Complex Systems Informatics and Modeling Quarterly (CSIMQ)
eISSN: 2255-9922
Published online by RTU Press, https://csimq-journals.rtu.lv
Article 122, Issue 21, December 2019/January 2020, Pages 21–33
https://doi.org/10.7250/csimq.2019-21.02

Conceptual Modeling Education as a "Design Problem"

Robert Andrei Buchmann[1*], Ana-Maria Ghiran[1],
Victoria Döller[2], and Dimitris Karagiannis[2]

[1] Business Informatics Research Centre, Faculty of Economics and Business Administration, Babeș-Bolyai University, Cluj-Napoca, Romania
[2] Research Group Knowledge Engineering, Faculty of Computer Science, University of Vienna, Vienna, Austria

robert.buchmann@econ.ubbcluj.ro, anamaria.ghiran@econ.ubbcluj.ro,
victoria.doeller@univie.ac.at, dimitris.karagiannis@univie.ac.at

Abstract. This article frames Conceptual Modeling education as a design problem, in the sense of the Design Science research framework, motivated by student preconceptions and oversimplifications causing a gap between how the discipline is perceived at bachelor level and the holistic understanding of model value that is required for research work. The treatment to this design problem must comprise teaching approaches and artifacts capable of positioning Conceptual Modeling as a standalone discipline having a value proposition for any application domain, rather than a technique subordinated to other disciplines. The underpinning thesis is that modeling languages should be primarily understood as purposeful knowledge schemas that can be subjected to agile adaptations in support of model-driven systems or knowledge processes, by analogy to how a database schema is evolved in response to changing requirements of a data-driven system or data analytics needs. This thesis is supported by enablers provided by the Open Models Laboratory and the Agile Modeling Method Engineering framework – resources that support the development of treatments to the design problem framed by the article.
Keywords: Conceptual Modeling Education, Design Science, Agile Modeling Method Engineering, OMiLAB, Teaching Artifacts.

1 Introduction

This article extends the ideas and content presented as a position paper in [1], discussing the oversimplified perception of bachelor students on Conceptual Modeling topics and proposing position statements supported by teaching approaches aimed to shift this perception. The shift is

* Corresponding author

Reference: R. A. Buchmann, A. M. Ghiran, V. Döller, and D. Karagiannis, "Conceptual Modeling Education as a "Design Problem"," Complex Systems Informatics and Modeling Quarterly, CSIMQ, no. 21, pp. 21–33, 2019. Available: https://doi.org/10.7250/csimq.2019-21.02

Additional information. Authors' ORCID iD: R. A. Buchmann – https://orcid.org/0000-0002-7385-1610, A. M. Ghiran – https://orcid.org/0000-0001-7890-9386, V. Döller – https://orcid.org/0000-0002-5770-635X, and D. Karagiannis – https://orcid.org/0000-0003-2162-9740. PII S225599221900122X. Received: 2 November 2019. Accepted: 27 December 2019. Available online: 31. December 2019.

required in order to align students' understanding with what is expected from them in research work (doctoral level or project-based), where Conceptual Modeling is often employed as a standalone discipline with diverse application areas and theoretical lenses, instead of reducing it to the dominant understanding (of bachelor students) that it is a Software Engineering chapter providing graphical means for documenting requirements, software or process design.

Considering this a Design Science problem, we can frame it according to the design problem template and research framework presented in [2]:

Improve student understanding of Conceptual Modeling possibilities
by supporting it with dedicated teaching strategies and educational resources
such that their oversimplified understanding at bachelor level is enriched
in order to ease their learning curve and facilitate productivity in research work.

Like any design problem, it is anchored in *requirements* which emerge from oversimplifications detected in the perception of bachelor students. These are compensated in this article by several position statements, further supported by teaching strategies employed as "treatments" to the problem.

As a working definition within this educational context, we see Conceptual Modeling as a standalone discipline that uses or creates conceptualizations for any domains – i.e. either concepts are the building blocks of a model (e.g. class diagrams, ER diagrams) or their instances are the model elements (e.g. business process model, enterprise architecture models, domain-specific models). We limit our discussion to diagrammatic Conceptual Modeling, since non-diagrammatic approaches (e.g. ontology engineering) are typically met in separate courses for which this article's position statements must be reconsidered – some links will be, however, established, on the grounds of machine readability of conceptual models.

Within the stated scope, recent research on Conceptual Modeling education appears to be dominated by Software Engineering scenarios, e.g.: an approach to measuring domain modeling logs has been developed in [3]; another approach to quantifying UML class diagram creation errors was proposed in [4]; students implemented a custom variant of an ER modeling method in [5]; a teaching experience report addressed the modeling tasks of system architects in [6]. A recent literature survey [7] concludes that about three quarters of publications on Conceptual Modeling education focus on object-oriented and data modeling – this can be linked with the fact that most bachelor students in Business Informatics / Information Systems study programs face their first modeling tasks in the context of software design and database design, a natural consequence of the joint curricula for Information Systems established by ACM and AIS [8]. Even papers on teaching Enterprise Modeling are scarce, with some notable but isolated works, e.g. [9]. One direct consequence of this state of affairs is that students graduate bachelor programs with the perception that Conceptual Modeling is a chapter/technique of Software Engineering and has no relevance outside the goals associates with that discipline; Business Process Management is gaining some attraction in this respect (less than a quarter of the surveyed education-focused papers according to [7]), however it's still not sufficient to establish a general notion of "model value" that can inspire junior research work taking on Conceptual Modeling as a standalone discipline that uses or creates conceptualizations for any domains. This is what we consider to be the "design problem of Conceptual Modeling education" – tackled here with a number of position statements mapped on requirements, some examples of treatments of the problem and a value proposition that relies on community involvement.

The yearly NEMO (Next Generation Enterprise Modeling) summer school [10] has been a venue for raising awareness on Conceptual Modeling educational resources and their application possibilities, and discussion at these events helped synthesise the requirements derived in Section 4, further motivating the development of teaching artifacts to alleviate the problem – a few examples will also be referenced in this respect.

