# SUPPORTING TEACHING COMPLEX LEARNING TASKS WITH INSTRUCTIONAL DESIGN AND AUTOMATED FEEDBACK: AN EXPERIENCE REPORT

*Monique Snoeck, monique.snoeck@kuleuven.be, KU Leuven, Belgium, Full Professor, Head of the Research Center on Information Systems Engineering, PhD.*

**ABSTRACT:** Conceptual domain modelling is a complex learning task. To achieve good performance in conceptual domain modelling, students need a lot of feedback and lots of exercises. Given a teacher's limited amount of available time, there is a need to reflect on how to address the complexity of the learning task more effectively, how to improve the exercises provided to the students, and it is required to find ways to provide students with feedback more efficiently. This paper presents an experience report with the use of an instructional design method and automated feedback to address the challenge of teaching conceptual domain modelling more effectively. In particular, to obtain a more clear view on learning goals, and in particular on how to sequence the learning goals, a learning goal taxonomy was developed based on Bloom's taxonomy. Then, to achieve a better organisation of lectures, exercises and feedback in a systematic way, and in line with learning goals, elements of the 4C/ID method were applied. Finally, as a way to improve the delivery of feedback, the provision of simple feedback forms was automated. All in all, the experiences are positive for the teacher as well as for the students.

**Keywords:** Conceptual Modelling, Instructional Design, Technology-enhanced learning, automated feedback.

## Introduction

The subjects that are taught in higher education are often "complex learning tasks". By this we refer to problem solving tasks where different solutions may be considered for the same problem, and where different paths can be followed to obtain a solution. Moreover, complex learning tasks are characterized by the fact that many competences need to be integrated to perform well.

A typical example of such complex learning task is conceptual domain modeling, a topic taught in many computer science and information systems engineering curricula. Conceptual domain models are used to map the fundamental concepts of a domain of interest, and as the fundament for a high-level design of an information system. Conceptual domain modeling is a complex learning task: delivering models of good quality requires the integration of a series of competences in the field of requirements engineering (e.g. requirements gathering, understanding and analysis), and problem-solving skills (creativity, design skills, critical thinking skills). Furthermore, there are neither unique correct solutions, nor unique paths to arrive at a good solution.

Teaching complex learning tasks effectively poses a number of challenges to the teachers. A first challenge is a thorough understanding of the different competences needed to achieve good performance, as well as the required level of integration of these competences. Furthermore, a good understanding of how partial goals can be achieved as intermediate steps to the final learning goal is needed for assessing the competences of a student more accurately and providing better guidance.

A second challenge relates to the organization of the teaching itself. Especially in the case of larger groups of students with diverse backgrounds, handling the different pace at which students acquire the learning goals can be quite a challenge. While some students progress quickly and need more advanced exercises, other students are struggling and need more basic exercises. Determining the best pace at which to deliver theory and exercises is thus far from easy. And even when allowing students to study at their own rhythm, keeping track of what material to offer when to whom is quite challenging. A third challenge is related to feedback provisioning. By the fact that each problem may have several solutions, possibly with different ranges of quality, and that different paths can be followed to reach these solutions, just providing feedback about solutions being right or wrong is insufficient. Students will need (and request) personalized feedback and of explanatory nature rather than merely corrective.

This paper presents an experience report and explains how these different challenges were addressed by making use of educational theories and automation in a conceptual modelling course. The course "Architecture and Modelling of Management Information Systems" is taught yearly to a group of around 180 master students with varying backgrounds in terms of prior exposure to computer science topics. Both the large variance in prior knowledge and the size of the group make the teaching of the complex topic of conceptual modelling quite challenging. Section 2 explains how the use of Bloom's taxonomy helps in organizing learning goals, understanding their mutual relationships and developing a more stepwise approach to teaching and evaluation. Section 3 explains how the application of the Four-Components Instructional Design (4C/ID) method helps in developing and organizing the learning material. Section 4 then explains how the needs for more feedback were addressed by means of automation. Finally, section 5 concludes the paper with lessons learned.

