Teaching knowledge graphs: A journey from logic to Web development and semantics-driven engineering

Robert Andrei Buchmann[1] and Ana-Maria Ghiran[1]

[1] *Babeş-Bolyai University, Faculty of Economics and Business Administration, 58-60 Teodor Mihali Street, Cluj-Napoca, 400591, Romania*

Abstract

This experience paper reports on the almost 15 years long journey of teaching knowledge representation and engineering topics in the Business Informatics study programs of the host university where the authors have been active during this time. The journey has been subjected to several factors - some local, some global - having various degrees of influence on the design rationale and deployment of teaching what is nowadays branded by the "Knowledge Graphs" buzzword, as well as its underlying Conceptual Modeling paradigm. Local factors include the local IT labor market dominated by an outsourcing culture that pressure curricular contents to align to immediate needs of influential IT service providers - typically working on maintaining/patching legacy systems, providing quality assurance for products developed elsewhere or developing low-innovation products. Global factors refer to the slow uptake of the Semantic Web paradigm and its gradual pragmatic re-branding and focus shifts - from ontology engineering to Linked Open Data, to semantic graph databases, to the Schema.org markup incentive and so on. For some years this has gone hand in hand with limited availability of educational tooling, proofs of concept, proofs of commercial value or means of producing educational content. The paper reflects on lessons learned and outcomes of the (constructivist) strategies employed to maintain and consolidate a knowledge representation curricular offer.

Keywords

Conceptual Modeling, Constructivist Education, Knowledge Graphs, Metamodeling, Agile Modelling Method Engineering, Semantics-driven Engineering

1. Introduction

This paper reports on a longitudinal experience in the authors' host university, in Business Informatics study programs, with teaching various forms of knowledge representation and engineering for almost 15 years, through several stages of curricular and content re-designs motivated by local factors (pressure from the local labor market) and global factors (the sinuous uptake of semantic technologies and available educational tooling).

Knowledge representation was introduced through a bachelor-level "old-fashioned AI" lecture (built around Prolog and predicate logic), evolving towards a master-level course on Protégé-based ontology engineering, later extended towards RDF-based Linked Data management, then returning to bachelor-level as a Web development course making exploratory use of semantic graph databases, and finally producing master-level spin-offs on Knowledge Engineering and Semantics-driven Engineering topics. This evolution managed, since several years now, to define a research stream of consistent scientific output and active involvement from students and junior researchers. This was also the key performance target for this steady re-design effort, backed by an underlying motivation to enforce a constructivist teaching strategy [1] with as many anchors as possible to what students already know when encountering these topics. In the initial, logic-centered forms of the teaching content, the only anchor was to propositional logic - studied in high school by most students and perceived as being associated with math rather than

ER2023: Companion Proceedings of the 42nd International Conference on Conceptual Modeling: ER Forum, 7th SCME, Project Exhibitions, Posters and Demos, and Doctoral Consortium, November 06-09, 2023, Lisbon, Portugal

robert.buchmann@econ.ubbcluj.ro (R.A. Buchmann); anamaria.ghiran@econ.ubbcluj.ro (A.M. Ghiran)

0000-0002-7385-1610 (R.A. Buchmann); 0000-0001-7890-9386 (A.M. Ghiran)

information systems development. The current version builds on references to databases (from SQL to NoSQL), general Web development (moving away from Java libraries towards PHP, Python, JavaScript), Web interoperability (from REST APIs to SPARQL endpoints), search engine optimization (from traditional SEO to Schema.org), and diagrammatic knowledge capture (conceptual modeling). Logic and reasoning-mechanisms become an upper layer of cognitive mechanisms serving for semantic enrichment in the above contexts already familiar to students, rather than a topic perceived as a niche of academic interest but little pragmatic relevance. The content further feeds into subsequent courses on network/graph analytics or the interplay between knowledge graphs and machine learning.

Diagrammatic conceptual modeling also plays a key role contributing as a means of knowledge graph enrichment, after many years of being taught strictly as visual documentation - typically as chapters/tools of other disciplines. We've previously discussed this by framing conceptual modeling education as a "design problem" in [2] and tackling it there from a Design Science Research perspective, to emphasize two key principles: the dual nature of diagrammatic models (both human-readable and machine-readable) and the potential agility of their metamodels enabling a mediation role recently recognized by the literature [3] and a more generalized model value proposition [4].