The remainder of the article is structured as follows: in the next Section we will present the design problem context, followed in Section 3 by an inventory of students' oversimplified

understanding – from which teaching requirements are derived and mapped on possible treatments in Section 4. In Section 5 we refer to one key enabler for the proposed position statements – the OMiLAB (Open Models Laboratory) digital ecosystem [11], [12] which successfully supports the holistic understanding of "model value" through an open community approach and open use educational artifacts.

2 The Problem Context and Stakeholders

The teaching experience on which this article is based involves bachelor students in the Business Informatics domain, who get in contact with Conceptual Modeling method during the last two semesters of their bachelor program. The gap between how they experience modeling and what is expected from them in Ph.D. programs (both Computer Science and Business Administration) is the basis for the hereby introduced design problem. The teaching angle to be presented here as a "treatment" has been applied in the first author's institution, to Business Informatics master level programs during two semesters: one focusing on students becoming familiar with the diversity of modeling languages and modeling purposes; the second focusing on modeling method engineering and raising awareness on domain-specific model value (this is where modeling tool development is included). The only precondition is that master students should have finished a bachelor program on Business Administration, Business Informatics (where they come in contact with modeling aspects of Business Process Management) or Computer Science (where they have initial contact with modeling methods for Software Engineering).

Teaching Conceptual Modeling is not a novelty – one can identify programs where it is introduced as a discipline by itself, with dedicated teaching materials [13][14]. This article's contribution is a new Design Science lens applied on teaching Conceptual Modeling, bringing the benefit of explicit requirements analysis and strategies that target those requirements. Traditional approaches to teaching Conceptual Modeling take such requirements as tacit/implicit, resorting to standards and established practices to tackle the most common modeling use cases and purposes.

The teaching angle advocated by the article aims to defuse certain inertia and ambiguity in how Conceptual Modeling languages are understood – by students, by some practitioners, and by junior researchers who do not have an engineering perspective on the nature and constituents of a modeling language or method (e.g. the building blocks proposed in [15]). Below we provide the list of dilemmas collected from students debuting with junior research work and/or dissertation theses on topics related to Conceptual Modeling:

My thesis is on Marketing – specifically Service Design and Service-Dominant Logic – how can Conceptual Modeling help me, since I know it as being a Software Engineering technique (this typically being the first contact of students with modeling tasks)*? Why are there so many modeling languages? Why not use PowerPoint, since I have many more graphical shapes available in a single tool? Isn't it possible to model everything with a single language/standard? How can I combine parts of different modeling languages in an integrated way? How could I represent "this"* (domain-specific thing) *with my preferred standard?*

Answers to these questions are well understood and considered implicit by experienced researchers, but not easily available in explicit form to debutants – a gap that we should bridge with dedicated teaching artifacts. The gap manifests when students start doing research work and they find themselves pushed towards a variety of paradigms – Design Science, Knowledge Management, Enterprise Modeling, Service Engineering, etc., which, typically, raise the level of abstraction and formulate modeling requirements above the use cases of graphically documenting software projects (in the authors' experience and institutions, this is the most common modeling use case among bachelor students in Information Systems studies). Upon initial contact with the mentioned paradigms junior researchers engage in a learning curve that significantly shifts their understanding on the nature and value of models and raises their

awareness on Conceptual Modeling requirements. Engaging in the learning curve typically delays their productivity in research work or even raises contradictions and misconceptions.

This learning curve must be eased to help them operationalise model value and accommodate the perspective shift towards the following proposed thesis: *Conceptual Modeling is not a technique subordinated to a particular application domain; instead, it has its own compelling value proposition in research, practice and education regardless of application domain.* The NEMO summer school introduces Conceptual Modeling to students with the following slogan: "We use abstraction to reduce complexity in a domain, for a specific purpose". This slogan is detached from typical application areas of modeling (e.g. Software Engineering, Business Process Management), while shifting the rationale of Conceptual Modeling to the general characteristics of *domain complexity* and *purposefulness*. It is a slogan that triggers interesting questions and self-reflection from the students' side, some of them inspiring the position statements to be formulated in Section 3.

However, the slogan needs to be operationalised in order to remove entry barriers for novices who want to assimilate Conceptual Modeling as part of their complexity management and digitization skillset. They are the stakeholders of this design problem, together with managers and institutions that need to supervise their work and would thus benefit from an eased learning curve and improved productivity when employing or investigating Conceptual Modeling methods for project-specific purposes. As an initial step in pursuing these goals, we synthesised the requirements for the hereby proposed "design problem" from several traditional oversimplifications detected in how bachelor students understand the value and applicability of conceptual models. Master programs can benefit from this article's recommendations by adopting the position statements and by getting involved with the community hub at Open Models Laboratory [11] where various treatments for this design problem are being accumulated and shared.

3 Problem Refinement and Position Statements

From past teaching experience we have extracted several "oversimplifications" by which students and junior researchers limit their own understanding (and research opportunities) when dealing with complex questions related to Conceptual Modeling. We address them through corresponding position statements that can generate insight, stimulate lateral thinking and force a clash between existing preconceptions and hands-on experience. Later, in Section 4 and Section 5, an overview of educational resources in support of these position statements is provided – i.e. usable artifacts being engineered in response to the hereby proposed design problem are discussed.