## Organizing Learning Goals according to Bloom's taxonomy

Bloom's taxonomy has been developed in 1960 and has been revised in 2002 (Krathwohl, 2002). It is organized along two dimensions: the cognitive dimension defining six cognitive processes: Remember, Understand, Apply, Analyze, Evaluate and Create, and the knowledge dimension defining four types of knowledge: Factual, Conceptual, Procedural and Metacognitive knowledge (see Figure 1).

| | Cognitive Process Dimension | | | | | |
|---|---|---|---|---|---|---|
| Knowledge Dimension | Remember | Understand | Apply | Analyse | Evaluate | Create |
| Factual | LO1 | | | | | |
| Conceptual | | LO2, LO6 | LO3, LO7 | LO5 | LO8 | LO10 |
| Procedural | | | LO4 | | LO9 | |
| Metacognitive | | | | | LO11 | |

*Figure 1. Revised Bloom's Taxonomy*

The knowledge dimension allows distinguishing different types of knowledge:

*Factual knowledge* is the basis of the studied disciplines, such as basic terminology and/or notation. The symbols of UML class diagrams or BPMN is an example of factual knowledge.
*Conceptual knowledge* implies understanding the connections and interrelationships between the basic elements learned on factual level. The concepts of "class", "association", "activity", "event" as semantics concepts represented by the symbols, and how these concepts are related, are examples of conceptual knowledge.
*Procedural knowledge* refers to the subject-specific methods, procedures and rules. This could refer to usage of tools, processes to follow to achieve certain goals, etc.
*Metacognitive knowledge* is strategic knowledge related to the learning process and the student's awareness of own knowledge. An example is a student's knowledge about one's progress in the course, best working hours of the day, inclination to procrastination and effective countermeasures that work for him/her, etc.

The cognitive processes are defined by action verbs that characterize the learning tasks associated with them:

*Remember*: to define, recall, identify. An example would be to be able to name a list of symbols, or to draw the symbol given their names (e.g. draw the symbol for a non-interrupting timer event in BPMN).
*Understand*: to discuss, explain, match. An example would be explaining the difference between an interrupting and non-interrupting event in BPMN.
*Apply*: to use, practice, execute. An example would be to be able to use a programming environment to load and compile a program.
*Analyze*: to examine, analyze, compare. An example would be to find the differences between two proposed solutions and formulate the impact of these differences.
*Evaluate*: to check, verify, critique. An example would be to verify whether a solution (a program, a model) satisfies given requirements.
*Create*: to design, build, improve. An example would be creating a model or a software program from given specifications.

The ultimate learning objective of a course on conceptual modelling is the ability to *Create* models. This learning objective can be classified in the cell *Conceptual knowledge/Create*. One can easily see that to reach this final goal, intermediate objectives need to be achieved that are classified in the other cells of the framework. Examples of intermediate learning objectives, with their classification in the corresponding cell in Figure 1 are as follows:

LO1: Students need to master the modelling notation.
LO2: Students need to understand the concepts behind the symbols.
LO3: Given a requirement, a student needs to recognize what concept is suited to capture this requirement.
LO4: A student needs to be able to use a modelling tool to draw a diagram.
LO5: Given a UML class diagrams and an ER-model, a student needs to analyze their similarities and differences.
LO6: A student understands the different types of model quality.

LO7: A student is able to recognize model quality problems and label them with the correct type.
LO8: Given two alternative solutions for a same requirement, a student needs to be able to decide which one is best.
LO9: A student is able to evaluate whether a procedure for capturing requirements is suited for a given context or not.
LO10: A student is able to create a solution from a given set of requirements.
LO11: A student can evaluate the effectiveness of his/her study method.

