Supporting Qualitative Model Construction in Open Modelling Tasks

Marco Kragten[1], **Tessa Hoogma**[1], **and Bert Bredeweg**[1,2]
[1]Faculty of Education, Amsterdam University of Applied Sciences, Amsterdam, The Netherlands
[2]Informatics Institute, Faculty of Science, University of Amsterdam, Amsterdam, The Netherlands
{m.kragten, t.e.hoogma, b.bredeweg}@hva.nl

Abstract

Constructing models of system behaviour is challenging for students, especially in open modelling tasks where no predefined solution is provided. In such tasks, students must coordinate knowledge of the modelling vocabulary, the domain, and the purpose of the model. We present a set of support features implemented in DynaLearn to scaffold qualitative model construction while preserving task openness. These include automated vocabulary-based checks, reflective prompts for domain- and purpose-related decisions, textualization of the model into natural language, and scenario advice for defining initial conditions. We describe the design and implementation of these features and illustrate their use in an example task. Together, they provide complementary support for constructing and interpreting qualitative models of dynamic systems.

1 Introduction

Understanding and modelling system behaviour is a central goal in science education [1,2]. Through modelling, students not only develop domain-specific knowledge, but also acquire insight into how scientific knowledge is constructed, represented, and evaluated [2,3]. In particular, modelling supports students in reasoning about dynamic systems, where behaviour emerges from interactions between multiple components over time.

A common approach to modelling in education involves the use of simulation environments with formal modelling languages [3]. While such environments offer powerful opportunities for learning, they also pose substantial challenges for students. Constructing a model requires students to translate a system description into a formal representation, often leading to trial-and-error strategies and superficial engagement with the modelling process [4].

In this paper, we focus on **open modelling tasks**, in which students construct a model based on a system description without being provided with a predefined reference model [5,6]. In contrast to closed tasks with a fixed correct solution, open modelling tasks require students to make decisions about how to represent the system, which elements to include, and how these elements are related. This makes them particularly valuable for fostering deeper understanding of modelling and of the epistemological nature of models, but also significantly increases complexity [4].

To support modelling in such tasks, we adopt the qualitative modelling approach as implemented in DynaLearn [7]. Qualitative models describe system behaviour using a symbolic, non-numerical vocabulary that aligns closely with everyday reasoning [8]. This allows students to focus on the structure and behaviour of systems without the additional burden of mathematical formalisation. At the same time, constructing qualitative models requires students to correctly use a formal modelling vocabulary and to integrate multiple modelling ingredients into a coherent representation.

The **quality** of a student-generated model can be characterised in terms of **correctness, completeness, and parsimony** [9,10]. Correctness concerns whether the model accurately represents the system, completeness whether all relevant elements are included, and parsimony whether the model avoids unnecessary complexity. Furthermore, constructing high-quality models requires different types of knowledge: knowledge of the modelling vocabulary, domain knowledge, and knowledge of the purpose of the model. In open modelling tasks, students must coordinate these types of knowledge simultaneously, which often leads to incomplete or inconsistent models [10].

This highlights the need for **targeted support** that helps students construct higher-quality models without constraining the openness of the task. Existing approaches often rely on comparing the student-generated model with a predefined reference model, where feedback is provided by identifying differences between the two. While this can support correctness, it limits students' autonomy and reduces opportunities for meaningful modelling decisions [5]. In this paper, we present a set of support features implemented in DynaLearn that guide students during model construction while preserving the open nature of the task.

The contribution of this paper is twofold. First, we describe a set of complementary support features that address different types of knowledge involved in modelling: automated checks of the modelling vocabulary, reflective prompts for domain and purpose-related decisions, textualization of the model to support semantic interpretation, and scenario advice for defining initial settings. Second, we show how these features

are integrated into a modelling environment to support students throughout the modelling process in open modelling tasks.

2 Open Modelling Task

In this study, we focus on open modelling tasks, in which students construct a model based on a **system description** without being provided with a predefined reference model. In such tasks, there is **no single correct answer**; instead, students must decide how to represent the system using the modelling vocabulary, including which model ingredients to include and how they are related. Fig. 1 shows an example of such a task.

Photosynthesis and dissimilation task

Description of the system
A plant is located under an airtight bell jar (see the figure below). You have a lamp whose light intensity you can adjust. When the lamp is off, only dissimilation (cellular respiration) takes place in the plant, which causes an increase in the amount of carbon dioxide in the bell jar. As soon as the light intensity of the lamp increases, photosynthesis begins. At a certain light intensity, photosynthesis becomes greater than dissimilation, causing the amount of carbon dioxide in the bell jar to start decreasing.