The starting point for this journey was a traditional and inertial disconnect between AI topics and the other disciplines of our programs, as well as between knowledge representation (in the sense of symbolic AI) and knowledge acquisition by diagrammatic means. Symptoms of this disconnect included a general sense of disjointedness regarding the study programs and certain oversimplifications in understanding the nature and applicability of conceptual modeling (we discussed these fallacies based on teaching cases in [2] and [5]). The strategy for tackling the situation was to bring forth conceptual modeling as a standalone discipline providing tooling and thinking that can be adopted for diverse purposes and application domains - be it software engineering, business process management, knowledge graph building. This has generally flipped the perception on conceptual modeling - from tooling that belongs to other disciplines to a standalone discipline having diverse application areas, serving them with means of abstraction, complexity reduction and semantic mediation. This is not limited to information systems or enterprise modeling, but also extends to domain-specificity through DSMLs (domain-specific modeling languages) - we presented a teaching artifact for maintaining a repository of cooking recipes in [5].

Secondly, in constructivist spirit we aimed to identify points of convergence and possible bridging between knowledge representation, enterprise modeling and the disciplines already part of the study program stem - i.e. business analysis, software/Web development, databases. Finally, we benefitted from an accelerated commercial visibility and adoption of semantic technologies - moving away from viewing tools as experimental artifacts of "old-fashioned" symbolic AI towards modern databases (graph databases), search engine optimization (through Schema.org), Web development (through RDF-based programming libraries), Web interoperability (as a viable alternative to XML/JSON) and finally converging with diagrammatic modeling as means of enabling machine reasoning over enterprise models (BPMN, Archimate etc.) or as possible mediators for model-driven engineering.

The remainder of the paper is structured as follows: the evolution stages of this curricular re-design are summarized in the next section. Section 3 provides insights about the current structure, design rationale and learning outcomes based on a revised Bloom taxonomy. Section 4 enumerates the teaching tools and key enablers. Section 5 comments on related work on educational experiences and strategies for conceptual modeling. The paper concludes by highlighting some success indicators.

2. Context and evolution stages

One major challenge with teaching knowledge representation and conceptual modeling topics in the host university has been the context of a dominant outsourcing culture of the regional IT

industry, which rewards (and influences through direct involvement in curricula design or even volunteering for teaching activities) the topics that are strictly relevant to their portfolio of IT service provision: software testing, maintenance or customization of legacy systems, development of low-innovation products (templated Web shops, apps, REST interfaces), database or cloud management, document management.

In such a pragmatic context, there have been struggles with preserving curricular modules that do not have a direct correspondent in the prioritized skill profiles - examples of disciplines suffering from this have been Business Process Management (partially revigorated recently by the popularity of RPA), Enterprise Architecture Management, Conceptual Modeling/Model-driven Engineering, Artificial Intelligence (a rollercoaster of "winters" and hype "springs"). A vicious circle started to manifest - of students arguing that there are "no jobs" for certain skills versus companies arguing that there is no talent available in the region for the same skills. Education is responsible for defusing such situations, therefore we took on a long term endeavor to advocate the value proposition for knowledge representation and conceptual modeling.

The Artificial Intelligence topics in the Business Informatics programs have been redesigned many times to optimize their position and relevance, oscillating between formalism-focused and tool-focused. Initially an AI course aimed to balance coverage of symbolic and non-symbolic AI notions in line with the AIMA textbook[2] , but was perceived as disconnected from the rest of the program, from student preferences and prior knowledge or local industry needs. While the original vision was not entirely dropped, a number of revisions have been tested at master level (and pushed to bachelor level once validated and streamlined). A first wave of revisions was applied in 2009 by introducing a Knowledge Management Systems course focusing on ontology engineering with Protégé and OWL/XML-based tooling, backed by description logics foundations, which also met some resistance caused by user experience gaps already recognized by the literature [6][7] or because of the comprehension challenges raised by description logics and ontology formalisms [8][9]. A second wave of revisions followed in 2011-2012 shifting towards the Linked Data paradigm - i.e. a perspective of open data management and federation took precedence, although the graph nature was still obscured by the legacy Prolog-inspired focus on logic, and by the cumbersome RDF/XML standard. During 2015-2016, as commercial tooling and RDF graph visualizers became available to students, the SEO incentive of Schema.org and DBPedia gained traction and RDF programming libraries improved in robustness, this could be pushed closer towards the dominant interest of students - i.e. connected at bachelor level with Web application development.