When students fail to achieve the final learning objective, i.e. the student creates a bad model, the diagnosis of the reason of failure can be performed according to Bloom's taxonomy: Does the student know the notation and understand the concepts behind the notation? When making modelling choices, is the student able to analyze and explain the difference between to options? Is the student able to evaluate the pros and cons of each option? Does the student know and understand criteria for modelling quality? etc.

The framework also allows to assess to what extent a set of exercises used for either formative or summative evaluation covers the entire scale of cognitive processes and knowledge types. In (Bogdanova & Snoeck, 2017) the authors examined a variety of assessment instruments sourced from books, exams, and MOOCs. This revealed that the assessment materials are considerably unbalanced, which may cause difficulties both in the teaching and the learning processes as the majority of courses seem to focus on create-level assessments only. This lack of variety in assessment material supposedly finds its origin in the lack of an explicit taxonomy of learning objectives for conceptual modelling. To support the creation of a more variated set of exercises (Bogdanova & Snoeck, 2019) developed the **C**onceptual **M**od**e**ling goals **L**earning **O**bjective **T**axonomy (CaMeLOT), illustrated with possible corresponding student tasks. CaMeLOT is based on the revised Bloom's taxonomy, but as the different modelling concepts themselves also have prerequisite relationships, in addition, a concept map of the domain provides further support for organizing the learning objectives.

In the past Bloom's taxonomy has been successfully used in many different fields, such as biology (Crowe, Dirks, & Wenderoth, 2008), anatomy (Thompson & O'Loughlin, 2015), histology (Zaidi et al., 2017), and software engineering (Dolog, Thomsen, & Thomsen, 2016; Starr, Manaris, & Stalvey, 2008). Nevertheless, the taxonomy presents some limitations as well. In particular, classification of learning objectives can be somewhat ambiguous, and, as also noted by (Starr et al., 2008), concept shifting may occur: in case of broad and not too well delineated content areas, certain concepts or terms may be switched to similar ones. For example, in modelling, the distinction between *applying* knowledge about concepts and *creating* a model may seem difficult to make. Nevertheless, in a general sense, CaMeLOT was perceived as a useful tool by course instructors to help them in being more creative when developing exercises, and making the process of developing course material more systematic.

### Applying 4C/ID for course design

Simple instructional models such as Bloom's taxonomy work well as a starting point for improving course design. However, because of the simplicity of the model it cannot cater for complex learning. Richer instructional design theories such as the Four Components

Instructional Design (4C/ID) method (Van Merrienboer, J. J. G. Kirschner, 2018), which specifically targets complex learning, are providing better support to perform more fundamental changes.

The four components of the 4C/ID method are the learning tasks, the part-task practice, the supportive information and the just-in-time information.

According to 4C/ID, the learning tasks (or problems) offered to a student should always be (close to) real-life cases. The cases can range from very simple to complex, but should preferably always form a good representation of the type of tasks that students will encounter in their future work. As students cannot solve cases right away from the beginning, support in making the tasks should be provided. The most elaborated form of support is when the task is performed by a skilled person, and the student can observe how to approach a task. Less extensive support can be provided by e.g. performing the first (or last) steps, and having the student complete the task. A child baking cookies will e.g. only be asked to stick out the cookies from the already rolled out dough, and put them on the baking plate, while the other steps are performed by an adult (making and rolling out the dough, baking the cookies).

This concept of varying complexity and varying degrees of support can easily be applied to modelling tasks as well. Cases can range from small models needing just two or three classes, to complex cases requiring the students to create models of more than 20 classes. Support can range from providing students with a full model solution with explanation, over partial solutions they need to complete, to having the students making a solution from scratch.

