Ontology-driven user interface development: Architecture and development proposal

João Pedro Sousa Nunes[1,2,*,†], Fernanda Farinelli[1,*,†] and Eduardo Ribeiro Felipe[3,*, †]

[1] *Faculty of Information Science, University of Brasilia, Campus Darcy Ribeiro - DF, Brasilia, Brazil*

[2] *Brazilian Institute of Science and Technology, SAUS Q 5, L 6, Bl H, Brasília, DF, Brazil*

[3] *University of Itajubá, Campus Itabira - MG, Brazil*

Abstract

This paper presents a development study proposing a system architecture for ontology-driven user interfaces. The system architecture integrates an OWL ontology, dynamically adapting the user interface based on semantic data extracted through RDFLib in Python. The proposed system facilitates the seamless interaction between ontology and user interface, allowing real-time data updates and instance management. Using Design Science Research, we developed an academic activities ontology and an ontology-driven user interface system that ensures data consistency and interoperability. The ODUI architecture is flexible and can be applied across various domains, providing an adaptable solution for domain-specific information systems. Future work includes integrating NoSQL databases to address the implementation of full CRUD functionality for managing instances and to enhance data storage and retrieval efficiency.

Keywords

Ontology-Driven User Interface, Ontology-driven GUI, Ontology-Driven Data Quality.

1. Introduction

Ensuring data quality in information systems is a critical challenge that impacts decision-making processes. A key factor contributing to poor data quality is the inadequacy of user interfaces, which often fail to guide users in accurately entering data. This issue is evident at the University of Brasília (UnB), where the registration of academic activities in the Integrated System for Academic Activity Management (SIGAA) is often inconsistent and incomplete. As a result, important academic events may not reach their target audience, and retrieving relevant data becomes inefficient.

The misalignment between user interfaces and the system's underlying knowledge domain contributes to validation errors, incorrect entries, and incomplete data. This highlights the need for user interfaces that dynamically adapt to the modeled knowledge domain to improve data quality. Such interfaces should facilitate accurate user interaction and be integrated with the domain knowledge they support.

In response to these challenges, this study poses the following research question: *How can we design user interfaces that dynamically adapt to the modeled knowledge domain to improve data quality in information systems?* The goal is to create interfaces that not only facilitate accurate user interaction but also integrate seamlessly with the domain knowledge they are built to support.

While the current study focuses on the registration of academic events, the proposed approach is flexible and can be applied to other domains. This paper presents a development study that proposes a system architecture for an Ontology-Driven User Interface. The architecture is designed to generate

dynamic, adaptable interfaces that reflect the structure of the knowledge domain, guiding users to input data accurately and comprehensively. By leveraging ontologies, which formally represent the domain's structure, the system ensures that user interfaces adjust to specific data requirements, thereby improving both usability and data quality.

2. Methodology

This research employs the Design Science Research (DSR) methodology, which is particularly well-suited for developing artifacts to solve problems. DSR involves iterative cycles of design, implementation, and evaluation, ensuring that the developed artifacts are both practical and theoretically sound [1]. In this study, two primary artifacts are proposed to develop: the ontology for academic activities at the UnB and a generic system architecture for an Ontology-driven User Interface (OUI). The ontology construction follows the proposal presented in the studies [2,3], which consists of five phases: Conceptual (define scope), Initiation (refining requirements), Design (conceptual modeling), Implementation (ontology formalization), and Deployment (ontology publish).

To understand the state of the art in adaptive and ontology-driven user interfaces, bibliographic research was conducted using databases like SciELO, Portal de Periódicos Capes, IEEE Xplore, and Web of Science. Searches included terms like "Adaptive User Interface," "Ontology-driven User Interface," and "Dynamic User Interface" in both English and Portuguese. This research provided insights into current architectures, technologies, and challenges, guiding the proposed system architecture.