Similar challenges were met by diagrammatic conceptual modeling - scattered among many disciplines (object-oriented programming, systems design, database design, business process management), modeling was dominantly perceived as visual documentation following some loose guidelines - i.e. missing important aspects such as model-driven engineering or metamodeling. A recent revision turned this into a standalone discipline with diverse application areas, dedicating a full semester to multi-perspective modeling through UML, BPMN, DMN, Archimate and brief introductions to other notations or DSMLs.

Finally, on master level a convergence between the Semantic Web topics and the diagrammatic modeling methods was also introduced - to be detailed in the next section.

3. Content and task designs

The current content design is split between bachelor-level and master-level courses. In the bachelor-level, diagrammatic conceptual modeling and knowledge graphs are covered by parallel courses and kept generally separate, with only brief cross-references between them. On master level, the two types of conceptual modeling converge into knowledge engineering and semantics-driven engineering modules where the interplay between diagrammatic enterprise models, DSMLs (domain-specific modeling languages) and knowledge graphs is exploited. An overview of the redesigned content and tasks is provided in Figure 1, to be detailed in the following.

[2] https://aima.cs.berkeley.edu/

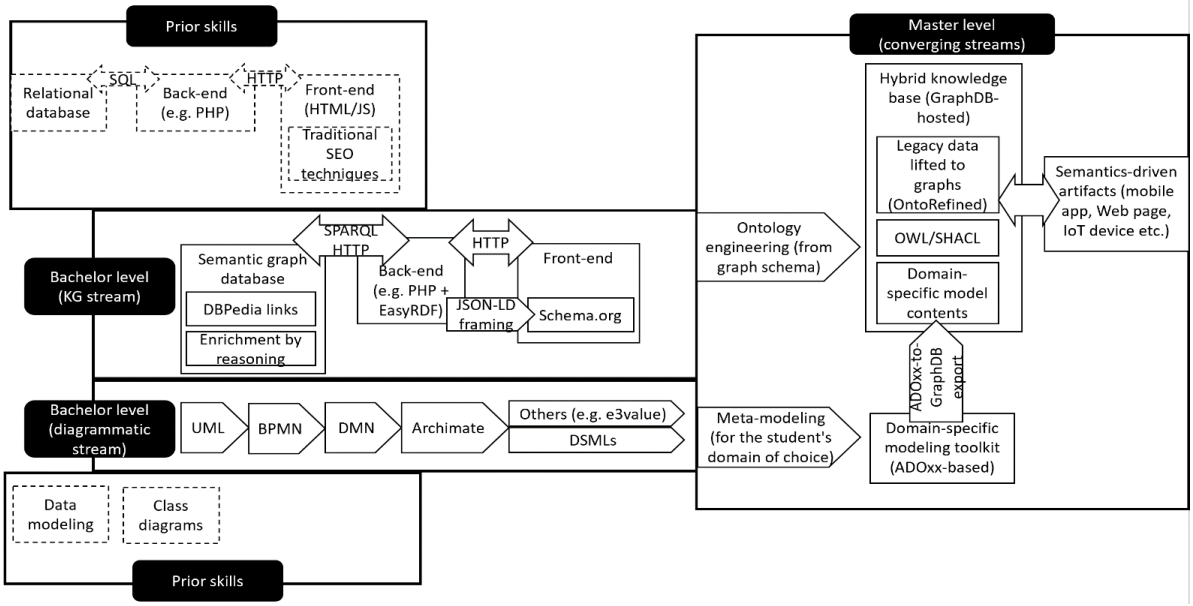

Figure 1 The re-designed content/tasks for conceptual modeling education

The current approach for teaching **knowledge graphs** at bachelor level is a bottom-up one - instead of starting from high-level logical formalisms and knowledge representation foundations (that traditionally never made it into pragmatic examples), we hook into popular disciplines for which students already acquired pragmatic skills. We rely on existing Web development projects where students gained the skills to develop a simple Web shop using a traditional technological stack - JavaScript-PHP-MySQL, in a context also touching on e-business principles and search engine optimization. On this pre-existing skillset, students are tasked to gradually incorporate granular knowledge graph ingredients from a Web developer perspective, in an additive manner:

1. First, the MySQL database of the Web shop is replaced with an RDF triplestore holding equivalent data (Web shop-oriented, i.e. users, products, orders etc.), which gives opportunities for several early "revelations" induced by analogies and differentiators: how tabular structures are mapped to graph structures; how primary keys compare to URIs; how table JOINS compare to graph path navigation; how a relational schema compares to a graph schema. The discussion focuses on database concepts already familiar to students, intentionally avoiding any AI-related background or open world assumptions;
2. SQL queries in the Web shop are replaced by equivalent graph queries without breaking the legacy functionality. The simplest PHP library is used to run the queries[3], which mimics the way students are used to run SQL queries in PHP. Later the low-level HTTP details of using the SPARQL HTTP protocol are also revealed - this time by analogy with general HTTP interoperability;
3. Then, the HTML front-end is dynamically enriched by JSON-LD graph fragments using Schema.org mark-up[4], which gives the opportunity to discuss novel approaches to semantic SEO and search engine driven interoperability. It is also an opportunity to showcase the ease by which JSON-LD graph fragments can be injected into Web pages if the back-end store is already graph-based (with additional tricks such as JSON-LD framing[5] to obtain a targeted JSON structure);
4. The Linked Data aspect is then revealed by incorporating DBPedia links, and expanding a few of the queries to bring federated data to the existing front-end;

[3] https://www.easyrdf.org/
[4] https://schema.org/
[5] https://www.w3.org/TR/json-ld11-framing/

5. Finally, basic machine reasoning is introduced by a few simple SPARQL inferences to generate information not present in the initial data store - e.g. generating networks of users/actors involved with the same products/items.

This flow ensures not only that students work on code and patterns they already developed before, but that at every step there's an opportunity for analogies, comparisons and anchoring to patterns they are familiar with, before spiraling away from them. Only a superficial AI framing is provided in the bachelor-level theoretical lectures, the general focus being on providing a natural extension for Web developers towards alternative databases, novel querying and SEO techniques, new types of objects manipulated in Web scripts (e. g. graph objects in PHP).

In parallel, the stream of **diagrammatic conceptual modeling** is covered by a separate course that primarily provides training on modeling standards (predominantly UML, BPMN, DMN, Archimate). It also hints towards research-driven languages (i*, e3value) as well as DSMLs. The objective here is manifold:

1. to reveal the diversity of modeling languages in relation to diversifying purposes, to discuss their occasional overlapping or semantic divergence, and finally their inherently limited competence (limited by a constraining metamodel);
2. to reveal the notion of "model queries" (as a flavor of "competence questions") - i.e. means of retrieving contents from a repository of models; this can be demonstrated either by XPath over the XML serializations provided by most standards, or by dedicated model query languages such as AQL in the Bee-Up modeling tool[6] ;
3. to detach conceptual modeling from the software engineering domain where it is previously used by other courses (through class diagrams and data models);
4. as a consequence of all the above, to position conceptual modeling as a standalone discipline supporting knowledge structuring and retrieval for any application areas.

The two streams (diagrammatic modeling and non-diagrammatic knowledge graphs) converge on master level along two modules building both abstraction and engineering skills:

a) In **Knowledge Engineering** we discuss similarities and possibilities of interplay between ontologies and metamodels. OWL ontologies and inference rules come now into focus (expanding from the earlier graph database schema perspective), with foundational background on description logics and pragmatic examples of axioms/rules over the previously developed graph-driven Web project;

b) In **Semantics-driven Engineering** we introduce means of engineering artifacts that retrieve knowledge by semantic queries or reasoning applied over a semantic repository of hybrid content consisting of (a) legacy datasets semantically lifted (with transformation rules in the OntoRefine tool[7]), (b) OWL/SHACL axioms and rules, (c) diagrammatic contents converted to RDF - initially from established model types available in the Bee-Up modeling tool (BPMN, DMN, UML etc.) and later from model types pertaining to DSMLs developed by students to support their preferred domain and competence questions. This hybrid knowledge graph is hosted by an OWL-enabled repository on GraphDB[8] and exposed to semantics-driven artifacts developed by each student according to their engineering preferences (mobile apps, Web pages, IoT devices). Many repetitions of this engineering approach have been crystallized in a specific flavor of model-driven engineering that we've discussed in more detail in [10] under the label of "model-aware engineering" and can be clearly distinguished from traditional model-driven approaches where a fixed metamodel makes possible stable transformation rules.