Oversimplification 1. *Conceptual Modeling is a form of graphical documentation – i.e. it produces visual representations that convey some meaning*. This interpretation is confirmed in the literature that advocates Conceptual Modeling "for the purposes of understanding and communication" [16] and is propagated among students by the common task of having their software projects documented in diagrammatic form. However, these documentations employ most often semantically poor drawing software rather than modeling tools, lacking in terms of metamodel enforcement or any model-driven features. Our position statement to compensate for this perception is that *Conceptual Modeling produces knowledge structures that can have a visual manifestation*. With the term "knowledge structure" we point to several defining qualities, by forcing an analogy with the database world:

- Conformance to a "knowledge (representation) schema" – i.e. each model element is an instance of a prescribed concept in a semantically consistent way (e.g. a dotted arrow should not change meaning from one use to another in the same type of model);
- This conformance further enables "model queries" – a term we use for any form of model content retrieval (as the basis for the development of model-based functionality, reporting, etc.). Examples of model queries can be formulated by analogy with the more familiar "data

queries" – e.g. for a BPMN diagram: *give me all tasks following this particular decision made in my department.* Thus, we argue that a modeling language provides a schema for a model repository – an analogy with traditional databases that students easily grasp (and that can be further extrapolated towards the other oversimplifications discussed here);

- Models can be serialized in ways that preserve not only their visual appearance, but also machine-readable semantics – from traditional standard serializations such as XPDL to the more recent proposal of RDF graphs for model-driven data fabrics [17].

Oversimplification 2. *Modeling languages are vocabularies fixed to serve some consensus.* This interpretation is supported by the availability of popular standards – however even standards enable some level of customization (e.g. UML stereotypes [18]). Moreover, the diversity of standards, showing at the same time conceptual overlapping and purposeful specializations, suggests that a "one size fits all purposes" vocabulary is not realistic. Therefore, our position statement is that *modeling languages are knowledge schemas that can be tailored to satisfy purposeful (possibly evolving) requirements.* Going back to the database analogy, a database schema may be taken for granted, sufficiently stable for a large community of users who interact with it on content (data) level; however, requirements will occasionally trigger schema changes in order to support the evolution of information systems or decision processes. A modeling language can be perceived through a similar lens, as suggested in Figure 1, where a query-able model repository is presented as complementing a traditional database and the modeling software plays the role of "knowledge editor" for that repository.

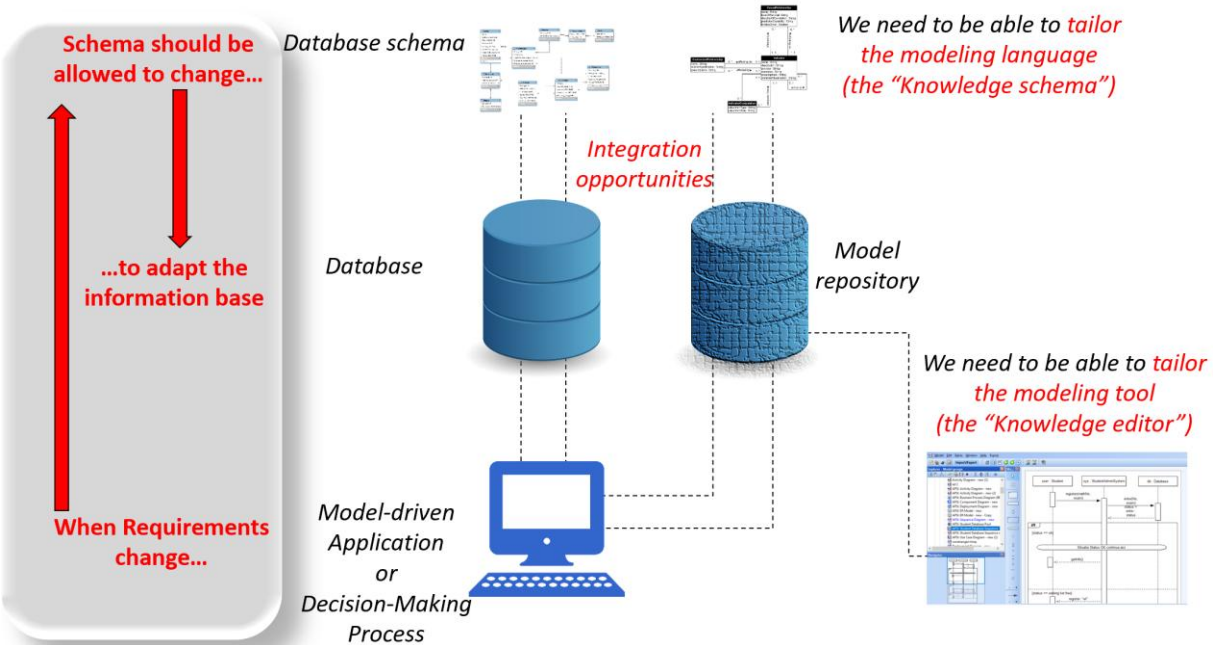

Figure 1. The Database/Model repository analogy as rationale for agility in modeling languages

Oversimplification 3. *Conceptual Modeling is a set of techniques subordinated to Software Engineering* (or another discipline that provides initial contact with a modeling language). Our position statement is that *Conceptual Modeling can be applied to any domain where complexity must be managed through abstraction and structuring*. We encourage students who develop models having no explicit relation to Software Engineering to apply a modeling lens to their work and to reflect on the value proposition that Conceptual Modeling brings to their domain.

For instance, students with a background in Marketing may adopt modeling tools available for their field (e.g. Product-Service system modeling [19], Value modeling [20]); or, by taking a Design Science approach, they may propose their own abstractions relevant to their field (e.g. Service, Customer) consolidated in a customized modeling language that is detached from Software Engineering and acts mainly as knowledge acquisition and decision support. To

connect this with the previous points, such abstractions can be guided by model queries as means of information retrieval and model analysis.

Oversimplification 4. *Whatever needs to be modeled, I can do it with language X (no need for other languages).* Our position statement is that the claim "I can model everything" commonly means "Whatever I cannot model, *I will compensate by (i) squeezing unstructured information into labels/annotations; or by (ii) hacking semantics*".

Figure 2 indicates such cases for a solution given by students who were asked to use BPMN to model a cooking recipe – see the two prominent ways in which they deal with the absence of domain-specific concepts (*Ingredient*) or properties (*Quantity*). The examples generate obvious complications when resorting to "model queries" (e.g. running AQL queries in BEE-UP [21]) and the solution of "language redesign" by adding missing concepts can be proposed as a form of agile schema adaptation with the help of fast prototyping support (i.e. using metamodeling platforms).