3. Theoretical Background

Human-Computer Interface (HCI), is a multidisciplinary field dedicated to designing, evaluating, and implementing interactive computing systems for human use, aiming to improve user interaction by making systems more accessible, efficient, and enjoyable. HCI integrates knowledge from computer science, design, psychology, ergonomics, sociology, and engineering to develop intuitive user interfaces that meet user needs. Key components include usability, accessibility, and user experience, addressing technical, cognitive, and emotional aspects. Current HCI trends involve natural interfaces, augmented and virtual reality, adaptive interfaces, and AI-enhanced interactions [4,5,6].

Adaptive User Interface (AUI), a subfield of HCI, designs interfaces that automatically adjust to user needs, preferences, and abilities. AUI uses user modeling for profiles, dynamic real-time adaptation, and manual personalization. Examples include educational systems, virtual assistants, and context-aware mobile apps. Key challenges involve balancing usability with system complexity and ensuring privacy in user data collection [7,8].

Ontology-driven user interfaces (ODUIs) represent an approach to designing and developing user interfaces that dynamically adapt to the underlying knowledge domain represented by ontologies. By using ontologies as a core component, systems interpret user inputs, provide intelligent suggestions based on semantic relationships, and ensure accurate data entry. ODUIs promote interoperability by adhering to shared ontological standards, ensuring semantically consistent data that integrates seamlessly with other systems using the same ontology. This results in a flexible, evolving interface that offers optimized, context-aware user interactions. Studies [9-16] highlight the benefits of incorporating ontologies into interface design.

Recent research has advanced adaptive user interfaces, especially in component-based frameworks and ontology-driven systems. One study [17] introduced the CoBAUI framework, which simplifies UI development using reusable components, enabling dynamic context monitoring and adaptation through Angular-based structures, making it more flexible and industry-friendly. Another study [18] used ontological models to adapt mobile apps for users with cognitive impairments, generating personalized settings from health data to enhance usability. Additionally, research [19] integrated UI design patterns with ontology models to create adaptive systems that adjust to user needs and context, allowing dynamic changes based on user behavior or environment.

4. Ontology of Academic Events

The Ontology of Academic Events (ONTAE)[2] defines terms related to academic activities, such as *courses* and *academic events* (e.g. conferences, lectures), along with attributes like title, workload, and location. Built on the Basic Formal Ontology (BFO) version 2, ONTAE, a practice that promotes high-quality ontology development [20,21], ONTAE reuses established ontologies (e.g. Information Artifact Ontology and the Ontology for Biomedical Investigations), enhancing data interoperability and development efficiency [22]. ONTAE was formalized in OWL (Web Ontology Language) and developed using Protégé software.

Figure 1 provides an overview of ONTAE, showcasing the ontology's modeling of *academic activities* and *academic roles*, as well as the hierarchical relationships between these classes. The class *academic activity* is depicted as a subclass of *process* and *occurrent* from BFO, indicating that it represents events or activities occurring over time. Within this structure, the subclasses *course* and *academic events*. The class *academic role* was modeled as subclasses of the *role* class from BFO and represents roles held by individuals (*Person*) that are realized within the context of *academic activities*.

