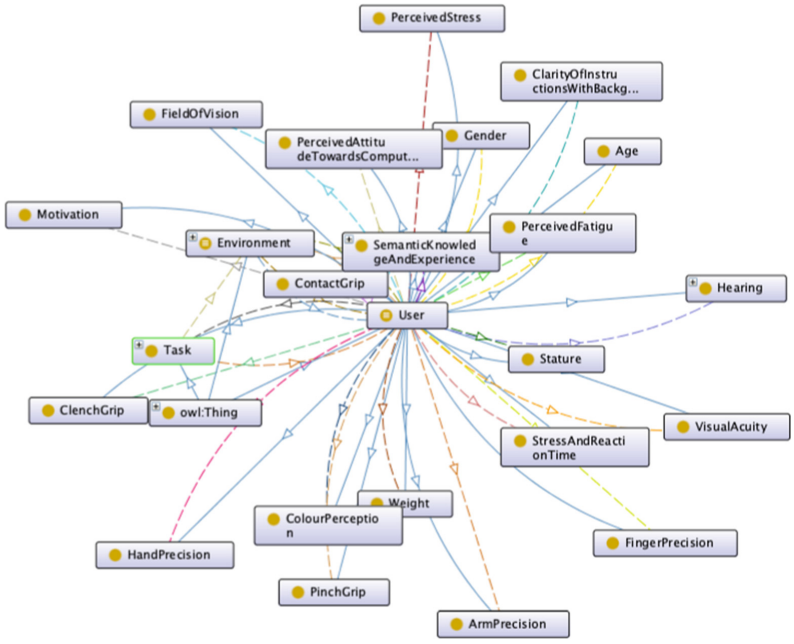

Fig. 3. Conceptual model from the User Ontology.

4 Results

The User Device Ontology is developed in OWL language, based on RDF and RDF Scheme, and implemented in Protégé Tool [25], presenting classes, properties and instances. The present ontology is heavy weighted (as opposite to those using controlled vocabularies), i.e., its internal structure represents semantic knowledge enriched with value and logical constraints. This particular type of ontology also has the potential of assisting designers in the requirements phase [32, 41]. Figure 4 depicts a fragment of the *User Device Ontology,* highlighting the restrictions that ties all the ontology parts together, i.e., the classes, sub-classes, class properties, range of properties, relations and data properties. This particular instance highlights the *User* requirement for operating *Devices* of the type *Specialized Device.* The restriction demonstrates the requirement of a user with *Semantic Knowledge and Experience* rated as value *3*, which according with [42] represents a domain expert user. The semantics of the restriction might also be expressed by the formal assertion in First Order Logic (FOL) notation:

$$\forall(SpecalisedDevice \in Devices)\exists User$$

$$\ni hasSemanticKnowledgeAndExperienceValue(User) \equiv 3$$

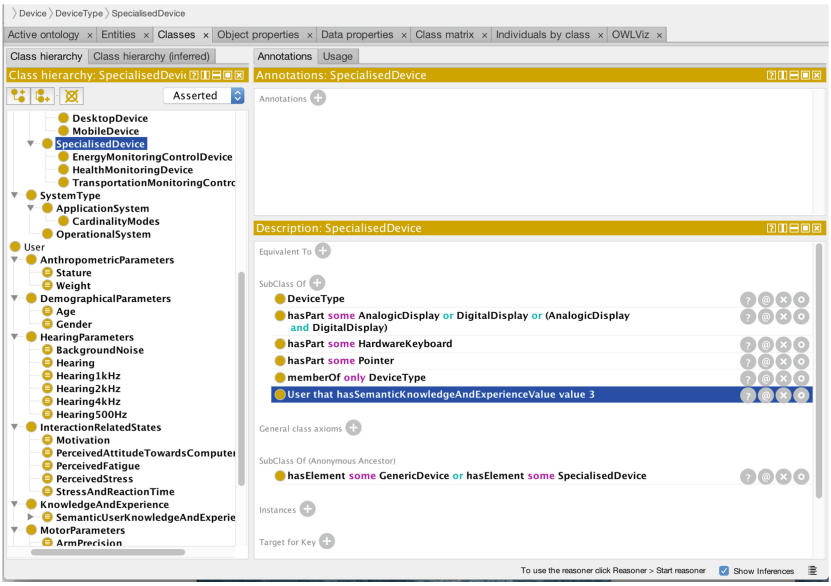

Fig. 4. Representation of the expert requirement for specialized devices in Protégé Tool [25]