[6] https://www.adoxx.org/live/adoxx-query-language-aql
[7] https://www.ontotext.com/products/ontotext-refine/
[8] https://www.ontotext.com/products/graphdb/

Table 1 summarizes the targeted outcomes that are basis for exam evaluations aligned with the revised Bloom taxonomy of learning levels proposed in [11]. Each cell lists outcomes for both the knowledge graph and the diagrammatic modeling stream.

Table 1 Outcomes according to the Bloom (revised) taxonomy

Taxonomy level	Bachelor-level outcomes	Master-level outcomes
Remember	The ability to recall and give informal definitions on basic RDF-related concepts: URI, blank nodes, triple, predicate etc. The ability to recall whether a modeling standard offers or not certain concepts (i.e. matching-based, not recalling the full lists of concepts available in each modeling standard).	The ability to recall and give informal definitions on basic OWL-related concepts: inverse property, symmetric property etc. The ability to recall and give informal definitions on the building blocks of a modeling method or Agile Modeling Method Engineering phases cf. [12].
Understand	The ability to describe in natural language a situation (and associated data) depicted in an RDF graph written in Turtle. The ability to describe in natural language a situation or pattern depicted in a diagrammatic model (using the notions previously mentioned)	The ability to describe in natural language the knowledge structure depicted by an OWL ontology and a set of SPARQL/SHACL rules. The ability to describe in natural language a metamodel depicted as a class diagram. The ability to draw a mock diagram conforming that metamodel.
Apply	The ability to "translate" a piece of natural text into an RDF graph written in Turtle, with an RDFS schema and a minimal number of terms adopted from Schema.org. The ability to run queries on that. The ability to represent a given situation or pattern in a diagrammatic model, while preserving as much as possible of the details provided in the textual description.	The ability to define the RDFS+OWL knowledge structure capturing all relationships and classes involved in a natural language description of a situation. The ability to draw the domain-specific metamodel capturing all relationships and types involved in a natural language description of a situation.
Analyze	The ability to assess whether a certain SPARQL query can be satisfied by an RDF graph exemplar - given in Turtle or JSON-LD formats. The ability to assess whether a competence question can be satisfied by the contents of a diagrammatic model.	The ability to assess whether an RDF triple will be generated or not by a certain mix of OWL axioms and SPARQL/SHACL rules. The ability to assess whether a competence question can find its answer in models created according to a given domain-specific metamodel (given a legend a mock symbols).
Evaluate	The ability to find syntactic and structural mistakes in RDF graph exemplars (written in Turtle), relative to a situation it is supposed to represent.	The ability to formulate all explicit triples that will be generated from a given RDF dataset by a given set of OWL axioms and SPARQL/SHACL rules.

	The ability to find mistakes in diagrammatic models, relative to a situation or pattern they are supposed to represent (same modeling languages mentioned above).	The ability to assess whether a domain-specific diagram deviates from a domain-specific metamodel it is supposed to be governed by.
Create	The ability to build a Web site that populates and stores all data in a knowledge graph, including links to DBPedia and a graph fragment published as JSON-LD in the front-end. The ability to create a multi-perspective cross-consistent set of models (of different model types) depicting different facets of a complex situation or architecture.	The ability to implement a novel modeling tool deploying a DSML. The ability to use it to describe at least two different situations, to ensure it is not a "single use" (i.e., too specific) language. The ability to build a knowledge graph incorporating the models depicting those situations, with additional semantic enrichment and data linked to them. The ability to build a front-end that demonstrates for this knowledge graph the application of reasoning, model navigation and data aggregation constrained by model contents.

4. Key technological and organizational enablers

The tooling required to deploy the new course designs consists of two ecosystems: (i) for the knowledge graph management part, the tooling around Ontotext's GraphDB was chosen; (ii) for the modeling, metamodeling and model-to-RDF interoperability, OMiLAB's Digital Innovation environment [12] was adopted.

Ontotext's GraphDB was chosen due to the availability of a free edition that students can immediately get hands-on experience with and, equally important, due to the commercial credibility as a production-ready system offering rich connectors, plug-ins and APIs that can be accessed from different programming languages. Earlier tools have been perceived by students as "professor-ware" (irrelevant outside academic context) or locked into a Java ecosystem (for the early Semantic Web tools). Usability, visualization and easy configuration features of GraphDB made it a key ingredient earlier than competing products became available, and licensed versions were also successfully adopted in institutional projects with scaling requirements that can demonstrate to students production-readiness [13].