The *lecture* class is described as an *academic event* that involves participants with specific roles, occurs at a scheduled *date* and *time*, is located in some *academic space* (a subtype of the BFO class *site*), and has a *title* and *description*. A *lecture* realizes at least one *speaker* role (a person who delivers the lecture) and some *attendee* role (a set of persons who attend a lecture). In ONTAE, BFO *continuants* are used to represent entities that persist through time without being tied to a specific temporal event. These include both *material entities*, such as *Person* (from OBI) as a subtype of *object*, and *immaterial entities* like *roles* and *sites*. In addition, these *academic activities* happen on specific *dates* and *times* which are structured using BFO class *temporal region* (subtype of *occurent*), allowing for the precise representation of the intervals during which they take place.
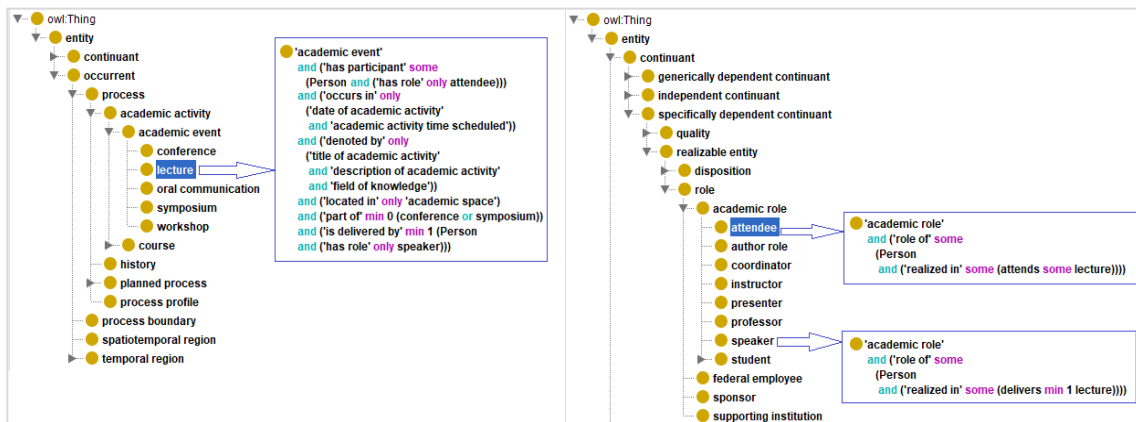

Figure 1: Detailed Hierarchy for *academic activity* and *academic role*. From authors, 2024.

5. Ontology-Driven System Architecture

Figure 2 shows the proposed system architecture for an ODUI (Ontology-Driven User Interface), showcasing the integration of components and technologies to dynamically adapt the user interface to the knowledge domain modeled by ontologies. First, an ontology engineer builds an *OWL ontology*, which serves as the core input for the system. In the *back end*, this ontology is parsed using the *RDFLib library* in Python, extracting semantic data for use by a *Flask web server*, which connects to the front end and handles the communication between the parsed ontology and the user interface.

---

[2] ONTAE OWL file available at https://purl.archive.org//ontologias/ONTAE/ONTAE.owl

The front end is built using *React*, along with *HTML5*, *CSS*, and *JavaScript*, to render a dynamic UI where the user interacts with options and data-driven by the ontology.
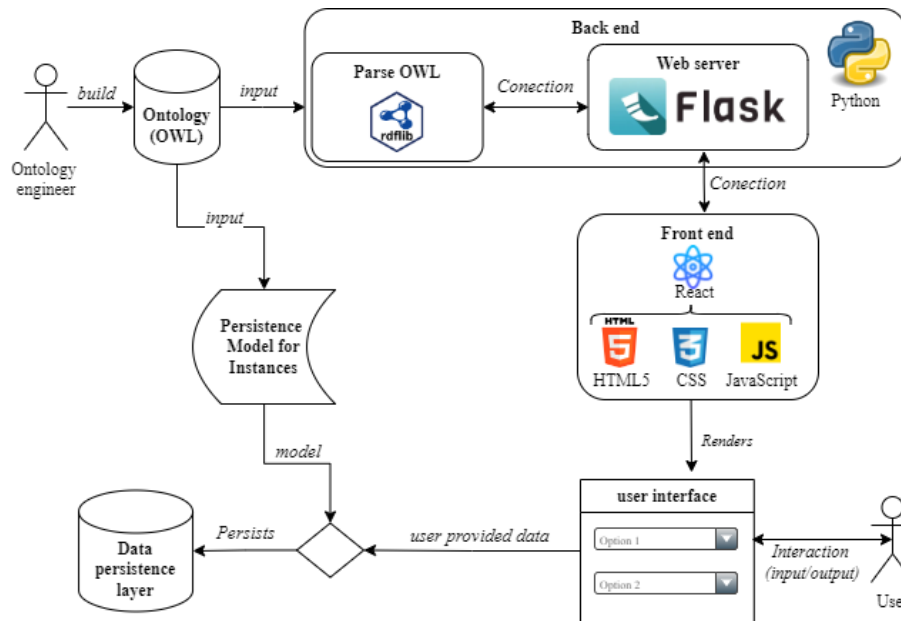

Figure 2: Proposed System Architecture for an ODUI. From authors, 2024.