A snippet of the OWL code for the User Device Ontology is provided in Fig. 5, contemplating the definition of classes and sub-classes.

```
<!-- http://www.semanticweb.org/saulo/ontologies/2020/9/User-Ontology/Age -->

<owl:Class rdf:about="http://www.semanticweb.org/saulo/ontologies/2020/9/User-Ontology/Age">
    <rdfs:subClassOf rdf:resource="http://www.semanticweb.org/saulo/ontologies/2020/9/User-Ontology#User"/>
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="http://www.semanticweb.org/saulo/ontologies/2020/9/User-Ontology/ageValue"/>
            <owl:someValuesFrom rdf:resource="http://www.w3.org/2001/XMLSchema#integer"/>
        </owl:Restriction>
    </rdfs:subClassOf>
</owl:Class>

```

Fig. 5. Fragment of OWL code, for the User Device Ontology.

Ontologies are evaluated according with different viewpoints (e.g., development quality, mathematical correctness, metrics). A *reasoner* might provide knowledge consistency checking, based on checking reflexive, transmission and redundancy properties of ontologies, providing contradictions implicit in the definitions, if any exist. In the case of the proposed ontology, logical consistency is evaluated against the HermiT OWL reasoner [43], and all detected classes are given as satisfiable. Additionally, descriptive metrics for the ontology's asserted classes are presented in Table 6.

Table 6. Descriptive metrics for the ontology's asserted class hierarchy

Name	Metric
Axiom	422
Logical axiom count	253
Declaration axioms count	168
Class count	93
Object property count	46
Data property count	25
Individual count	4
Annotation Property count	1
SubClassOf	126
EquivalentClasses	26

5 Conclusion and Future Work

This presented an on-going version of user device ontology tailored for assisting the design of interactive systems. The ontology aims contributing for the improvement of user interface design, considering semantic knowledge from two domains: human factors and devices. By providing a heavy weighted ontology involving knowledge from the explored domains, we intent assisting designers in the constraints/requirements phase of user interface development, with potential of narrowing the gap between user interface design and implementation. Therefore, we achieved the goals of the work, namely, we discussed the necessity and usefulness of human knowledge during the user interfaces development; we described a multidimensional knowledge representation model related with a (abstract) human model and device model and we presented an ontology based on the multidimensional models.

Future research includes extending the ontology with the complete four-domain models taxonomy, which are part of a research strand involving the design of an interactive systems evaluation platform, and submitting it for a journal. Also, we intent extending the validation method for the ontology, i.e., converting the protégé ontology for Alloy notation, which will allow testing instances and validate formal assertions about the represented knowledge. Counter-examples might be useful on the validation of mathematical/logical assertions. Additionally, moving to a realistic case study will allow the authors inform and validate the use of such ontologies in the development of user interfaces for interactive systems.

Acknowledgments. This work was supported by Conselho Nacional de Pesquisa (CNPq), COMPETE: POCI-01–0145-FEDER-007043, and FCT – Fundação para a Ciência e Tecnologia within within the R&D Units Project Scope: UIDB/00319/2020. We also acknowledge the support of Instituto Federal de Educação, Ciência e Tecnologia de Goiás (IFG).

