Learning Analytics Metamodel: Assessing the Benefits of the Publishing Chain's Approach

Camila Morais Canellas, François Bouchet, Thibaut Arribe, Vanda Luengo

Learning Analytics Metamodel: Assessing the Benefits of the Publishing Chain's Approach

Camila Morais Canellas[1,2][0000−0002−7226−0931], François Bouchet[1][0000−0001−9436−1250], Thibaut Arribe[2], and Vanda Luengo[1][0000−0001−8978−0944]

[1] Sorbonne Université, CNRS, LIP6, F-75005 Paris, France
[2] Société Kelis, France
{camila.canellas, francois.bouchet, vanda.luengo}@lip6.fr, thibaut.arribe@kelis.fr

Abstract. In this work, we propose a learning analytics implementation based on a model-driven engineering approach. It aims at assessing the benefits that could arise from such an implementation, when pedagogical resources are produced via publishing chains, that already use the same approach to produce documents. Previously, we have discussed these potential benefits from a more theoretical point of view. In the present work, we present a concrete implementation of a metamodel to integrate a learning analytics system closely linked to the knowledge of the semantics and structure of any document produced, natively. Finally, we present an initial evaluation of this metamodel by modelers and discuss the limits of this metamodel and the future changes required.

Keywords: Learning Analytics (LA) · Model-Driven Engineering (MDE) · Metamodel · Publishing Chains.

In this work, we describe a metamodel for a learning analytics (LA) solution into a platform framework that uses a model-driven engineering (MDE) approach to design and publish pedagogical resources via publishing chains, as well as a first evaluation of the metamodel proposed.

Learners' interactions with courses and resources offered by learning platforms on the Web have generated a vast amount of learning-related data. For a decade now, the interdisciplinary field of Learning Analytics (LA) has focused its attention on the collection, processing, and analysis of such data. A common definition of learning analytics is: "Learning analytics is the measurement, collection, analysis, and reporting of data about learners and their contexts, for the purposes of understanding and optimizing learning and the environments in which it occurs" [16]. According to [17], when applying learning analytics, one can focus on what e-learning interaction traces need to be captured, how to process them and how to present them to stakeholders in a useful way. Although many times the analyses carried out in these systems are based on traces already collected [14], there is currently no consensus on the interactions actually relevant for effective learning [1] and, subsequently, on the traces to be recorded and analyzed.

Our work takes place in a context where detailed knowledge of the structure and semantics of the documents learners interact with is known a priori. We posit that this knowledge represents an asset which value has to be assessed in the context of an LA implementation. Our interest relies on how to natively include this detailed knowledge of the structured document in the trace analysis cycle. The idea is to illustrate how this knowledge a priori can be used to enhance educational resources produced by a set of MDE authoring tools. The ultimate aim is to create solutions that support stakeholders in making data-driven decisions in order to improve learning, while also taking advantage of the existing context of publishing chains to do so.

The rest of this work is structured as follows: in section 1 we situate our research with related literature regarding models for learning analytics, in section 2 we define the model-driven engineering approach we use and in section 3 we describe how the same approach is already used to create pedagogical materials — resulting in the semantics and structure knowledge of such documents to be known beforehand, as illustrated in section 4. In section 5 we present the details of the proposed metamodel dedicated for the application of LA processes. In section 6 we describe the first evaluation of the metamodel proposed. A discussion can be found in section 7. This work summarizes parts of [4] as the present work is an extension of this previous work. More particularly, it adds a more detailed description of the proposed metamodel, as well as an initial evaluation based on its use by two expert modelers. It relies on the same context and illustration of the processes already described, which are necessary for understanding of the stakes at hand. On the other hand, the present work is less focused on the potential benefits of our approach and on the learning analytics processes, previously discussed in [4].

1 Related Work

The underlying assumption [14] of a number of LA solutions is that the interaction data is either already at hand, or recordings of all student interactions with a given system will start to be collected, usually then it is decided to use one of the available standards, such as IMS Caliper or Experience API (xAPI), among others. Moreover, one may argue that a result of the "lack of staff and technology available for learning analytics projects" [12, p. 366] would favor an outsourcing of the issue of traces collected, and a standard would be used, without consulting a local expert for the establishment of these.