Your task
Create a computer model that can simulate the behaviour of the described system.

Fig. 1. Open modelling task on photosynthesis and cellular respiration (translated from the original description in Dutch).

Students are asked to construct a model of a plant under a jar with a controllable light source, in which the balance between photosynthesis and cellular respiration determines the carbon dioxide concentration.

Although the task description is concise, constructing a model from it is **non-trivial**. Students must identify relevant entities (e.g., plant, lamp, jar), determine appropriate quantities (e.g., light intensity, rate of photosynthesis, carbon dioxide concentration), and specify causal relationships, thereby distinguishing between the structure of the system and its dynamics.

In addition, the task requires interpreting more subtle aspects of the system description. For example, the statement that "at a certain light intensity, photosynthesis becomes larger than respiration" requires students to represent a comparison between processes. Furthermore, to simulate the model, students must assign appropriate initial values.

3 Qualitative Vocabulary for Modelling System Behaviour

DynaLearn provides a **qualitative modelling vocabulary** for representing system behaviour as described in the open modelling task (Fig. 1) without relying on numerical equations [7]. Students can construct qualitative models across **five levels of increasing complexity** [13]. The support described in this paper is implemented at levels 2, 3, and 4. In this study, we focus on modelling at **level 4**, which includes all modelling ingredients from lower levels and extends these with additional constructs for representing more complex system behaviour.

Fig. 2 shows an example of a qualitative model that can be constructed based on the task in Fig. 1. As the task is open, **multiple alternative models** are possible depending on how students interpret the system and which aspects they include. The model in Fig. 2 is therefore not intended as a reference or correct solution, but serves to illustrate how the qualitative modelling vocabulary can be used to represent the described system.

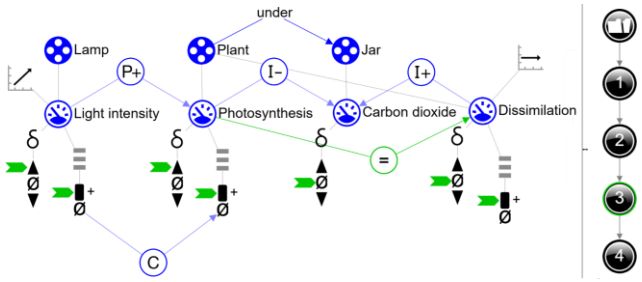

Fig. 2. One possible qualitative model for the task in Fig. 1 and its simulation results. The left panel shows the model with the simulation outcome for the selected state (state 3; indicated by green arrows). The right panel shows the state graph, starting from the initial scenario and followed by four consecutive states.

At the core of the vocabulary are several types of modelling ingredients. **Entities** represent physical objects or abstract concepts in the system. In the example model (Fig. 2), entities include the *Lamp*, *Plant*, and *Jar*. Entities can be structurally related through **configurations**, which describe how entities are connected or arranged. For instance, the

model can represent that the plant is located *under* the jar through a configuration.

Quantities represent measurable and changeable properties of entities. Each quantity has a **direction of change** (δ), indicating whether it is decreasing, constant, or increasing. In the example model, relevant quantities include *Light intensity* (of the lamp), *Photosynthesis* and *Dissimilation* (of the plant), and *Carbon dioxide* concentration (of the jar). Quantities are further defined by a **quantity space**, consisting of a set of possible values structured as alternating points and intervals. For example, light intensity can be represented with a quantity space containing a point value *0* (lamp off) and a positive interval *(+)*. Similarly, photosynthesis can take on values *0* or +.

To express constraints between quantities, the vocabulary includes **correspondences (C)**, which specify co-occurring values or directions of change. In the example, a correspondence is defined between *Light intensity = 0* and *Photosynthesis = 0*, capturing that photosynthesis does not occur in the absence of light. Dissimilation is also represented with a quantity space (0 and +), as it is an ongoing process that affects the system and whose magnitude must be specified.

Relationships between quantities are expressed through two types of **causal relationships**: *proportionalities* and *influences*. A **proportional relationship (P)** describes how changes in one quantity lead to changes in another, either in the same direction (P+) or in the opposite direction (P−). In the example model, light intensity has a *positive proportional relationship* with photosynthesis. An **influence (I)** represents the effect of a process on another quantity. Influences are used when a quantity functions as a process that adds to or removes from the system over time. These can be positive (I+) or negative (I−). In the model, photosynthesis acts as *a negative influence* on carbon dioxide concentration, while dissimilation acts as *a positive influence*.