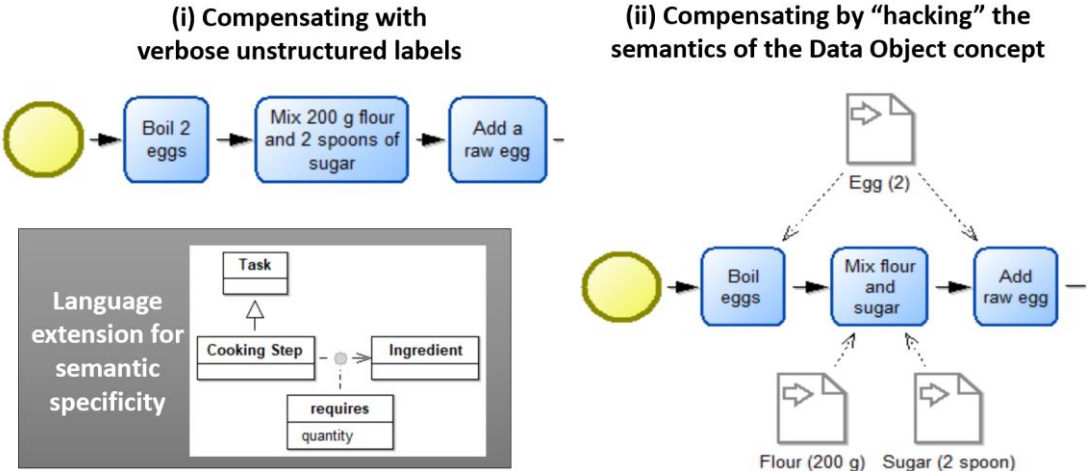

Figure 2. Compensating for the lack of domain-specific expressivity – examples

Oversimplification 5. *Model value is created solely by modelers.* This interpretation finds confirmations in Business Analyst jobs where modeling methods are taken for granted and packaged together with established best practices, e.g. BABOK [22]. Our position statement is that *value is co-created, during the lifecycle of a model, by at least a modeler and a modeling method engineer* – the latter being responsible for agilely capturing the adequate abstraction in order to satisfy the former's requirements and modeling use cases. Other stakeholders (e.g. domain experts) may also be involved as distinct roles.

Oversimplification 6. *Modeling languages are of two kinds: general-purpose and domain-specific.* Our position statement is that *both domain-specificity and modeling purpose are orthogonal dimensions*, as suggested in Figure 3: (i) the *purpose* axis ranging between "general purpose" and "narrow purpose"; (ii) the *specificity* axis ranging between "domain agnostic" and "system specific", with various intermediate degrees of specificity (perhaps including "technology-specific", "enterprise-specific"). The notion of "language agility" emerging from the previous points allows languages to shift within this Purpose-Domain space. In a recent paper [23] we introduced a dedicated modeling method for managing modeling method requirements, with a language that assimilates the notion of "purpose" as a first class modeling construct and gives a clear sense of the shifting nature of "domain specificity" as concepts and their relevant specializations are linked to purpose, and filtered or selected based on it.

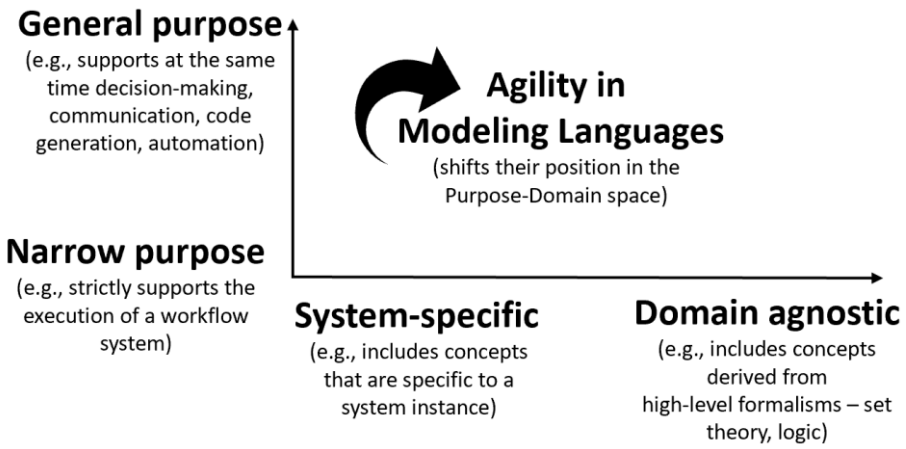

Figure 3. Expanding the general-purpose/domain-specific dichotomy to a Purpose-Domain space

4 Requirements and Treatments

The oversimplifications inventoried in Section 3 must be addressed by explicit knowledge assets and resources that are capable of substantiating the position statements and the requirements derivable from them. Figure 4 suggests how these position statements justify each other with a snowballing effect – a dependency graph that can be the basis for designing courseware workflows that are able to generate the insight proposed by the position statements.

One key aspect that can be observed in Figure 4 is the *trigger role of model queries* for this knowledge snowballing effect. As mentioned before, we include under "model queries" any means of retrieving model contents – from a serialized version of models (e.g. XML-based, JSON-based, RDF-based [17]), with a query engine built in the modeling environment itself (e.g. AQL in BEE-UP [21] reusable for any ADOxx modeling platform [24] based tools), through some dedicated APIs. Regardless of the technological choice, minimal toy examples can be designed to provide a convincing first contact and to sustain curiosity along the subsequent course flow.