The size and levels of support can be used to arrange the exercises in lab sessions to as along growing complexity and diminishing support. Figure 2 shows and example of how to organize conceptual modelling exercises. The course's running case is a full case addressing both the data modelling and the behavioral modeling aspects. As the model solution is given and extensively motivated, this is a case with full support. The first lab sessions focus on data modeling, starting with more simple cases in lab 1, and proceeding to more complex cases in lab 2. Per session, the support decreases from the first to the second exercise. Once the students master data modelling, focus can be put on behavioral modeling. To avoid the students having difficulties with behavioral modeling due to errors in the data model, the data model can be given as a kind of support. By the end of the course, students can be given full cases, where they need to address both aspects of modelling themselves. Besides providing students with partial models, other types of support can be provided, e.g. in the form of guiding questions, or by indicating steps to follow.

Part-task practice are exercises that help students to perform basic tasks in an almost automatic way without reflection. This can for example be achieved by making use of quizzes offering students drilling exercises, e.g. to learn notations, vocabulary, tool menus, etc. A more elaborated form of part-task practice is developing exercises that specifically focus on frequently occurring errors. In (Bogdanova & Snoeck, 2018) we report on an experiment that demonstrates the effectiveness of such error-correcting exercises to avoid repeating these same mistakes in subsequent tasks.

The supportive information is the learning material that is permanently at the disposal of students. In can be offered as a textbook, a series of videos or on a website. Just-in-time information is the information that is given to students at the moment they need it. The help function of a tool is an example of such information, as well as a teacher's advice given when a student is observed making a mistake.

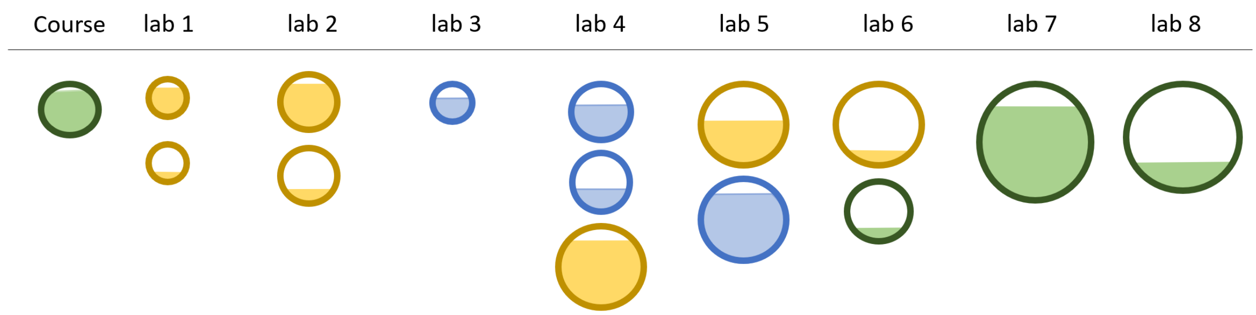


***Figure 2**. Organizing tasks in lab sessions along growing complexity and diminishing support - The size of the circle indicates the complexity of the task, and the filling the level of support. Yellow tasks represent data modelling, with no requirement for behavioral modelling. For blue tasks, the data model is given (= given support), and the students need to create the behavioral model. Green tasks are full cases where the students need to perform both data and behavioral modeling.*

The application of instructional design methods requires a deep understanding of cognitive schemas and knowledge required to perform a task. As for conceptual modelling, much of this knowledge is still tacit, using a rich method such as 4C/ID right from the first time of teaching the course would have been too overwhelming. Starting with Bloom's taxonomy was therefore needed: it allowed developing a better understanding of the different types of leaning objectives that can be considered, it helped creating a more rich pallet of exercises and developing a better understanding of student's cognitive patterns. This increased understanding allowed reaping the benefit of the 4C/ID method.