Calculi allow the execution of qualitative operations such as addition and subtraction. For example, a calculus can represent that photosynthesis *minus* dissimilation *equals* net production. To keep the example model (Fig. 2) simple and focused, calculi are not included. However, they are supported in DynaLearn, and their representation is included in the support features described in this paper.

Simulation in DynaLearn is based on a **scenario**, which specifies the initial settings of the model. This includes assigning initial values and directions of change to quantities. In the example, photosynthesis starts at value *0* (scenario and state 1, not shown in Fig. 2) with an increasing direction of change, implemented through an **exogenous influence** (an external factor that continuously affects a quantity). *Dissimilation* is modelled as a constant positive process, also using an exogenous influence. The initial relationship between these processes can be further specified using **inequalities** (e.g., *Photosynthesis* < *Dissimilation*), indicating that at the start of the simulation, respiration dominates.

Running the simulation generates a **state graph**, which represents the possible states of the system and transitions between them. For the example model, the simulation produces a sequence of states in which carbon dioxide first increases (due to dominant dissimilation) and later decreases once photosynthesis exceeds dissimilation. Fig. 2 shows one of those states, and Table 1 summarises the full simulation results. Each state describes the value and direction of change of all quantities, typically represented as tuples <υ, δ>.

Table 1. Simulation results for model shown in Fig. 2 (P refers to *Photosynthesis* and D refers to *Dissimilation*).

Quantity	S_1	S_2	S_3	S_4
Light intensity	<0, +>	<+, +>	<+, +>	<+, +>
Photosynthesis	<0, +>	<+, +>	<+, +>	<+, +>
Carbon dioxide	< u, −>	< u, −>	<u, 0>	< u, +>
Dissimilation	<+, 0>	<+, 0>	<+, 0>	<+, 0>
P ? D	P < D	P < D	P = D	P > D

u = undefined.

While this vocabulary enables the representation of complex system behaviour, it also introduces substantial demands on students. Constructing a model requires correctly identifying modelling ingredients and ensuring that they are meaningfully connected. Students must coordinate multiple constraints simultaneously: quantities must be linked to the correct entities, causal relationships must be correctly specified, relevant quantity spaces need to be defined, and initial settings must specified in order to support a valid simulation.

As a result, modelling in an open task is not only a matter of translating a system description into a formal representation, but also of ensuring internal consistency and functional coherence. This complexity underpins the need for targeted support during the modelling process.

4 Support features

To support students in constructing qualitative models in open modelling tasks, we implemented a set of complementary support features in DynaLearn. These features are designed to support the **different types of knowledge** required to construct higher-quality models, namely knowledge of the modelling vocabulary, domain knowledge, and knowledge of the purpose of the model. Rather than guiding students towards a predefined solution, the support aims to help students **construct higher-quality models** by providing feedback and prompts during the modelling process. All support features can be toggled on or off by the student via icons on the right side of the modelling canvas, allowing students to regulate the level of support during model construction.

In this section, we describe **four types of support**. We first present automated checks based on the qualitative modelling vocabulary, followed by reflective prompts for aspects that cannot be automatically assessed, a textual representation of the model to support semantic interpretation, and scenario advice to support setting initial values for simulation. These four support types are complementary in nature and address

different stages and aspects of the modelling process. Automated checks provide immediate feedback on formal correctness during model construction, reflective prompts encourage students to evaluate their modelling decisions, textualization supports interpretation of the model as a coherent description, and scenario advice supports the transition from model construction to simulation. Together, they provide a layered form of support that combines guidance with opportunities for student reasoning.

The design of these support features was informed through an iterative process involving domain experts, modelling experts, and pedagogical experts. Across multiple design sessions, successive versions were developed, tested on a range of models, and refined through discussion.

4.1 Automated Vocabulary-Based Support

The first support type consists of automated checks on the **correctness and completeness with respect to the modelling vocabulary**, which can be assessed without domain-specific knowledge or knowledge of the purpose of the model [9,10]. These checks are triggered whenever a student constructs or modifies a model ingredient. If an issue is detected, a **red light bulb** appears above the corresponding ingredient, accompanied by a short message indicating what is incorrect or incomplete and how it can be resolved. An example of this feedback is shown in Figure 3.