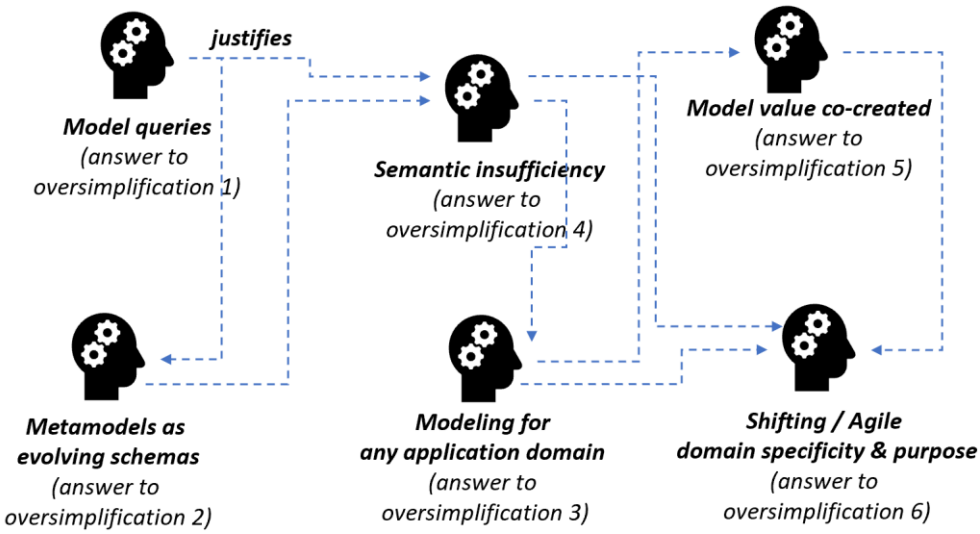

Figure 4. A workflow template for Conceptual Modeling courseware

In order to operationalise the position statements, we designed teaching strategies at master level that aim to satisfy the following requirements that have been derived from the oversimplifications discussed in Section 3:

- *Focus on machine readability*: focus on model queries as a key distinction between drawings and models, and as an immediately useful manifestation of a metamodel (perceived as a

model schema). Students familiar with SQL data queries will easily follow an analogy between (a) the need to design (then possibly evolve) a database schema and (b) the need to design (then possibly evolve) a modeling language.

- *Minimalism*: educators are experienced with introducing data queries to beginners in no more than 1–2 meetings – e.g. a simple front-end displaying data retrieved from a database, followed by small schema changes to enable richer query results. Comparable simple tutorials should introduce model queries – first on standards (e.g. "the list of BPMN user tasks in a given diagram"); later on "embryo" modeling tools developed by/with students with the help of fast prototyping platforms (we use the metamodeling platform ADOxx [24]).
- *Intuitive constructivism:* stimulate a Design Thinking approach to enable students to prototype their own agile (iterative) modeling tools through hands-on experience with fast prototyping (metamodeling) platforms, for their preferred application domain – preferably detached from Software Engineering to illustrate generality of model value, but targeting domains for which domain knowledge is not an obstacle and can be outlined in a quick discussion (e.g. through domain storytelling).
- *Open-ended agility:* take the minimally working examples of "embryo" modeling tools implemented by students and evolve them towards increased complexity and refined domain-specificity in an additive manner, with small increments that can be immediately reflected in richer model queries.
- *Link modeling rationale to domain and purpose*: highlight domain-specificity in any conceptualization, as well as the possibility of evolving that specificity by means as simple as possible (e.g. concept specialization with custom notation). Link domain expertise to competency questions that models should be able to answer through model queries. Provide examples of varying degrees of specificity – e.g. see in [25] the case of a "technology-specific" modeling language that is tailored for generating code fragments for a minor PHP programming library.
- *Refine "purpose" into explicit "modeling method requirements"*: invoke model-driven mechanisms and/or decision support as sources for modeling tool requirements that must be satisfied by modeling methods [23].
- *Generalization*: position Conceptual Modeling as means of knowledge capture/externalization within a Knowledge Management context, rather than as a Software Engineering phase. Software Engineering can then be introduced as one of many application areas (along with Enterprise Architecture Management, Service Engineering, etc.) that can benefit from knowledge captured in model form for a variety of purposes (some examples of purposes: for Software Engineering: communication, code generation, requirements analysis).

Examples of teaching cases and teaching artifacts have been presented in some of our recent works for such application domains as Smart Cities [26], Cooking Recipes [27], and Internet of Things [28].

We exemplify in Figure 5 two modeling languages developed with students, guided by the requirements enumerated in this section. Both examples are domain-specific – however, in order to satisfy the *Minimalism* and *Intuitive constructivism* requirements, the development of such artifacts makes use of only minimal domain knowledge outlined through domain storytelling from which students can derive competency questions, and from those – the concepts and relationships they need to operationalise in a language/tool. Interactive discussions can further stimulate the enrichment of students' domain knowledge through approaches such as Design Thinking, laddering, etc. But in order to produce their very first domain-specific modeling tool, we recommend targeting simple competency questions that models should be able to answer – this will intuitively translate to model queries giving a clear understanding of how domain knowledge is channelled into machine-readable structures (through the process of modeling method engineering).

On the left side of Figure 5 a maintenance process modeling language is enriched with conceptualizations of devices and sensors that can be mapped to process steps as well as to live sensor APIs, towards the purpose of bridging the design-time modeling environment with the run-time sensors setup; on the right side, a cooking recipe is mapped with ingredients and cooking resources in order to support "bill of material"-style reports. In both cases, purpose and domain are the drivers of language agility, as these languages evolve from more rudimentary versions (starting with "embryo" languages comprising 2–3 concepts) through iterative hands-on experience of students.

One commonality that is evident is that both examples are process-centric languages, which may lead to an oversimplification not enumerated in Section 3 (an undesired effect of the choice of examples) – that all modeling languages should have some type of control flow/process description at their centre. Educators should prevent this fallacy by also bringing into discussion model types that do not involve process thinking (e.g. goal modeling, product modeling, network-based or architectural modeling, etc.).