References

1. Benyon, D., Designing Interactive Systems: A Comprehensive Guide to HCI, UX and Interaction Design (2014)
2. Leveson, N.: Engineering a Safer World: Systems Thinking Applied to Safety. MIT press, Cambridge (2011)
3. Gruber, T.: A translation approach to portable ontologies. Knowl. Acquisition **5**, 199–220 (1993)
4. Hesse, W.: Ontologies in the Software Engineering Process. In: EAI, pp. 3–16 (2005)
5. Tanasescu, V. (2004). An ontology–driven life–event portal. Master. Comput. Sci.
6. Kaklanis, N., et al.: Towards standardisation of user models for simulation and adaptation purposes. Univ. Access Inf. Soc. **15**(1), 21–48 (2014). https://doi.org/10.1007/s10209-014-0371-2
7. Moon, A., Choi, Y.I., Lee, B.S.: Context-aware user model for personalized services. In 2008 Third International Conference on Digital Information Management, pp. 858–863. IEEE (2008)
8. Eason, K.D.: Ergonomic perspectives on advances in human-computer interaction. Ergonomics **34**(6), 721–741 (1991)
9. Dix, A., Finlay, J., Abowd, G.D., Beale, R.: Human-Computer Interaction. Pearson Education, London (2003)
10. Happel, H.J., Seedorf, S.: Applications of ontologies in software engineering. In: Proceedings of Workshop on Sematic Web Enabled Software Engineering (SWESE) on the ISWC, pp. 5–9 (2006)
11. Jakus, G., Milutinovic, V., Omerovic, S., Tomazic, S.: Concepts, Ontologies, and Knowledge Representation. Springer, Cham (2013)
12. Milligan, C., Halladay, S.: The realities and facilities related to knowledge representation. In: IPSI Belgrade, Proceedings of the IPSI-2003 Montengro Conference, Sveti Stefan, Montenegro (2003)
13. Chan, C.W.: The knowledge modeling system and its application. In: Canadian Conference on Electrical and Computer Engineering 2004 (IEEE Cat. No. 04CH37513), vol. 3, pp. 1353–1356. IEEE (2004)
14. Gaševic, D., Djuric, D., Devedžic, V.: Model Driven Engineering and Ontology Development. Springer, Cham (2009)
15. Guarino, N.: Formal ontology in information systems. In: Proceedings of the first International Conference (FOIS'98), vol. 46. June 6–8, Trento, Italy. IOS press (1998)
16. Beißwanger, A.E. Developing Ontological Background Knowledge for Biomedicine (Doctoral dissertation) (2013)
17. Lenat, D.B.: CYC: a large-scale investment in knowledge infrastructure. Commun. ACM **38**(11), 33–38 (1995)
18. Miller, G.A., Beckwith, R., Fellbaum, C., Gross, D., Miller, K.J.: Introduction to WORDNET: an on-line lexical database. Int. J. Lexicogr. **3**(4), 235–244 (1990)
19. Clark, P.: Some Ongoing KBS/Ontology Projects and Groups (2000)
20. Musen, M.A., Gennari, J.H., Eriksson, H., Tu, S.W., Puerta, A.R.: PROTEGE-II: computer support for development of intelligent systems from libraries of components. Medinfo **8**(Pt 1), 766–770 (1995)
21. Tan, H., Adlemo, A., Tarasov, V., Johansson, M.E.: Evaluation of an application ontology. In: Proceedings of the Joint Ontology Workshops 2017 Episode 3: The Tyrolean Autumn of Ontology Bozen-Bolzano, Italy, September 21–23, 2017, vol. 2050. CEUR-WS (2017)