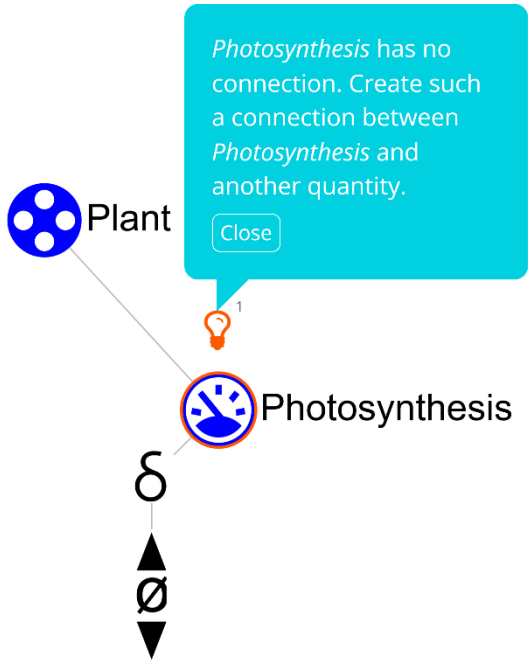

Fig. 3. Automated vocabulary-based feedback in DynaLearn. The red light bulb indicates an issue with the quantity *Photosynthesis* (of *Plant*), with a counter showing one active message. The system detects that the quantity is not connected to the model and provides a message prompting the student to establish a connection with another model component.

In this example, the student has created the quantity *Photosynthesis* associated with the entity *Plant*, but has not yet connected it to other quantities. The system detects that the quantity is not integrated into the causal structure of the model and provides a targeted message prompting the student to establish a connection. While such a connection is often realized through a causal relationship, other forms of integration are also possible, for example through a calculus.

When applicable, a notification is specific and contextualised using the names of the model ingredients, reducing abstraction and helping students relate the feedback directly to their model. A counter displayed on the light bulb indicates the number of active warnings. Students can inspect and close messages, but when multiple issues are present, feedback is presented sequentially: the next warning is only shown after the previous issue has been addressed. This ensures that students focus on resolving problems step by step, rather than being overwhelmed by multiple simultaneous messages.

Table 2 provides an overview of the automated checks implemented in the system.

Table 2. Overview of automated vocabulary-based checks.

Action by student	Feedback message
Create entity	"Give the entity a name."
Name entity	"*Plant* is not connected to the model. Add a quantity or connect it to another entity."
Create quantity	"Give the quantity a name."
Name quantity	"*Photosynthesis* has no connection. Create such a connection between *Photosynthesis* and another quantity."
Create configuration	"Give the configuration a name."
Create influence relation	"A process quantity requires a quantity space."
Create interval for process first	"The quantity space of a process quantity must have a zero point (0)."
Create zero point first	"The quantity space of a process quantity must also have a positive or negative value."
Combine I and P on same target quantity	"An I and P relationship may not influence a single quantity at the same time."
Create calculus without target quantity	"The calculus between *Photosynthesis* and *Dissimilation* has no connection. Create such a connection between calculus and a quantity." *

* Not in example model.

4.2 Reflective Support

The second type of support consists of **reflective prompts**, aimed at aspects of model quality. Reflective prompts cannot be automatically assessed from the representation. These checks require **domain knowledge** or **knowledge of the purpose of the model**, and are therefore not implemented as automated feedback. Instead, they are presented as questions that prompt students to reflect on their modelling decisions.

This support is represented by a **grey light bulb**, shown above the corresponding model ingredient. In contrast to the automated checks, this feature is **disabled by default**, as continuous prompting may otherwise become overwhelming during model construction. Students can activate it when needed to reflect on specific modelling choices.

The reflective prompts become accessible only after all automated (red light bulb) issues for the corresponding ingredient have been resolved, ensuring that basic vocabulary constraints are satisfied before engaging in higher-level reflection. A counter on the grey light bulb indicates the number of available prompts.

When the student selects the grey light bulb, a text box appears presenting a reflective question, along with response options (*Yes*, *No*, and *Close*). Selecting *Yes* advances to the next prompt for that ingredient. Once all prompts have been answered affirmatively, the grey light bulb disappears. If a student selects *No* or closes the prompt, the question will reappear upon reopening, **encouraging reconsideration at a later stage**. As with the automated vocabulary-based messages, the prompts refer explicitly to the names of the model ingredients, when applicable, to maintain a close connection to the student's model.

An example of the reflective support is shown in Fig. 4.