The OMiLAB Digital Innovation environment is a digital ecosystem and a hardware-software installation providing a complex toolset: (a) the previously mentioned **Bee-Up**[9] - out of the box modeling tool for BPMN, ER, EPC, UML, Petri Nets, DMN and other languages. The tool also provides an **RDF export** for any of the supported languages, allowing several layers of semantic enrichment of diagrammatic elements; (b) **the ADOxx metamodeling platform** [10] - for implementing modeling tools and experimenting not only with DSML, but also with the inner workings of Bee-Up to enable design-oriented research (such as [14] where BPMN was specialized to describe user experience flows), or empirically-oriented research that requires the logging of modeling actions as in [15]. **An RDF export for any DSML deployed on ADOxx** is openly available[11], based on representation and reasoning patterns discussed in [16]; (c) a number of hardware (robotic) components and adapters to enable interoperability between models and Internet of Things environments, which further provide input to an IoT course that

[9] https://bee-up.omilab.org/activities/bee-up/
[10] https://www.adoxx.org/live/home
[11] Tool available at https://code.omilab.org/resources/adoxx-modules/rdf-transformation

is not in the scope of this paper - we only mention its relation to a number of conference tutorials presented in recent years[12] . The ecosystem and tooling hereby summarized was collected over the years based on exploratory teaching in several contexts: (a) interactions facilitated by the NEMO summer school series and the enterprise modeling community involved there[13]; (b) recent adoption in our university of an OMiLAB node [14] and its digital innovation toolkits; (c) coordination of the master program on Business Modeling and Distributed Computing[15], where most of the hereby reported curricular revisions were initially tested, before transferring some of the modules to bachelor level.

5. Related work

We incorporate under the notion of "conceptual modeling education" the efforts pertaining to both diagrammatic (using UML, BPMN etc.) and non-diagrammatic (i.e. ontologies, knowledge graphs) conceptual modeling. The two categories are rarely investigated in convergence and even rarer taught in tandem, because of limited tooling - except for the previously mentioned support in Bee-Up and ADOxx, recent research reported tools that may be employed for comparable teaching tasks: AOAME [17], Archi's plug-in for Neo4J [18], EAKG [19], earlier works hinting at specific demonstrators for business process modeling [20]. A recent systematic mapping on the convergence of conceptual modeling and the Semantic Web provides a comprehensive inventory [21]. We could not identify teaching reports on using such tools, nor on knowledge graph development education - scholarly work focuses on adoption of knowledge graphs for educational knowledge management. Related works have covered comprehension challenges regarding ontology formalisms such as frames or description logics [8][9].

On the other hand, research on the education of diagrammatic conceptual modeling is much better represented: [22] analyzed student modeling tasks to quantify modeling errors, [15] investigated modeling styles, [23] proposed a Bloom-based framework for teaching conceptual modeling and metamodeling, [24] advocated several course re-designs to strengthen the position of conceptual modeling as a standalone discipline, [25] discussed the need for animated notations to improve BPMN diagrams comprehension.

6. Connections to learning theories

Although we did not derive novel learning theories from this experience, we can point to existing theories that are embodied in the reported approach (and were previously ignored by the legacy approach).

First of all, constructivist learning [1] posits that students should build on what they already know and should derive knowledge by their own construction effort rather than by direct assimilation of content. The initial "old-fashioned AI" course built only on priors related to propositional logic (introduced to most students in high schools and not revisited afterwards) and was perceived as being mostly disconnected from the rest of the curriculum or software engineering practices employed by the local industry. The new approach successfully hooks into (and extrapolates from) already familiar technologies and practices: databases, Web development and architecting, SEO, diagrammatic modeling. Students extend code they already developed with granular ingredients that are able to frame knowledge graphs more naturally, as learning objects connected to those already acquired.