### Technology-supported feedback

Feedback is essential in fostering students' learning (Hattie & Timperley, 2007). Training student in a complex learning tasks such as conceptual modeling requires a lot of individual feedback: given a certain problem, different students will come up with different solutions for the problem, and will follow different paths to arrive at a solution. Giving corrective feedback (i.e. telling a student whether his/her solution is right or wrong) will not suffice as students need to understand why solutions are better or worse, and how to evaluate differences between solutions. It is therefore important to make use of many different possible forms of feedback, including more advanced forms (S. Serral, Ruiz, Elen, & Snoeck, 2019). Given that the size of the class grew from a manageable number of 50 students to approximatively 180 students, providing all students with a sufficient amount of personalized feedback was a growing concern that was addressed by making maximal use of technology-supported feedback.

*Corrective feedback* (right or wrong) is the simplest form of feedback and can be used for simple exercises such as e.g. gauging for the understanding of the case description given, or for the understanding of basic concepts (see Figure 3). However, providing just corrective feedback is often not enough. *Elaborative feedback* helps students understanding why certain solutions are more right or wrong than others. Elaborative feedback can be provided as explanations in multiple choice questions, or e.g. as annotated model solutions or student solutions, explaining their good and bad elements (see e.g. Figure 4).

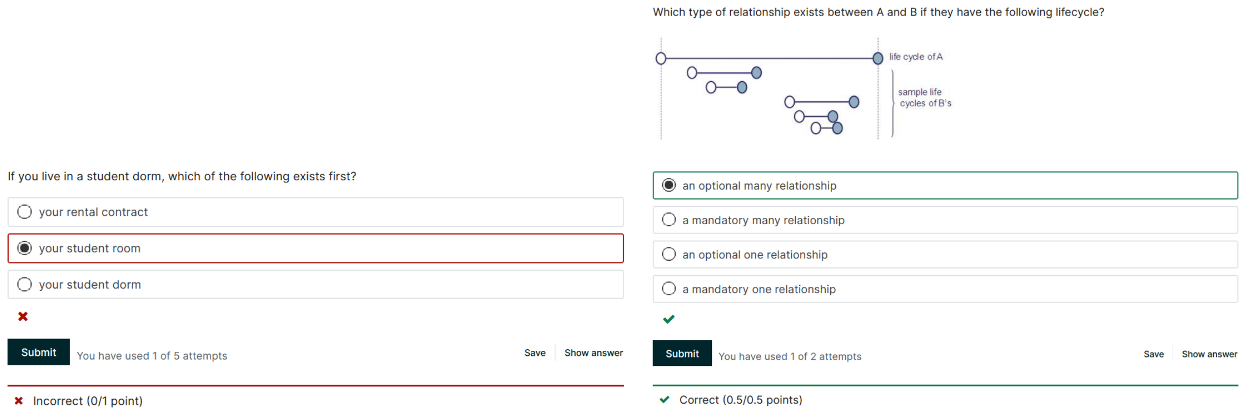


*Figure 3. Examples of questions with corrective feedback*

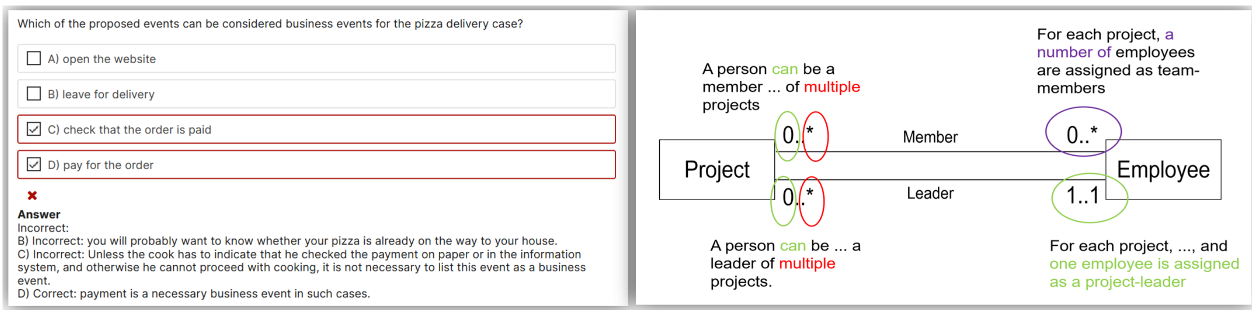


*Figure 4. Elaborative feedback as explanations in multiple choice questions or as annotated model solution*