The user's input is processed and combined into a *Persistence Model for Instances*, which maps the ontology's data for storage in the *Data Persistence Layer*, ensuring that the user-provided data is consistently stored, managed, and aligned with the ontology structure. Currently, the *Data Persistence Layer* is stored as a *JSON file*, allowing flexible storage and easy retrieval of data.

When the web server starts, the `/get_subclasses` route triggers parsing functions that utilize the *RDFLib library*, designed specifically for handling RDF data. These functions parse the *OWL file*, extracting all subclasses of the root class, which is defined as `ONTAE_00000000` (*academic activity*). Once the subclasses are retrieved, they are sent to the front end in *JSON format*, where they are displayed to the user, allowing selection of a subclass for further actions. If the selected subclass has additional subclasses, the `/get_subclasses` route is called again, enabling further user selections. If no more subclasses are available, the front end invokes the `/get_class_details` route, which retrieves all relevant properties, related classes, cardinalities, restrictions, and data types associated with the selected class, returning this information in a JSON format.

Figure 3 presents a snippet of this JSON output for the class *lecture*, illustrating how the system returns detailed information about class properties and relationships. The JSON output contains properties (predicates) that link the selected class (subject) to its related classes (objects). Each property is retrieved with its corresponding URI, label, and data type. Data types are derived from the data properties associated with the related class, as specified in the domain tag. Additionally, the output includes constraints or restrictions applied to these properties, such as `xsd:maxLength` for strings, or cardinality constraints like `only`, `exactly x`, `min x`, `max x`, and `some`. In some cases, if a related class has subclasses, the parsing method delves into the superclass to retrieve their URIs and labels, along with any inherited cardinality restrictions.

```
{
  "property": "http://purl.obolibrary.org/obo/RO_0001025",
  "label": "located in",
  "relatedClass": "academic space",
  "subclasses": [
    {
      "uri": "http://www.semanticweb.org/ontologias/ONTAE/ONTAE_00000012",
      "label": "laboratory"
    },
    {
      "uri": "http://www.semanticweb.org/ontologias/ONTAE/ONTAE_00000013",
      "label": "auditorium"
    },
    {
      "uri": "http://www.semanticweb.org/ontologias/ONTAE/ONTAE_00000014",
      "label": "classroom"
    }
  ],
  "cardinality": "only"
}
```

Figure 3: JSON output of properties and classes for the *lecture* class. From authors, 2024.

For example, in Figure 3, the object property *located in* links the input class to the academic space class, which includes subclasses such as *laboratory*, *auditorium*, and *classroom*. The cardinality `only` indicates that the event can occur in only one of these spaces, and the user interface reflects this by presenting a dropdown for the user to select one of the subclasses when registering an instance.

The user interface dynamically processes cardinalities for related classes. If a related class has a minimum cardinality of `0`, the field will be optional for the user. If the minimum cardinality is `1`, the field becomes mandatory. In cases where cardinalities include a maximum (e.g., `max x`), the interface may provide an `Add` button, allowing the user to add multiple fields until the maximum is reached. Cardinalities such as `some` provide a button to add multiple instances, while the `only` cardinality restricts the user to selecting just one option. For `exactly x`, the interface ensures that precisely x instances must be completed. These dynamic behaviors are demonstrated in Figure 2.

This approach, known as dynamic rendering, involves rendering the form based on the characteristics of each event, retrieving relationships and properties of the chosen class.

6. Discussion

This study demonstrates the potential and challenges of applying an ontology-driven architecture for adaptive user interfaces. The Ontology of Academic Events was written in OWL and provides a flexible framework for modeling domains. OWL is essential for this architecture, as RDFLib, the library used for parsing, relies on ontologies being in OWL or RDF format. Tools like Protégé facilitate this process by allowing ontologies to be saved in formats compatible with the system.