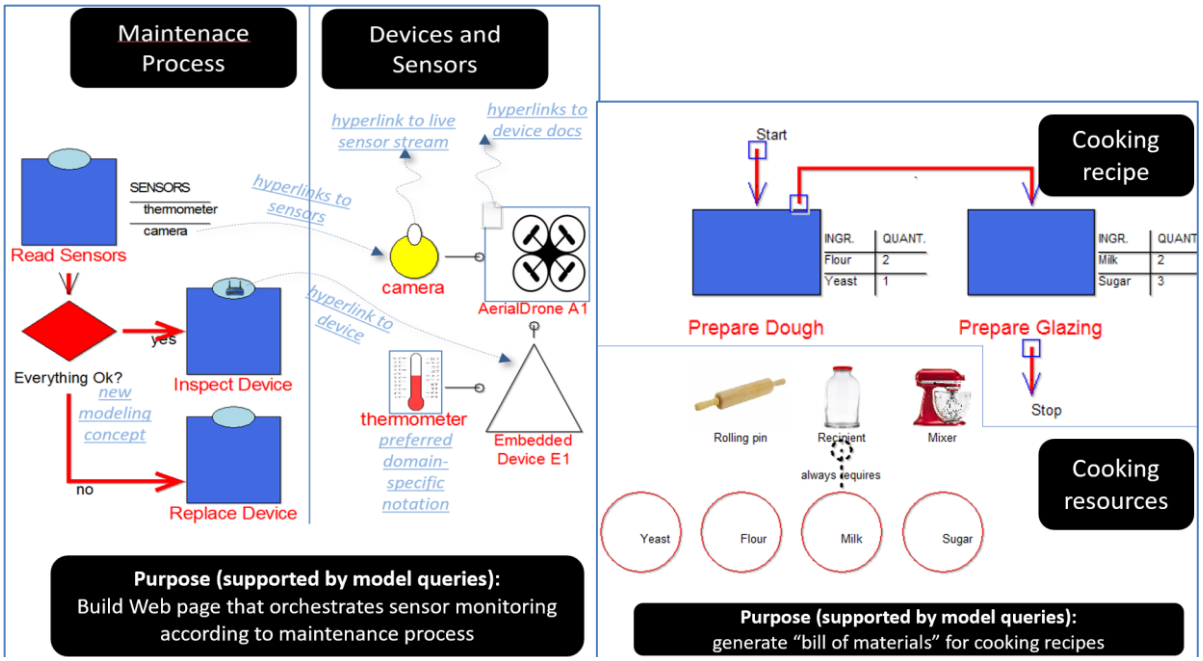

Figure 5. Model samples created with domain-specific purposeful modeling languages and tools implemented with Conceptual Modeling students (adapted from [27] and [28])

The success of the proposed treatments manifests in the availability of publishable research results authored by master students and presented in prestigious conferences – see BIR 2019 [17], ENASE 2018 [29], ICEIS 2018 [30] CAISE 2018 workshops [25] and PoEM 2018 workshops [31]. This has been a premiere in the first author's institution and further inspires industry collaborations on model-driven engineering for which students have, for the first time, a holistic understanding of the nature of models and how they can drive the engineering of other types of artifacts even when models do not play the role of graphical documentation.

5 The Value Proposition of OMiLAB for Conceptual Modeling

The proposed requirements and treatments need certain enablers – not only as underlying theories, but also for building proofs-of-concepts as part of tutorials and student exercises. Such enablers are available in the Open Models Laboratory (OMiLAB) [11] – a digital ecosystem built around the holistic model value proposition advocated in this article. Educators and students adopting the hereby presented position statements can benefit from OMiLAB resources in several ways:

- by tweaking open source modeling tools to shift their domain-specificity and purposefulness (e.g. BPMN for DevOps or for cooking recipes, ER for Knowledge Graphs);
- by implementing novel modeling methods as proofs-of-concepts created for a selected domain/purpose, including certain types of model-enabled evaluation (via reasoning, model analysis, etc.); this can be achieved with the help of the Agile Modeling Method Engineering (AMME) framework [32], which established the conceptualization process underlying the position statements hereby presented;
- by creating and evaluating model-driven artifacts with the help of interoperability features that can be adopted for any modeling language (e.g. RDF export, XML export, Model-as-a-service);
- by snowballing literature reviews starting from the rich corpus of publications reported by various projects hosted by OMiLAB, see [33];
- or, by employing modeling tools that are already available for open use, for a variety of languages (e.g. BEE-UP [21] supports in the same tool BPMN, EPC, UML, ER, Petri Nets, model queries as well as RDF export for any of these model types).

One key resource for the conceptualization and operationalisation of this value proposition is ADOxx [24] – a metamodeling platform for the fast prototyping of modeling tools, i.e. for tailoring their "knowledge schema" for a selected purpose or desired specificity, including a plug-in for converting any types of models to RDF graphs [34]. Another key resource is the Digital Product lab instance demonstrating the use of models as an intermediate knowledge layer between Design Thinking scenes and cyber-physical systems [12]. International OMiLAB nodes make such resources and infrastructures available to regional communities for both research and education purposes – see the developments of OMiLAB Korea [35].

6 Conclusions

The perception on Conceptual Modeling methods has shifted in time – from seeing them as ways of expressing mental constructs in graphical form, to employing them for complexity management, or for building formal specifications in support of model-driven engineering. The literature discusses extensively the nature and categories of Conceptual Modeling – e.g. differences between general-purpose and domain-specific modeling. This heterogeneity reflects the multitude of angles from which students can approach Conceptual Modeling, but it also raises confusion among junior researchers who debut with oversimplified preconceptions – summarized in this article as components of the "design problem of Conceptual Modeling education". By applying the Design Science framework we make explicit such issues that are rather tacit or entirely neglected when teaching modeling disciplines, or modeling techniques attached to other disciplines. We also derive requirements for treatments that can address these problems, including some examples of solutions in this respect.

The traditional positioning of Conceptual Modeling, which established the "purpose of understanding and communication" [16] inspired a stream of research on the relation between modeling and the human factor – e.g. studies on model comprehension [36] or on participatory modeling [37]. The problem formulated in this article emerges from the complementary aspect of machine-readability; as Conceptual Modeling can nowadays benefit from the advances of knowledge engineering to establish a bridge between humans and machines – thus "understanding and communication" includes now machine interpretability, semantic interoperability and human-to-machine communication through models. Considering this evolution, Conceptual Modeling education must reformulate its value proposition positioning it as a design problem, being able to establish an explicit strategy of addressing the challenges of Conceptual Modeling.