Besides the fact of having competing specifications, it is not clear at this time whether one or the other works better [14]. One of the aspects related to these specifications that has received criticism is the lack of relevant information if one follows these specifications by implementing only the mandatory aspects (*required*). The same authors [14] point out that this lack of relevant information leads to situations where we consider the system to conform to one or another specification via certification tests, but in practice it produces data streams with

largely redundant events of one type, or that do not describe any user behavior that could be useful for an intended analysis.

An effort on enriching traces via models found in the literature is Trace-Based Reasoning (TBR) [8]. The so-called obsels are formally described in the trace model and each is characterized by a name, a timestamp, and a set of properties, usually attribute-value pairs. The authors briefly mention that a metamodel should define these properties in order to ensure interoperability between different traces.

Researchers behind TBR mention that the name "obsel" is chosen — instead of event — in order to emphasize that an obsel is recorded with a purpose (defined a priori). This idea to think about the traces beforehand and enrich them with context information is a common point with our approach here. However, a difference lays in the fact it seems interesting to us to explore how to guide this modeling of traces not only a priori and via modeling, but also starting from a specific question, translated into one or more indicators. In other words, it is because we are trying to answer a question that we record interactions, but also, traces are potentially enriched with some information about the documents read.

Among the works going towards models more closely related to learning analytics, in [5] the authors present a dedicated metalanguage defining both data and needs: *Usage Tracking Language* (UTL). Its components allow defining information to enrich the collected traces and to structure the information through the definition of intermediate data and indicators. This model has the advantage of adapting to the elements it describes via the addition/modification of information already available, and its focus is to enable stakeholders to take educational scenarios into account in the process of collecting and analyzing traces.

The metalanguage is made of three parts. First, its UTL/T (traces) part makes it possible to represent the transformation of the traces generated by describing them as a set of data including information on their source. One may also define the relationship of the latter with the expected primary data in order to create an indicator, as well as define the information extraction method from the traces. Second, its patron part (UTL/P) is based on the DGU (Defining, Getting, Using) model and proposes the usage trace before the learning session [6] and not once the traces have been collected. This aims to allow a future comparison of descriptive scenarios with predictive scenarios of the educational situation, and therefore make it possible to describe the structure of an observable. Finally, the UTL/S (pedagogical scenario) part makes it possible to semantically link the pedagogical scenario to a given indicator.

This work shows us the benefit of enriching the analyses (and traces to do so) with context information that can be modeled, in this case, from the educational scenario, in our case, with the detailed information from the various documents consulted, and natively in a previous phase by the modeler (what could relieve the author from defining the traces).

To conclude, here we see an effort made to model the traces and indicators, but also the educational scenarios, so that one can enrich the other during the

analyses. Our approach is therefore close to those cited, but we intend to explore an approach guided by educational documents in order to complete and enrich traces. As we have mentioned, documentary production in our context already relies on modeling, which allows us to explore this fine knowledge of the document in the context of learning analytics.

We sustain that, by allowing to specifically define, at the same time, both the desired LA indicators and the educational resources' production, we could enable both the document and its usage analysis to be closely linked. In order to do so, we use an MDE approach both to produce documents — already happening in our context — and analyze their usage — also via an MDE approach —, that has been used in other domains such as automotive, banking, printing, web applications, among others [11], yet not as much in LA systems.

2 Methodology: Models

When developing complex systems, model-driven engineering is an approach that allows one to focus on a more abstract level than classical programming [7]. It allows a description of both the problem at hand and its solution. The focus is to create and leverage domain models — in our case, the LA one — as a conceptual model taking into account all the topics related to this specific task/domain. Model-driven engineering is considered as a form of generative engineering [7] as some or all of a computer application is generated from models. The procedure includes simplifying aspects of reality (or a solution to a problem) by creating a model or a set of models. Thus, a model is a simplified representation of a concept or object, which can be more or less abstract/ precise. Likewise, a metamodel is a model which defines the expression language of a model, i.e., the modeling language [13].