Developing a student's ability to elaborate themselves on the correctness and suitability of a solution can be achieved even better by providing students with cognitive feedback: prompts, cues, questions ... that help the learners to reflect on the quality of their modelling process and resulting models, so that they construct more effective cognitive schemas to improve future performance (Estefania Serral, De Weerdt, Sedrakyan, & Snoeck, 2016) (G. Sedrakyan & Snoeck, 2017). A very simple form of cognitive feedback, such as translating a student's model to text (This is what your model says: "..."; is this what you meant to express?) proves to be already quite effective to foster a student's self-reflection on his/her modelling performance.

Individual feedback fosters the evolution towards a more student-centered, active learning approach. This can be achieved by cutting down on lecture time in favor of lab sessions where students can exercise at their own pace. Also, "flipped classrooms", where students study the easy parts on their own, allow reserving contact hours to deal with students' individual questions rather than for lecturing.

The experience with the positive effects of cognitive feedback, and the increase of the class size to around 100 students triggered a search for automating feedback. The MERODE modelling tool[1] allows a student to draw a conceptual model as ac combination of a class diagram, finite state charts and an object-event table that captures the interaction aspects. The tool was enriched with different forms of automated 'on demand' cognitive feedback: a model-to-text feature and verifying the model for obvious missing elements (e.g. no way to create or end objects in a class) (Snoeck, Haesen, Buelens, De

---

[1] http://merlin-academic.com

Backer, & Monsieur, 2007). More advanced cognitive feedback has been developed as a prototyping tool: an in-depth understanding of a model requires the ability to mentally picture and test the application that will result from the model, something that is very hard to achieve for novice modelers. Students can export their model to a code generator that will generated a prototype Java application. This enables the students to simulate a model. To help students linking the application's behavior to its origin in the model, the code generation was enriched with cognitive feedback: when the application refuses an action, the error window visualizes the location of the constraint in the model, e.g. "You cannot perform the 'ship' action on this order, because the finite state charts of 'Order' says your order needs to be in the state 'confirmed'". Experimental research shows that such cognitive feedback enhances the students' performance significantly (G. Sedrakyan, Poelmans, & Snoeck, 2017; Gayane Sedrakyan, Snoeck, & Poelmans, 2014). The same approach applied to the model-driven engineering of user interfaces (Jenny Ruiz et al., 2017), shows a similar positive effect on students' performance (J. Ruiz, Serral Asensio, & Snoeck, 2020). Mining the logs of student activity furthermore shows a difference in the process of modelling between better and worse students (Gayane Sedrakyan, De Weerdt, & Snoeck, 2016). This opens up the perspective for process-oriented feedback as a complement to the current task-oriented feedback (Hattie & Timperley, 2007; Estefanía Serral & Snoeck, 2016). The different types of (task-oriented) feedback are very much appreciated by the students (Jenny Ruiz et al., 2017; Gayane Sedrakyan et al., 2014; Snoeck et al., 2007).

## Bringing it all together

Learning management systems are helpful in organizing the learning material in a systematic way. Rather than organizing a course's page along the type of material (slides, exercises, assignments, etc.), organizing the material along learning objectives allows creating modules that brings together the tasks, the supporting material, and part-task practice exercises that address a common set of learning objectives.

This also facilitates moving to a blended learning approach where part of the learning happens online. Most of the learning management systems offer facilities for incorporating quizzes and thus facilitate the inclusion of automated (corrective) feedback. Videos, quizzes and tasks can thus be brought together to foster active learning.