The article mapped each oversimplification regarding Conceptual Modeling to position statements that encourage a comprehensive perception of modeling languages as knowledge

schemas that reduce domain-specific complexity and operationalise semantics. Teaching designs in support of these arguments have been published recently [38] and dedicated workshops on Conceptual Modeling education are gaining attraction in the Information Systems community [39]. We take this opportunity to further call for teaching experiences and artifacts that can contribute to a holistic value of models and to the further refinement of the *Purpose-Domain space* where modeling languages can be positioned.

References

[1] R. A. Buchmann, A. M. Ghiran, V. Döller, and D. Karagiannis, "Conceptual Modelling in Education: a Position Paper," *Joint Proceedings of BIR 2019 Workshops and Doctoral Consortium*, CEUR-WS, vol. 2443, pp. 62–67, 2019. Available: http://ceur-ws.org/Vol-2443/paper06.pdf

[2] R. J. Wieringa, *Design Science Methodology for Information Systems and Software Engineering*, Springer, 2014. Available: https://doi.org/10.1007/978-3-662-43839-8

[3] D. Bogdanova and M. Snoeck, "Domain Modelling in Bloom: Deciphering How We Teach It," Poels, G., Gailly, F., Serral Asensio, E., Snoeck, M. (eds) *The Practice of Enterprise Modeling (PoEM 2017). Lecture Notes in Business Information Processing*, Springer, vol. 305, pp. 3–17, 2017. Available: https://doi.org/10.1007/978-3-319-70241-4_1

[4] M. Kayama, S. Ogata, K. Masymoto, M. Hashimoto, and M. Otani, "A practical Conceptual Modeling teaching method based on quantitative error analyses for novices learning to create error-free simple class diagrams," *Proceedings of 3rd Advanced Applied Informatics (IIAI 2014)*, IEEE, pp. 616–622, 2014. Available: https://doi.org/10.1109/IIAI-AAI.2014.131

[5] T. M. Glässner, F. Heumann, L. Keßler, F. Härer, A. Steffan, and H. G. Fill, "Experiences from the Implementation of a Structured-Entity-Relationship Modeling Method in a Student Project," *Proceedings of the 1st International Workshop on Practicing Open Enterprise Modeling within OMiLAB (PoEM 2017)*, vol. 1999, 2017. Available: http://ceur-ws.org/Vol-1999/paper4.pdf

[6] G. Muller, "Challenges in Teaching Conceptual Modeling for Systems Architecting," Jeusfeld M., Karlapalem K. (eds.) *Advances in Conceptual Modeling. ER 2015. Lecture Notes in Computer Science*, Springer, vol. 9382, pp. 317–326, 2015. Available: https://doi.org/10.1007/978-3-319-25747-1_31

[7] K. Rosenthal, B. Ternes, and S. Strecker, "Learning Conceptual Modeling: structuring overview, research themes and paths for future research," *Proceedings of ECIS 2019*, paper 137, Association for Informtion Systems, 2019.

[8] Association for Computing Machinery (2018). "Curricula Recommendations". https://www.acm.org/education/curricula-recommendations/ (accessed 14/10/2019).

[9] I. Bider, M. Henkel, S. Kowalski, and E. Perjons, "Teaching Enterprise Modeling Based on Multi-media Simulation: A Pragmatic Approach," *Proceedings of the 6th International Conference on E-Technologies*, Ed. by M. Benyoucef, M. Weiss, and H. Mili. Springer, pp. 239–254, 2015. Available: https://doi.org/10.1007/978-3-319-17957-5_16

[10] OMiLAB, NEMO Summer School – the official page. Available: http://nemo.omilab.org (accessed: 28/10/ 2019).

[11] OMiLAB, Open Models Laboratory Portal – official website. Available: http://omilab.org (accessed: 28/10/ 2019).

[12] D. Bork, R. Buchmann, D. Karagiannis, M. Lee, and E. T. Miron, "An Open Platform for Modeling Method Conceptualization: The OMiLAB Digital Ecosystem," *Communications of the Association for Information Systems*, vol. 44, pp. 673–697, 2019. Available: https://doi.org/10.17705/1CAIS.04432

[13] U. Kastens, H. K. Büning, *Modellierung: Grundlagen und formale Methoden*, Hanser Verlag, 2008 (in German).

[14] U. Seidl, M. Scholz, C. Huemer, and G. Kappel, *UML@Classroom*, Springer, 2015, Available: https://doi.org/10.1007/978-3-319-12742-2

[15] D. Karagiannis and H. Kühn, "Metamodelling Platforms," *Proceedings of EC-Web 2002 – DEXA 2002, LNCS*, Springer, vol. 2455, pp. 182–182, 2002. Available: https://doi.org/10.1007/3-540-45705-4_19

[16] J. Mylopoulos, "Conceptual modeling and Telos," *Conceptual Modeling, Databases, and Case – an integrated view of information systems development*, Wiley, pp 49–68, 1992.

[17] M. Cinpoeru, A. M. Ghiran, A. Harkai, R. A. Buchmann, and D. Karagiannis, "Model-Driven Context Configuration in Business Process Management Systems: An Approach Based on Knowledge Graphs," Pańkowska M., Sandkuhl K. (eds) *Perspectives in Business Informatics Research. BIR 2019. Lecture Notes in Business Information Processing*, Springer, vol. 365. pp. 189–203, 2019. Available: https://doi.org/10.1007/978-3-030-31143-8_14

[18] UML Stereotypes, Available: https://www.uml-diagrams.org/stereotype.html (accessed: 28/10/ 2019).