A key challenge is dynamic UI rendering, where the interface must adapt to changing ontology properties, relationships, and cardinalities. While the system handles this functionality, further refinement is required to improve the user experience, especially when managing previously saved data. At this stage, the architecture only addresses the CREATE functionality of the CRUD (Create, Read, Update, and Delete) operations for ontology instances, with future work planned to incorporate the remaining CRUD functionalities. This architecture supports seamless interaction between the ontology and users, enabling real-time updates and data storage while maintaining a clear separation of concerns between the ontology, backend logic, and the frontend interface.

7. Final remarks

This ongoing study demonstrates the potential of an ontology-driven user interface (ODUI) for registering academic events, with a flexible architecture that can easily adapted to other domains, such as healthcare or any field requiring complex data and relationship management.

Future developments will include migrating from JSON-based instance storage to a more scalable solution, such as a NoSQL database (e.g., MongoDB or Neo4J), along with the integration of full CRUD functionality for managing instances and enhancing data storage and retrieval efficiency. The dynamic user interface will also be refined to handle complex class structures more efficiently, enhancing data input and editing capabilities. Further exploration of additional tools, such as DeepOnto and Owlready2, will be pursued to optimize data retrieval and manipulation. Finally, improvements in UX/UI design and accessibility will be prioritized to ensure the system is intuitive, user-friendly, and accessible to a broader audience, enhancing the overall user experience.

References

[1] Bax, M. P. (2013). Design science: filosofia da pesquisa em ciência da informação e tecnologia. Ciência Da Informação, 42(2), 521–533. https://revista.ibict.br/ciinf/article/view/1388

[2] Farinelli, F. (2020). Um diálogo entre o realismo ontológico e a engenharia de ontologias na construção de artefatos de representação. In REPRESENTAÇÃO DO CONHECIMENTO, ONTOLOGIAS E LINGUAGEM: pesquisa aplicada em Ciência da Informação (1a Ed., pp. 277–294). Editora CRV. https://scholar.google.com.br/citations?view_op=view_citation&hl=pt-BR&user=b2x6tLwAAAAJ&pagesize=80&sortby=pubdate&citation_for_view=b2x6tLwAAAAJ:O3NaXMp0MMsC

[3] Farinelli, F., & Elkin, P. L. (2017). Construção de ontologia na prática: Um estudo de caso aplicado ao domínio obstétrico. Ciência Da Informação, 46(1). http://revista.ibict.br/ciinf/article/view/4018

[4] Carvalho, J. O. F. de. (2003). O papel da interação humano-computador na inclusão digital. Transinformação, 15, 75–89.

[5] Hartson, H. R., & Hix, D. (1989). Human-computer interface development: Concepts and systems for its management. ACM Computing Surveys, 21(1), 5–92. https://doi.org/10.1145/62029.62031

[6] Hefley, W. E., & Murray, D. (1993). Intelligent user interfaces. Proceedings of the 1st International Conference on Intelligent User Interfaces, 3–10. https://doi.org/10.1145/169891.169892

[7] Ahmad, S., Rahman, M., Khan, M. H., & Umar, M. S. (2015). A novel framework for adaptive user interface. 2015 Communication, Control and Intelligent Systems (CCIS), 427–432. https://doi.org/10.1109/CCIntelS.2015.7437954

[8] Balint, L. (1995). Adaptive interfaces for human-computer interaction: A colorful spectrum of present and future options. 1995 IEEE International Conference on Systems, Man and Cybernetics. Intelligent Systems for the 21st Century, 1, 292–297. https://doi.org/10.1109/ICSMC.1995.537774

[9] Shahzad, S. K., Granitzer, M., & Tochterman, K. (2009). Designing User Interfaces through Ontological User Model: Functional Programming Approach. 2009 Fourth International Conference on Computer Sciences and Convergence Information Technology, 99–104. https://doi.org/10.1109/ICCIT.2009.330