## Lessons learned

Moving to a blended approach changes the nature of teacher-student interaction: a course becomes more student-oriented and less-student oriented. Overall, the offering of online material is very much appreciated by students. Automated feedback, code generation and online course material all score high on perceived utility by students. Short videos and recorded lectures are deemed useful for re-watching material students missed or didn't understand fully, slides are appreciated because of their more visual character, while the textbook is appreciated for its completeness, and found easier to grasp the global picture compared to online material. The different types of material clearly serve different goals and different learner preferences.

On the downside it should be noted that the development of the tools, automated feedback and online lectures is time consuming. Which material to convert to online should therefore be carefully considered: this should preferably be done for stable course parts, e.g.

parts that have been used and perfected during face-to-face teaching. In addition, care should be taken to modularize and implement the material in such a way that modifications can be performed without too much rework. Despite the required investment effort, automation is worth the effort as it pays off in subsequent years and it creates more time for coaching student on more interesting and challenging questions.

Another potential downside is that while the student-centered approach is appreciated by all, it seems to only work well for students with high self-regulation capabilities: a self-paced course leaves more room for procrastination. This could be partly addressed by means of permanent evaluation and process-oriented feedback. Yet it remains an open question to what extent this is the responsibility of teachers at higher education level, especially when teachers face large groups. To a certain extent, one can argue that developing one's self-regulation capabilities is part of the generic learning objectives of higher education.

## Conclusion

This paper presents the experiences with using Bloom's taxonomy, 4C/ID, and technology-supported feedback for improving the teaching of the complex learning topic of Conceptual Modelling. Overall, the experiences have been more than positive: even though the course is experienced as hard, it receives excellent evaluations. From a teacher's perspective, starting with Bloom's taxonomy and proceeding to the use of 4C/ID once sufficient insight in the structure of of learning objectives and students' cognitive schema's has been gained, is the preferred approach. Developing technology support for feedback provisioning can be quite time consuming, but will benefit to students, and free the teacher of basic forms of feedback provision. Finally, care has to be taken to monitor students' self-regulation capabilities required when following a student-centered approach.

## References


Bogdanova, D., & Snoeck, M. (2017). Domain modelling in bloom: Deciphering how we teach It. In *Lecture Notes in Business Information Processing* (Vol. 305). https://doi.org/10.1007/978-3-319-70241-4_1

Bogdanova, D., & Snoeck, M. (2018). Learning from Errors: Error-based Exercises in Domain Modelling Pedagogy. In R. A. Buchmann & M. Kirikova (Eds.), *The Practice of Enterprise Modelling 2018*: *Vol. LNBIP 335* (pp. 257–273). https://doi.org/10.1007/978-3-030-02302-7

Bogdanova, D., & Snoeck, M. (2019). CaMeLOT: An educational framework for conceptual data modelling. *Information and Software Technology*, *110*(June), 92–107. https://doi.org/10.1016/j.infsof.2019.02.006

Crowe, A., Dirks, C., & Wenderoth, M. P. (2008). Biology in Bloom: Implementing Bloom's Taxonomy to Enhance Student Learning in Biology. *CBE—Life Sciences Education*, *7*(4), 368–381. https://doi.org/10.1187/cbe.08-05-0024

Dolog, P., Thomsen, L. L., & Thomsen, B. (2016). Assessing Problem-Based Learning in a Software Engineering Curriculum Using Bloom's Taxonomy and the IEEE Software Engineering Body of Knowledge. *ACM Trans. Comput. Educ.*, *16*(3). https://doi.org/10.1145/2845091

Hattie, J., & Timperley, H. (2007). The power of feedback. *Review of Educational Research*, *77*(1), 16–17. https://doi.org/10.3102/003465430298487

Krathwohl, D. R. (2002). A Revision of Bloom's Taxonomy: An Overview. *Theory Into Practice*, *41*(4), 212–218. https://doi.org/10.1207/s15430421tip4104_2