[19] X. Boucher, K. Medini, and H. G. Fill, "Product-Service-System Modeling Method," *Domain-specific Conceptual Modeling*, Springer, pp. 455–482, 2016. Available: https://doi.org/10.1007/978-3-319-39417-6_21

[20] T. P. Sales, B. Roelens, G. Poels, G. Guizzardi, N. Guarino, and J. Mylopoulos, "A pattern language for value modeling in ArchiMate," *Proceedings of CAISE 2019*, Springer, pp. 230–245, 2019. Available: https://doi.org/10.1007/978-3-030-21290-2_15

[21] OMiLAB. Bee-Up page in OMiLAB. Available: http://austria.omilab.org/psm/content/bee-up/info, (accessed: 28/10/ 2019).

[22] IIBA, A Guide to the Business Analysis Body of Knowledge, Available: https://www.iiba.org/standards-and-resources/babok/ (accessed: 28/10/ 2019).

[23] D. Karagiannis, P. Burzynski, W. Utz, and R. A. Buchmann, "A Metamodeling Approach to Support the Engineering of Modeling Method Requirements," *Proceedings of the 27th Int. Requirements Engineering Conference (RE 2019)*, pp. 199–210, 2019. Available: https://doi.org/10.1109/RE.2019.00030

[24] BOC GmbH, AdoXX official website, Available: https://www.adoxx.org/live/home, (accessed: 28/10/ 2019).

[25] A. Harkai, M. Cinpoeru, and R.A. Buchmann, "The What Facet of the Zachman Framework: a Linked Data-driven Interpretation," Matulevičius, R., Dijkman, R. (eds.) *Proceedings of CAISE 2018 Workshops, Lecture Notes in Business Information Processing*, Springer, vol. 316, pp.197–208, 2018. Available: https://doi.org/10.1007/978-3-319-92898-2_17

[26] D. Bork, R. Buchmann, I. Hawryszkiewycz, D. Karagiannis, N. Tantouris, and M. Walch, "Using Conceptual Modeling to support innovation challenges in Smart Cities," *Proceedings of IEEE 14th International Conference on Smart City (SmartCity 2016)*, IEEE, pp. 1317–1324, 2016. Available: https://doi.org/10.1109/HPCC-SmartCity-DSS.2016.0187

[27] R. A. Buchmann and A. M. Ghiran, "Engineering the Cooking Recipe Modelling Method: a Teaching Experience Report," *Proceedings of the 1st International Workshop on Practicing Open Enterprise Modeling within OMiLAB (PrOse 2017) co-located with 10th IFIP WG 8.1 Working Conference on the Practice of Enterprise Modelling (PoEM 2027)*, CEUR-WS, vol. 1999, paper 5, 2017. Available: http://ceur-ws.org/Vol-1999/paper5.pdf

[28] A. M. Ghiran, C. C. Osman, and R. A. Buchmann, "A Metamodeling Approach to Teaching Conceptual Modeling at Large," *Proceedings of ISD 2019*, Association for Information Systems, 2019. Available: https://aisel.aisnet.org/isd2014/proceedings2019/ISDMethodologies/1/.

[29] R. A. Buchmann, M. Cinpoeru, A. Harkai, and D. Karagiannis, "Model-Aware Software Engineering – A Knowledge-based Approach to Model-Driven Software Engineering," Damiani, E., Spanoudakis, G., Maciaszek, L. (eds.) *Proceedings of the 13th International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE 2018)*, ScitePress, pp. 233–240, 2018. Available: https://doi.org/10.5220/0006694102330240

[30] A. Harkai, M. Cinpoeru, and R. A. Buchmann, "Repurposing Zachman Framework Principles for Enterprise Model-Driven Engineering," Hammoudi, S., Smialek, M., Camp, O., Filipe, J. (eds.) *Proceedings of the 20th International Conference on Enterprise Information Systems (ICEIS 2018)*, ScitePress, vol. 2, pp. 682–689, 2018. Available: https://doi.org/10.5220/0006710706820689

[31] A. Chiș-Rațiu, and R. Buchmann, "Design and Implementation of a Diagrammatic Tool for Creating RDF graphs," *2nd PrOse Workshop co-located with PoEM 2018*, CEUR-WS, vol. 2238, pp. 37–48, 2018. Available: http://ceur-ws.org/Vol-2238/paper4.pdf

[32] D. Karagiannis, "Conceptual modelling methods: The AMME Agile Engineering approach," *Proceedings of the 15th Int. Conf. Informatics in Economy (IE), LNBIP*, Springer, vol. 273, pp. 3–19, 2018. Available: https://doi.org/10.1007/978-3-319-73459-0_1

[33] D. Karagiannis, H. Mayr, and J. Mylopoulos, *Domain-Specific Conceptual Modeling*, Springer, 2016. Available: https://doi.org/10.1007/978-3-319-39417-6

[34] OMiLAB, ComVantage project space, Production Management Systems, EU-Project. Available: https://austria.omilab.org/psm/content/comvantage/downloadlist (accessed: 28/10/ 2019).

[35] Open Models Laboratory Korea, Available: http://omilab-korea.org (accessed: 28/10/ 2019).

[36] R. Petrusel, J. Mendling, and H. Reijers, "How visual cognition influences process model comprehension," Decision Support Systems, vol. 96, pp. 1–16, 2017. Available: https://doi.org/10.1016/j.dss.2017.01.005

[37] A. Gutschmidt, V. Sauer, M. Schonwalder, and T. Szilagyi, "Researching participatory modeling sessions: an experimental study on the influence of evaluation potential and the opportunity to draw oneself," *Proceedings of BIR 2019*, Springer, pp. 44–58, 2019. Available: https://doi.org/10.1007/978-3-030-31143-8_4

[38] S. Strecker, U. Baumol, D. Karagiannis, A. Koschmider, M. Snoeck, and R. Zarnekow, "Five inspiring course (re-) designs," *Business & Information Systems Engineering*, vol. 61, no. 2, pp. 241–252, 2019. Available: https://doi.org/10.1007/s12599-019-00584-5

[39] The workshop on Teaching and Learning Conceptual Modelling, Available: http://merode.econ.kuleuven.be/events/TLCM2019/ (accessed: 28/10/ 2019).