Secondly, the spiraling strategy advocated by J. Bruner [26] inspired a recurring revisitation of the same topics throughout the curriculum. Such spiraling was already manifesting by somewhat redundant revisitation of certain more mainstream topics (relational databases, SQL queries, Web development presented in different flavors and tools, by different courses or

[12] See the ER tutorial at https://er2023.inesc-id.pt/program-overview/tutorials/#tutorial3
[13] NEMO summer school series, https://nemo.omilab.org/
[14] OMiLAB-FSEGA node, https://econ.ubbcluj.ro/omilab/index.php
[15] BMDC master program, https://econ.ubbcluj.ro/programe/bmdc/index.php

modules), traditionally converging into bachelor or master theses. The current approach hooked into these spirals by presenting knowledge graphs and conceptual modeling as natural spin-offs of those dominant topics rather than disconnected disciplines or parallel content streams.

7. Concluding evaluation

The main goal of this longitudinal redesign was to enable, in the spirit of the humboldtian education model adopted by the university, a steady stream of scientific research from students - thus making them more prepared for emerging technologies, for innovation-oriented entrepreneurship (in contrast to a legacy outsourcing culture) or for junior research work in projects and PhD programs. None of this was happening in the areas of symbolic AI and model-driven engineering prior to 2016, when the second major curricular redesign was applied. Since then, conference publications based on master dissertations became regular, in venues such as ENASE, REFSQ, AMCIS, ISD, ECIS etc.[16] The first local start-up that produces a domain-specific (for cybersecurity) knowledge graph with a diagrammatic layer emerged in the region[17] and a number of companies known as early adopters of knowledge graphs opened off-shore offices in the region, creating further cooperation opportunities. Institutional projects became possible, employing talent already available among students without the need for a risky and expensive learning curve. One example of institutional project deals with the semantic lifting of legacy databases available in the university, for master data management [13].

In terms of weaknesses, there is still a disconnect from non-symbolic AI topics such as deep learning and natural language processing, which we aim to bridge in future revisions by developing demonstrators for neuro-symbolic AI and by exploiting the recently launched interfaces of GraphDB to OpenAI[18].

References

[1] V. Richardson, Constructivist Teaching and Teacher Education: Theory and Practice. In Richardson, V. (ed.): Constructivist Teacher Education: Building a World of New Understandings, Routledge, 1997, pp. 3–14.

[2] R. A. Buchmann, A. M. Ghiran, V. Döller and D. Karagiannis, Conceptual modeling education as a "design problem". Complex Systems Informatics and Modeling Quarterly, 21 (2019) 21-33. doi: 10.7250/csimq.2019-21.02.

[3] J. Recker, R. Lukyanenko, M. Jabbari, B. M. Samuel, A. Castellanos, From representation to mediation: a new agenda for conceptual modeling research in a digital world. MIS Quarterly, 45(1) (2021) 269-300. doi: 10.25300/MISQ/2021/16027.

[4] A. M. Ghiran, C. C. Osman, R. A. Buchmann, Advancing conceptual modeling education towards a generalized model value proposition, in: Advances in Information Systems Development, LNISO 39, Springer, 2020, pp. 1-18. doi: 10.1007/978-3-030-49644-9_1.

[5] R. A. Buchmann, A. M. Ghiran Engineering the cooking recipe modelling method: a teaching experience report. In: CEUR-WS 1999 (2017) https://ceur-ws.org/Vol-1999/paper5.pdf.

[6] H. Knublauch, M. Horridge, M. Musen, A. Rector, R. Stevens, N. Drummond, P. Lord, N. F. Noy, J. Seiedenberg, H. Want, The Protege OWL Experience, in Proceedings of OWLED 2005, CEUR-WS 188 (2005) https://ceur-ws.org/Vol-188/sub14.pdf.

[7] A. Rector, N. Drummond, M. Horridge, J. Rogers, H. Knublauch, R. Stevens, H. Wang, C. Wroe, OWL pizzas: practical experience of teaching OWL-DL: common errors and common patterns, in Proceedings of EKAW 2004, LNAI 3257, Springer, 2004, pp. 63-81, https://doi.org/10.1007/978-3-540-30202-5_5.