Ruiz, J., Serral Asensio, E., & Snoeck, M. (2020). Learning UI Functional Design Principles through Simulation with Feedback. *IEEE Transactions on Learning Technologies*. https://doi.org/10.1109/TLT.2020.3028596

Ruiz, Jenny, Serral, E., Snoeck, M., Ruiz., J., Serral., E., Snoeck., M., … Snoeck, M. (2017). UI-GEAR: User Interface Generation prEview capable to Adapt in Real-time. *MODELSWARD 2017 - Proceedings of the 5th International Conference on Model-Driven Engineering and Software Development*, *2017-Janua*, 277–284. https://doi.org/10.5220/0006115402770284

Sedrakyan, G., Poelmans, S., & Snoeck, M. (2017). Assessing the influence of feedback-inclusive rapid prototyping on understanding the semantics of parallel UML statecharts by novice modellers. *Information and Software Technology*, *82*. https://doi.org/10.1016/j.infsof.2016.11.001

Sedrakyan, G., & Snoeck, M. (2017). Cognitive feedback and behavioral feedforward automation perspectives for modeling and validation in a learning context. In *Communications in Computer and Information Science* (Vol. 692). https://doi.org/10.1007/978-3-319-66302-9_4

Sedrakyan, Gayane, De Weerdt, J., & Snoeck, M. (2016). Process-mining enabled feedback: “tell me what i did wrong” vs. “tell me how to do it right.” *Computers in Human Behavior*, *57*, 352–376. https://doi.org/10.1016/j.chb.2015.12.040

Sedrakyan, Gayane, Snoeck, M., & Poelmans, S. (2014). Assessing the effectiveness of feedback enabled simulation in teaching conceptual modeling. *Computers & Education*, *78*, 367–382. https://doi.org/10.1016/j.compedu.2014.06.014

Serral, Estefania, De Weerdt, J., Sedrakyan, G., & Snoeck, M. (2016). Automating immediate and personalized feedback taking conceptual modelling education to a next level. *Proceedings - International Conference on Research Challenges in Information Science*, *2016-Augus*. https://doi.org/10.1109/RCIS.2016.7549293

Serral, Estefanía, & Snoeck, M. (2016). Conceptual Framework for Feedback Automation in SLEs. *3rd International KES Conference on Smart Education and E-Learning (KES-SEEL-16)*, (to be published).

Serral, S., Ruiz, J., Elen, J., & Snoeck, M. (2019). Conceptualizing the Domain of Automated Feedback for Learners. *XXII Ibero-American Conference on Software Engineering*.

Snoeck, M., Haesen, R., Buelens, H., De Backer, M., & Monsieur, G. (2007). Computer aided modelling exercises. *Informatics in Education*. Retrieved from http://www.scopus.com/inward/record.url?eid=2-s2.0-34249046363&partnerID=MN8TOARS

Starr, C. W., Manaris, B., & Stalvey, R. H. (2008). Bloom’s Taxonomy Revisited: Specifying Assessable Learning Objectives in Computer Science. *SIGCSE Bull.*, *40*(1), 261–265. https://doi.org/10.1145/1352322.1352227

Thompson, A. R., & O’Loughlin, V. D. (2015). The Blooming Anatomy Tool (BAT): A discipline-specific rubric for utilizing Bloom’s taxonomy in the design and evaluation of assessments in the anatomical sciences. *Anatomical Sciences Education*, *8*(6), 493–501. https://doi.org/10.1002/ase.1507

Van Merrienboer, J. J. G. Kirschner, P. A. (2018). *Ten steps to complex learning* (3rd ed.). New York: Routledge.

Zaidi, N. B., Hwang, C., Scott, S., Stallard, S., Purkiss, J., & Hortsch, M. (2017). Climbing Bloom’s taxonomy pyramid: Lessons from a graduate histology course. *Anatomical Sciences Education*, *10*(5), 456–464. https://doi.org/10.1002/ase.1685