[10] Shengyang Luo, Yinglin Wang, & Jianmei Guo. (2009). Research on ontology-based usable user interface layout approach. 2009 IEEE International Conference on Intelligent Computing and Intelligent Systems, 234–238. https://doi.org/10.1109/ICICISYS.2009.5357748

[11] Paulheim, H., & Probst, F. (2010). Ontology-Enhanced User Interfaces: A Survey. International Journal on Semantic Web and Information Systems, 6(2), 36–59. doi: 10.4018/jswis.2010040103

[12] Aragones, A., Bruno, J., Crapo, A., & Garbiras, M. (2006). An ontology-based architecture for adaptive work-centered user interface technology. IP.

[13] Luo, S., Wang, Y., & Guo, J. (2009). Research on ontology-based usable user interface layout approach. 2009 IEEE International Conference on Intelligent Computing and Intelligent Systems, 234–238. doi: 10.1109/ICICISYS.2009.5357748

[14] Smirek, L., Zimmermann, G., & Beigl, M. (2016). Adaptive User Interfaces as an Approach for an Accessible Web of Things. Proceedings of the Seventh International Workshop on the Web of Things, 22–24. doi: 10.1145/3017995.3018000

[15] Freitas, A. A. C. de, Scalser, M. B., Costa, S. D., & Barcellos, M. P. (2022). Towards an ontology-based approach to develop software systems with adaptive user interface. Proceedings of the 21st Brazilian Symposium on Human Factors in Computing Systems, 1–7. doi: 10.1145/3554364.3559139

[16] Freitas, A. A. C. D., Costa, S. D., Scalser, M. B., & Barcellos, M. P. (2023). Using Networked Ontologies to Support the Development of Software Systems with Adaptive User Interface. Journal on Interactive Systems, 14(1), 257–273. doi: 10.5753/jis.2023.3256

[17] E. Yigitbas, K. Josifovska, I. Jovanovikj, F. Kalinci, A. Anjorin, and G. Engels, "Component-based development of adaptive user interfaces," in Proceedings of the ACM SIGCHI Symposium on Engineering Interactive Computing Systems, Valencia Spain: ACM, Jun. 2019, pp. 1–7. doi: 10.1145/3319499.3328229.

[18] D. Fedasyuk and I. Lutsyk, "Tools for adaptation of a mobile application to the needs of users with cognitive impairments," in 2021 IEEE 16th International Conference on Computer Sciences and Information Technologies (CSIT), LVIV, Ukraine: IEEE, Sep. 2021, pp. 321–324. doi: 10.1109/CSIT52700.2021.9648702.

[19] A. Braham, F. Buendía, M. Khemaja, and F. Gargouri, "User interface design patterns and ontology models for adaptive mobile applications," Pers Ubiquit Comput, vol. 26, no. 6, pp. 1395–1411, Dec. 2022, doi: 10.1007/s00779-020-01481-5.

[20] S. Schulz et al., "Guideline on developing good ontologies in the biomedical domain with description logics," Institut für Philosophie, Universit"at Rostock, Technocal report Version 1.0, 2012. Accessed: Sep. 06, 2024. [Online]. Available: https://www.iph.uni-rostock.de/forschung/homepage-goodod/guideline/

[21] S. Schulz, "The Role of Foundational Ontologies for Preventing Bad Ontology Design.," in Proceedings of the Joint Ontology Workshops 2018, Cape Town, South Africa: CEUR Workshop Proceedings, 2018. Accessed: Sep. 06, 2024. [Online]. Available: https://ceur-ws.org/Vol-2205/paper22_bog1.pdf

[22] M. Katsumi and M. Grüninger, "What is ontology reuse?," in Proceedings of the 9th International Conference (FOIS 2016), in Frontiers in Artificial Intelligence and Applications, vol. 283. Annecy, France: IOS Press, 2016, pp. 9–22. doi: 10.3233/978-1-61499-660-6-9.