Digital publishing chains are an example of systems based on MDE, as they use this approach to create document models that will then be used in document production. In our context, all pedagogical documents are produced using digital publishing chains; we provide the details of this publishing process in the next sections.

3 Models for Publishing Educational Documents

Some documents, such as pedagogical resources, often have a well-known consistency in structure, i.e., the parts that compose the document, and in semantics, i.e., what type of content (a definition, an example) corresponds to each part. Digital publishing chains can assist the production and publication of such structured documents, especially in their design processes [2]. A publishing chain can be defined as "a technological and methodological process consisting in producing a document template, assisting in content creation tasks and automating formatting" [9, p.2, our translation].

One of the benefits stemming from the use of digital publishing chains is the documents' homogeneity, even when editing and publishing large amounts

of documents. The models' definition that documents are based on is strongly linked to the profession or associated context. The document needs are assessed and then formalized in the appropriate model, therefore when an author decides to use one document model over another, he or she reflects a given intention.

Moreover, such systems have the advantage of separating content and form [3], allowing for authors to focus on the content itself. The form is later applied, automatically, when the document is generated. Thus, the same content could have different forms of presentation according to the publication format, which can include PDF, Web, among others [3]. Other features of such systems are making teaching practices explicit, sharing of practices, optimizing production management, and reducing costs in document production [10].

Typically, a document will be based on a document model, the document model is itself based on a metamodel, the metamodel can be based on a metametamodel and so on, depending on the level of abstraction needed. In our context, we consider "document primitive" which is a computer code abstracting documentary objects that later in the chain will allow the generation of specific code instantiating multiple document models [2]. This process is illustrated in the next section. Therefore, we propose to follow a similar approach for our LA modeling. For our metamodel, we propose to abstract the various elements related to implementing a learning analytics solution ("analytical primitives") via a metamodel, in order to design learning indicators. Unlike the approaches described in the state of the art, we suggest that the definition of indicator models should be done at the same time as the definition of documentary models (and not a posteriori), which makes it possible to take into account detailed information of the structured documents in question. This approach has the advantage of allowing the definition of the necessary traces (and their contents), thus reversing the usual approach [14] of starting from the traces to arrive at indicators.

4 The Publishing Processes

As mentioned before, in our particular context, **developers** have defined "building bricks" called document primitives. They are the building blocks that will be used as the basis for the creation of a document's model.

In the next phase, those bricks are made available to a modeling tool that allows the creation of any document model desired using and combining those initial bricks. In other words, the **modeler** defines a document **model** using the available document primitives.

When the document model is ready, it will be made available to a writing tool dedicated to structured documents. An **author** (a teacher or a pedagogical team, for example) uses this document model in order to create his or her **document** according to the model chosen (a course module, a case study game, etc.). The writing tool (s)he uses includes the transformation algorithms that automatically publish the document at the desired format. We are obviously interested in the **document** in its Web format and in the ways in which it will later be used by its final **users** (learners).

The challenge of choosing such an approach is to be able to abstract the technical aspects as much as possible — specially at the metamodel level — in order to reduce the work and facilitate the creation processes as the phases at the end are more numerous and stakeholders' roles are less technical.

4.1 Illustrating Our Context

As briefly described before, a number of phases take place and several stakeholders may intervene. In this section, in order to better illustrate these processes taking place when designing documents for educational purposes via model-driven engineering, we describe next a — highly simplified — practical example. Note that we first reported this example in our previous work (cf. [4]). In this example, we rely on the vocabulary of the existing Opale document model[3], used for creating linear academic courses (on-site training, distance or blended learning). It is worth noting that other document models can be used, for pedagogical purposes or not, such as to carry out case studies with a gamified twist, create exercisers with different modes of execution (self-learning, evaluation), build question banks for evaluations, among others.

First Phase – The Developer The first phase consists in providing the so-called document primitives and is done by the developers of the tool(s). Those could be, for our example, "text", "multimedia", "quiz" and "organization". Each of these bricks (see Fig. 1) will be used on the next phase to build the document model.