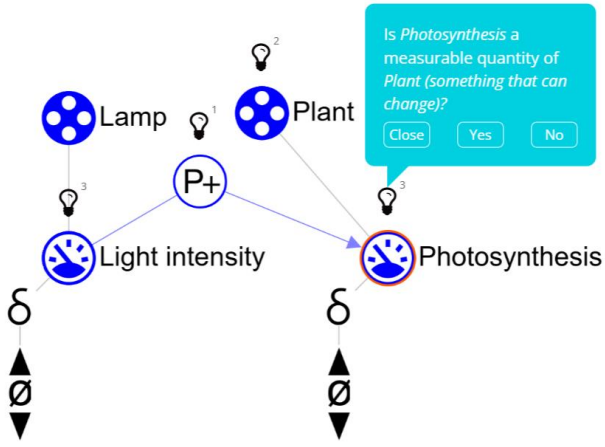

Fig. 4. Reflective support in DynaLearn (grey light bulb). The student selects the grey light bulb associated with the quantity *Photosynthesis*, revealing a set of prompts that guide reflection on whether the quantity is correctly defined and appropriately integrated into the model.

In this example, the student selects the grey light bulb associated with the quantity *Photosynthesis*, revealing a set of prompts that guide reflection on whether this quantity is correctly defined and appropriately related within the model.

The reflective prompts address aspects of model quality related to **correct interpretation and appropriate modelling choices**. For example, students are asked whether a quantity is a measurable property of an entity, whether a configuration represents a structural relationship rather than a causal one, or whether a causal relationship correctly reflects the intended behaviour of the system. These questions do not enforce a single correct answer, but instead support students in evaluating and refining their model based on their understanding of the domain and modelling goals.

In some cases, limited automation is used to tailor prompts, for example by adapting questions when duplicate names are detected for entities or quantities, e.g., *"Does Plant represent a unique entity (different from Plant)?"*, while still requiring the student to evaluate the correctness of the modelling choice. This type of adaptation is not applied to values, as identical values may legitimately occur across different quantities.

Table 3. Overview of reflective prompts (grey light bulb).

Model ingredient	Reflective prompt
Entity	"Is *Plant* a physical object or an abstract concept?"
	"Is the name *Plant* singular?"
	"Does *Plant* represent a unique entity?"
Configuration	"Does *under* describe a structural relationship (and not a causal one)?"
	"The configuration now says: *Plant* → *under* → *Jar*. Is that correct?"
Quantity	"Is *Photosynthesis* a measurable quantity of *Plant*?"
	"Does *Photosynthesis* represent a unique quantity?"
Quantity space	"Does the system exhibit different behaviour for different values in the quantity space?"
Influence relation	"*Photosynthesis* is a process influencing CO_2. Is that correct?"
Proportional relation	"If *Light intensity* increases, does *Photosynthesis* increase?"
Inequality	"Are the values of *Photosynthesis* and *Dissimilation* unequal?"
Calculus	"Is the value of *Photosynthesis* minus *Dissimilation* equal to *Net production*?"*

Note. Some prompts are dynamically adapted based on the model. For inequalities, the wording reflects the selected relation (e.g., *equal* when "=" is used). For calculus, prompts adapt to the selected operation (addition or subtraction) and the specified relation to the target quantity (e.g., equal to, greater than, or less than).
* Not in example model.

Together, the automated vocabulary-based and reflective support features address different aspects of model quality: automated vocabulary-based support enforce formal correctness of the modelling vocabulary, while reflective prompts support students in making informed modelling decisions that depend on domain knowledge and modelling purpose.

4.3 Textualization of the Model

The third support type consists of a **textualization of the model** (referred to as *Wording* in the DynaLearn interface), in which the qualitative model is continuously translated into a natural language description. This representation is shown in a floating text window above the modelling canvas.

The purpose of this feature is to support students in interpreting their model as a **coherent description of system behaviour**, rather than as a collection of separate modelling ingredients. By expressing the model in natural language, students can more easily evaluate whether the constructed model "makes sense" with respect to the system description and their intended interpretation. This supports reflection on both the **semantic correctness** of the model and the **coherence of its structure**.

The textual representation is continuously updated after each create, delete, or modify action. Changes in the text are visually highlighted to draw attention to their effect on the overall description. To improve readability, sentences follow a **consistent format**: the first word is capitalized, the remaining words are in lower case, and each sentence ends with a full stop. Quantities are displayed in blue.

In addition, **interaction** between text and model is supported: hovering over a sentence highlights the corresponding model elements, allowing students to relate the textual description directly to the underlying representation. Fig. 5 illustrates the interaction between the model and its textual representation,

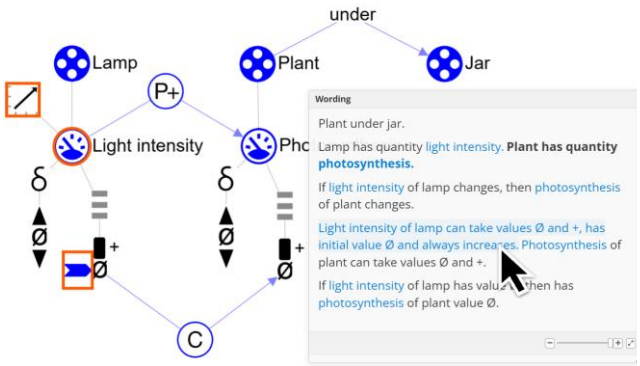

Fig. 5. Textualization (*Wording*) as displayed in DynaLearn. The natural language description is shown above the model and updates dynamically during model construction. When hovering over a sentence (e.g., *"Light intensity of lamp can take values 0 and +, has initial value 0, and always increases"*), the corresponding elements in the model are highlighted.