[16] A list of student papers is maintained at https://econ.ubbcluj.ro/omilab/publications.php

[17] See https://cyscale.com/products/security-knowledge-graph/

[18] See https://graphdb.ontotext.com/documentation/10.3/gpt-queries.html

[8] B. Dizza, B.-L. Shiry, A controlled experiment of teaching ontology formalisms, Human systems Management 35(3) (2016) 213-228. doi:10.3233/HSM-160870.
[9] P. Warren, P. Mulholland, T. Collins, E. Motta, Improving comprehension of knowledge representation languages: a case study with description logics, International Journal of Human-Computer Studies 122 (2019) 145-167. doi: 10.1016/j.ijhcs.2018.08.009.
[10] R. A. Buchmann, M. Cinpoeru, A. Harkai, D. Karagiannis, Model-aware software engineering - A knowledge- based approach to model-driven software engineering. Proceedings of ENASE 2018, ScitePress, 2018, pp. 233-240, doi: 10.5220/0006694102330240.
[11] D. R. Krathwohl, A revision of Bloom's taxonomy: An overview. Theory into practice 41(4) (2002) 212–218. doi: 10.1207/s15430421tip4104_2.
[12] D. Karagiannis, R. A. Buchmann, W. Utz, The OMiLAB Digital Innovation environment: agile conceptual models to bridge business value with Digital and Physical Twins for Product-Service Systems development. Computers in Industry, 138 (2022) 103631.
[13] R. A. Buchmann, R. Dragoș, A. M. Ghiran, From the application-centric to the knowledge-centric university: a design science proof-of-concept. In Proceedings of ISD 2021, AIS eLibrary, 2021, https://aisel.aisnet.org/isd2014/proceedings2021/currenttopics/14/.
[14] Ș. Uifălean, Employing knowledge graphs for capturing semantic aspects of robotic process automation. Proceedings of CAiSE 2023 Workshops, Springer, 2023, pp. 152-162.
[15] K. Rosenthal, J. Wagner, S. Strecker, Modeling styles in conceptual data modeling: reflecting observations in a series of multimodel studies, in Proceedings of ECIS 2022, AIS eLibrary, 2022, https://aisel.aisnet.org/ecis2022_rp/73/.
[16] R. A. Buchmann, D. Karagiannis, Pattern-based transformation of diagrammatic conceptual models for semantic enrichment in the Web of Data, in Proceedings of KES 2015, Procedia Computer Science 60, Elsevier, (2015) pp. 150-159. doi: 10.1016/j.procs.2015.08.114.
[17] E. Laurenzi, K. Hinkelmann, A. van der Merwe, An agile and ontology-aided modeling environment, in Proceedings of PoEM 2018, LNBIP 335, Springer, 2018, pp. 221-237. doi: 10.1007/978-3-030-02302-7_14.
[18] ***, Archi database plug-in, https://github.com/archi-contribs/database-plugin/wiki
[19] P. L. Glaser, J. A. Syed, E. Sallinger, D. Bork, Exploring enterprise architecture knowledge graphs in Archi: the EAKG toolkit, in Proceedings of EDOC 2022 workshops, LNBIP 466, 2023, pp. 332-338, Springer. doi: 10.1007/978-3-031-26886-1_21.
[20] O. Thomas, M. Fellmann Semantic process modeling – design and implementation of an ontology-based representation of business processes. Bus Inf Sys Eng 1(6) (2009) 438-451
[21] C. Th. Eggerth, Conceptual modeling and Semantic Web: a systematic mapping study, TU Vienna, 2022. doi:10.34726/hss.2022.99981.
[22] D. Bogdanova, M. Snoeck, Domain modelling in Bloom: deciphering how we teach it. In Proceedings of PoEM 2017. LNBIP 305, Springer, 2017, pp. 3-17. doi: 10.1007/978-3-319-70241-4_1.
[23] D. Bork, A framework for teaching conceptual modeling and metamodeling based on Bloom's revised taxonomy of educational objectives. In: Proceedings of HICSS 2019, AIS eLibrary, 2019, https://aisel.aisnet.org/hicss-52/set/methods_models/7/.
[24] S. Strecker, U. Baumol, D. Karagiannis, A. Koschmider, M. Snoeck, R. Zarnekow, Five inspiring course (re-)designs. Business & Information Systems Engineering, 61(2) (2019) 241-252. doi: 10.1007/s12599-019-00584-5.
[25] I. Maslov, S. Poelmans, Advancing the BPMN 2.0 Standard with an Extended Animated Notation: A Research Program for Token-Based Process Modeling Education, Proceedings of BIR 2023 Workshops and Doctoral Consortium, 2023, in press.
[26] J. S. Bruner, The process of education. Cambridge, MA: Harvard University Press, 1960.