22. Alonso, J.B.: Ontology-based software engineering: engineering support for autonomous systems. Integrating Cognition+ Emotion+ Autonomy, 8–35 (2006)
23. Corcho, O., Fernández-López, M., Gómez-Pérez, A.: Methodologies, tools and languages for building ontologies: where is their meeting point? Data Knowl. Eng. **46**(1), 41–64 (2003)
24. Staab, S., Studer, R. (eds.): IHIS, Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-540-92673-3
25. Noy, N.F., McGuinness, D.L.: Ontology development 101: a guide to creating your first ontology. 2001 (2004). http://protege.stanford.edu/publications.
26. Falbo, A.: Integração de Conhecimento em um Ambiente de Engenharia de Software (Doctoral dissertation. Universidade Federal do Rio de Janeiro, Rio de Janeiro), Tese de Doutoramento (1998)
27. Fernández-López, M., Gómez-Pérez, A., Sierra, J.P., Sierra, A.P.: Building a chemical ontology using methontology and the ontology design environment. IEEE Intell. Syst. **14**(1), 37–46 (1999)
28. Benjamin, P.C., et al.: IDEF5 Method Report. Knowledge Based Systems, Inc (1994)
29. Niles, I., Pease, A.: Linking lixicons and ontologies: mapping wordnet to the suggested upper merged ontology. In: Ike, pp. 412–416 (2003)
30. Heckmann, D., Schwartz, T., Brandherm, B., Schmitz, M., von Wilamowitz-Moellendorff, M.: Gumo – the general user model ontology. In: Ardissono, L., Brna, P., Mitrovic, A. (eds.) UM 2005. LNCS (LNAI), vol. 3538, pp. 428–432. Springer, Heidelberg (2005). https://doi.org/10.1007/11527886_58
31. Heckmann, D., Schwarzkopf, E., Mori, J., Dengler, D., Kröner, A.: The user model and context ontology GUMO revisited for future web 2.0 extensions. Contexts Ontol. Representation Reasoning, 37–46 (2007)
32. Ruiz, F., Hilera, J.R.: Using ontologies in software engineering and technology. In: Calero, C., Ruiz, F., Piattini, M. (Eds.) Ontologies for software engineering and software technology, pp. 49–102. Springer, Heidelberg (2006). https://doi.org/10.1007/3-540-34518-3_2
33. Shahzad, S.K.: Ontology-based user interface development: user experience elements pattern. J. UCS **17**(7), 1078–1088 (2011)
34. Paulheim, H., Probst, F.: A formal ontology on user interfaces-yet another user interface description language? Position Paper. In: CEUR Workshop Proceedings, vol. 747, pp. Paper-9. RWTH (2011)
35. Razmerita, L.: An ontology-based framework for modeling user behavior—A case study in knowledge management. IEEE Trans. Syst. Man, Cybern.-Part A: Syst. Humans **41**(4), 772–783 (2011)
36. Orellana, D.W., Madni, A.M.: Human system integration ontology: enhancing model based systems engineering to evaluate human-system performance. Procedia Comput. Sci. **28**, 19–25 (2014)
37. Engelschall, R.S.: Hierarchical User Interface Component Architecture. Doctoral Thesis, Universit Augsburg. Germany (2018)
38. ISO/IEC (International Organization for Standardization) (1998). Standard 9241: Ergonomic Requirements for Office Work with Visual Display Terminals (VDT)s, Part 11. Guidance on Usability. https://www.iso.org/obp/ ui/#iso:std:iso:9241:-11:ed-1:v1:en. Accessed 20 Jan 2019
39. Štorga, M., Marjanović, D., Andreasen, M.M.: Relationships between the concepts in the design ontology. In: 16th International Conference on Engineering Design–ICED 07 (2007)
40. Pilone, D., Pitman, N.: UML 2.0 in a Nutshell. O'Reilly Media, Inc (2005)
41. Decker, B., Ras, E., Rech, J., Klein, B., Hoecht, C.: Self-organized reuse of software engineering knowledge supported by semantic wikis. In: Proceedings of the Workshop on Semantic Web Enabled Software Engineering (SWESE), p. 76. ISWC Galway, Ireland (2005)

42. Karwowski, W.: International Encyclopedia of Ergonomics and Human Factors, -3, vol. Set. CRC Press (2006)
43. Glimm, B., Horrocks, I., Motik, B., Stoilos, G., Wang, Z.: HermiT: an OWL 2 reasoner. J. Autom. Reason. **53**(3), 245–269 (2014)