The textualization is implemented through a set **of rule-based mappings** from modelling ingredients to sentence structures. These rules define how individual elements are expressed, how sentences are merged (e.g., "Plant has quantities photosynthesis and dissimilation"), and how they are adapted to different types and variants of modelling ingredients. For example, correspondences can be expressed in different forms (e.g., directed or bidirectional, between values or between full quantity spaces), and the generated sentences are adjusted accordingly. Likewise, calculus expressions adapt to the selected operation (addition or subtraction) and relation (e.g., equal to, greater than, or less than). Table 4 summarises the main rules underlying the textualization. These rules ensure that the generated text remains both structurally consistent and readable as a coherent description of system behaviour. Fig. 6 shows the complete generated text for the example model.

Plant under jar.

Lamp has quantity light intensity. Plant has quantities photosynthesis and dissimilation. Jar has quantity carbon dioxide.

If light intensity of lamp changes, then photosynthesis of plant changes. Photosynthesis is a process of plant and affects carbon dioxide of jar. Dissimilation is a process of plant and affects carbon dioxide of jar.

Light intensity of lamp can take values 0 and +, has initial value 0 and always increases. Photosynthesis of plant can take values 0 and +. Dissimilation of plant can take values 0 and +, has initial values + and is always constant.

If light intensity of lamp has value 0, then photosynthesis of plant has value 0.

Photosynthesis of plant is initially smaller than dissimilation of plant.

Fig. 6. Full textualization of the example model (Fig. 2).

In addition to these local rules, **sentence ordering** follows a global structure. The textualization begins with the physical configuration of the system (entities and configurations), followed by quantities and causal relationships, and concludes with value-related information, including quantity spaces, correspondences, and inequalities.

Within each section, sentences are ordered according to the **spatial layout** of the model, following a left-to-right and top-to-bottom reading direction where possible. For example, in the quantities section, *"Lamp has quantity light intensity"* is presented before *"Plant has quantities photosynthesis and dissimilation"*, reflecting the spatial position of these entities in the example model.

Table 4. Rule-based textualization of modelling ingredients.

Model ingredient	Sentence structure	Example sentence
Entity	<entity>	Plant.
Configuration	<entity> <configuration> <entity>	Plant under jar.
Quantity	<entity> *has quantity* <quantity>	Plant has quantity photosynthesis.
Proportional relation	*If* <quantity> *of* <entity> *changes, then* <quantity> *of* <entity> *changes*	If light intensity of lamp changes, then photosynthesis of plant changes.
Influence	<quantity> of <entity> *is a process and affects* <quantity> *of* <entity>	Photosynthesis of plant is a process and affects carbon dioxide of jar.
Quantity space	<quantity> *of* <entity> *can take value* <value>	Light intensity of lamp can take values 0 and +.
Initial value / change	< quantity space sentence>, *and has initial value* <initial value>, *and* <exogenous influence> <initial change>	Light intensity of lamp can take values 0 and +, has initial value 0, and always increases.
Correspondence	Between quantity spaces: <quantity> *of* <entity> *and* <quantity> of <entity> *have corresponding values*	Light intensity of lamp and photosynthesis of plant have corresponding values.*
	Between values: *If* <quantity> *of* <entity> *has value* <value>, *then* <quantity> *of* <entity> *has value* <value>	If light intensity of lamp has value 0, then photosynthesis of plant has value 0.
Inequality	<quantity> *of* <entity> *is* <inequality> *than* <quantity> *of* <entity>	Photosynthesis of plant is smaller than dissimilation of plant.
Calculus	<quantity> of <entity> <calculus> <quantity> of <entity> <inequality> <quantity>	Photosynthesis of plant minus dissimilation of plant equals net production.*

Note. Model-derived ingredients are indicated using angle brackets (<>), while italicized words represent fixed parts of the generated sentence. Sentences are merged whenever possible. Sentences are grouped into paragraphs with whitespace separating structural, behavioural, and value-related descriptions. No distinction in wording is made for positive or negative proportional or influence relations. Values within a quantity space are expressed from low to high. * Not in example model.

4.4 Scenario Advice for Simulation

The fourth support type consists of **scenario advice**, which supports students in defining appropriate initial settings for running a simulation. This feature was developed in earlier work for more structured modelling tasks [5,6] and is also applicable in open modelling tasks. Here, we provide a brief description of its functionality.

When a student attempts to run a simulation, the scenario advisor inspects the current state of the model and identifies issues related to the specification of initial settings. These include **missing settings** and **inconsistent or superfluous settings** that may prevent meaningful proper simulation. Feedback is presented through a **red exclamation mark** displayed near the corresponding model ingredient, with a counter indicating the number of messages. When the student clicks the icon, a message is shown explaining the issue and suggesting how it can be resolved.

An example is shown in Fig. 7. In this case, the student attempts to run a simulation for a model in which the quantity *Light intensity* has a defined quantity space with values 0 and + but no initial change or value is set. The system detects this and provides a message prompting the student to assign an initial change ("Assign initial change to quantity?"), with the exclamation mark indicating that two messages are available, of which this is the first.

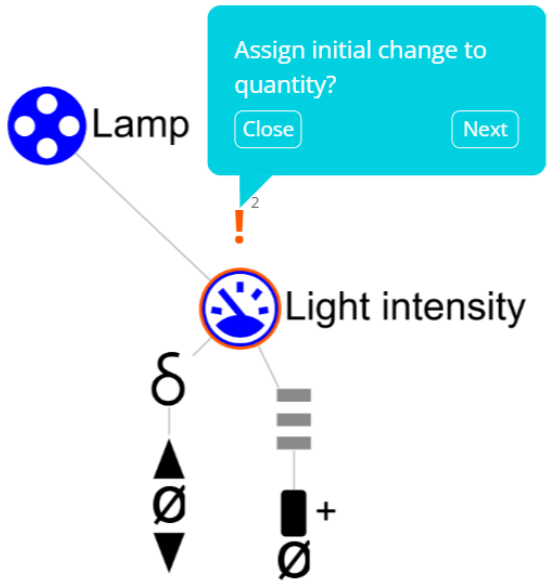

Fig. 7. Scenario advice. The student attempts to run a simulation without specifying an initial change and a value for the quantity *Light intensity*. The system detects this and displays one of the two messages (*"Assign initial change to quantity?"*).

In addition to supporting the specification of initial settings, the scenario advisor also highlights **feedback loops**. Two types are distinguished: positive feedback loops, in which changes are reinforced, and negative feedback loops, in which changes are dampened. Highlighting these structures helps students to recognise and interpret key dynamic features of the system's behaviour.

This form of support addresses a different phase of the modelling process than the previously described features. Whereas automated checks and reflective prompts focus on model construction, scenario advice supports simulating a **model**. By guiding students in defining appropriate initial settings and interpreting simulation results, it helps ensuring that the model can generate meaningful behaviour and supports reasoning about system dynamics.

Taken together, these support features illustrate how different types of knowledge required for modelling can be operationalised in a modelling environment, by combining automated feedback with opportunities for student reflection and interpretation.

5 Discussion

In this paper, we presented a set of complementary support features designed to support students in constructing qualitative models in open modelling tasks. The support addresses different aspects of the modelling process: (*i*) automated checks enforce correctness and completeness with respect to the modelling vocabulary, (*ii*) reflective prompts support decision-making requiring domain knowledge and knowledge of the modelling purpose, (*iii*) textualization supports interpretation of the model as a coherent description of system behaviour, and (*iv*) scenario advice supports the specification of initial settings and the generation of meaningful simulation behaviour.

A key design principle underlying these features is the balance between **guidance and openness**. Rather than steering students towards a predefined solution, the support aims to scaffold the modelling process while preserving students' autonomy in making modelling decisions. This is particularly important in open modelling tasks, where learning opportunities arise from interpreting the system, evaluating modelling choices, and refining representations [11].

At the same time, the complexity of such tasks should not be underestimated. Even with support, constructing a qualitative model of system behaviour remains challenging for students, as it requires the coordination of multiple types of knowledge and the integration of model components into a coherent whole. Moreover, it is well established that students do not always make **effective use of available support**, especially when it is optional or requires active engagement [12]. This raises important questions about how support should be designed and integrated to maximise its educational impact.

An additional consideration concerns the **complexity of the modelling task itself**. Factors such as the domain being modelled and the level of the modelling vocabulary (e.g., level 2 versus level 4 in DynaLearn) strongly influence the cognitive demands placed on students. These factors can be seen as "knobs" that can be adjusted to balance task difficulty and learning opportunities. Future work should investigate how support features interact with task complexity and how both can be tuned to different student levels.

The work presented here focuses on the design and implementation of support features. A next step is to **empirically evaluate** their effectiveness in supporting students' modelling processes and learning outcomes. Such evaluation should not only consider improvements in model quality, but also how students engage with the support and how it affects their reasoning about system behaviour.

Looking ahead, several directions for further development can be identified. One direction concerns the **personalisation of support**. In the current implementation, support features are presented uniformly to all students. However, their use could be adapted based on students' needs and modelling performance. For example, automated vocabulary-based checks are currently always provided when relevant, but could be selectively enabled or prioritised based on an assessment of the student's proficiency. Similarly, reflective prompts could be adapted more strongly, as the number of prompts may quickly become overwhelming as models grow in size and complexity. Personalising the type, timing, and amount of support may therefore help to balance guidance and cognitive load, and improve students' engagement with the support.

Another direction is to provide support for **interpreting simulation outcomes**, complementing the current focus on model construction and simulation setup. In addition, support could be extended to the **modelling process itself**, for example by encouraging an iterative approach in which students construct partial models, test them through simulation, interpret the results, and refine their model accordingly [4,11].

In conclusion, the presented support features provide a structured approach to supporting open modelling tasks by addressing different aspects of modelling knowledge and practice. They offer a promising basis for further research into how students can be effectively supported in learning by constructing and reasoning with qualitative models of dynamic systems.

References

[1] Jacobson, M., & Wilensky, U.: Complex systems in education: Scientific and educational importance and implications for the learning sciences. *The Journal of the Learning Sciences*, 15(1), 11-34, 2006.

[2] National Research Council: *Next generation science standards: For states, by states*, 2013.

[3] VanLehn, K.: Model construction as a learning activity: A design space and review. *Interactive Learning Environments*, *21*(4), 371-413, 2013.

[4] Sins, P., Savelsbergh, E., van Joolingen, W. R.: The difficult process of scientific modelling: An analysis of

novices' reasoning during computer-based modelling. *International Journal of Science Education*, 27(14), 1695-1721, 2005.

[5] Kragten, M., & Bredeweg, B.: The effectiveness of lightweight automated support for learning about dynamic systems with qualitative representations. In *Proceedings of the 39th ACM/SIGAPP Symposium on Applied Computing* (pp. 11-20), 2024.

[6] Bredeweg, B., Kragten, M., Holt, J., Kruit, P., van Eijck, T., Pijls, M., Bouwer, A., Sprinkhuizen, M., Jaspar, E., & de Boer, M: Learning with interactive knowledge representations. *Applied Sciences*, 13(9), 5256, 2023.

[7] Bredeweg, B., Liem, J., Beek, W., Linnebank, F., Gracia, J., Lozano, E., Wißner, M., Bühling, R., Salles, P., Noble, R. A., Zitek, A., Borisova, P., & Mioduser, D.: DynaLearn - An intelligent learning environment for learning conceptual knowledge. *AI Magazine*, 34(4), 46-65, 2013.

[8] Forbus, K. D.: *Qualitative representations: How people reason and learn about the continuous world*. MIT Press, 2019.

[9] Liem, J.: Supporting conceptual modelling of dynamic systems: A knowledge engineering perspective on qualitative reasoning. PhD thesis, University of Amsterdam, The Netherlands, 2013.

[10] Kragten, M., Hoogma, T., & Bredeweg, B.: How to Assess the Quality of Student-generated Qualitative Models during an Open Modelling Task? In *Workshop on advances in Artificial Intelligence for Exploratory Learning (AI4EXL): 26th International Conference on Artificial Intelligence in Education,* 2025.

[11] Bielik, T., Opitz, S. T., Novak, A. M.: Supporting students in building and using models: Development on the quality and complexity dimensions. Education Sciences 8(3), 149, 2018.

[12] Aleven, V., Mclaren, B., Roll, I., & Koedinger, K.: Toward meta-cognitive tutoring: A model of help seeking with a Cognitive Tutor. *International journal of artificial intelligence in education*, *16*(2), 101-128, 2006

[13] Bredeweg, B., Liem, L., Beek, W., Salles, P. & Linnebank, F.: Learning spaces as representational scaffolds for learning conceptual knowledge of system behaviour. In: M. Wolpers, P. A. Kirschner, M. Scheffel, S. Lindstaedt, & V. Dimitrova (Eds.), Technology Enhanced Learning, LNCS 6383, 46-61, 2010.