Fig. 1. First Phase: Metamodel with the document primitives for our illustrative document modeling, from [4]

Second Phase – The Modeler A modeler then uses the available primitives on the design tool to define document models. The challenge of this phase is to understand the needs of a given community of users, and translate them into a document model. The elements created, their organization and the language used are therefore appropriate for this group of users and the domain in question. Certainly, the same applies to the available transformation functions that translate the models into actual documents in various formats.

[3] Available at: `https://doc.scenari.software/Opale/en/`

In our example, the modeler uses the organization primitive to define a "learning activity" as being made of:

- "Introduction" (text primitive)
- "Concept" (text or multimedia primitives)
- "Content" made up of parts (text and multimedia primitives) titled "Information" and "Example"
- "Conclusion" (multimedia primitive)
- "Practice" made up of quizzes (quiz primitive)

As seen in Fig. 2, the modeler also defines that a learning module can have one or more "learning activities". A "learning activity" must have exactly one "introduction", one "conclusion" and a "practice" part at the end, and include between "introduction" and "conclusion" one or more "concept" and/or one or more "content" parts. A "practical" part must have one or more quizzes.

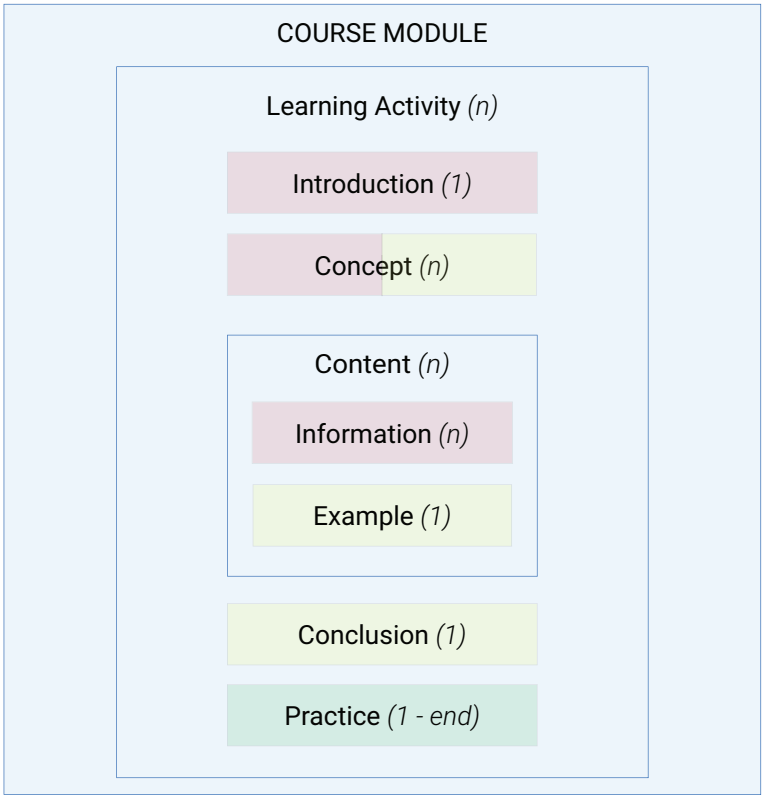

Fig. 2. Second Phase: Metamodel used to define the model for our illustrative document modeling, adapted from [4].

Regarding the generation of the document in a Web format, the modeler may choose to have a page created for each "learning activity". He or she could also define a menu reflecting the structure of the document, allowing users to browse the module, either by the pages names or by jumps to each internal part of the pages, the blocks mentioned above. The document model is at last made available for authors to use it on the editing tool.

It is important to observe that it is during this phase that the modeler defines both the structure (each block and its optional/mandatory components) and the semantics (the nature of what each block is supposed to contain) for each document based on this particular document model. Note that this information will also be used by the publishing algorithm. In other words, the different possible "parts" of the document are pre-established at the time of modeling and the content type of each part is defined by the chosen blocks.

Third Phase – The Author An author, such as an instructor, uses this document model to create his or her course. Its course consists of four "learning activities", each with an "introduction", two "concepts", four "contents", a "conclusion", followed by a "practice" activity to check the understanding of the theoretical content (see 3). Once the course has been created, the instructor can publish it, for example in a Web format.

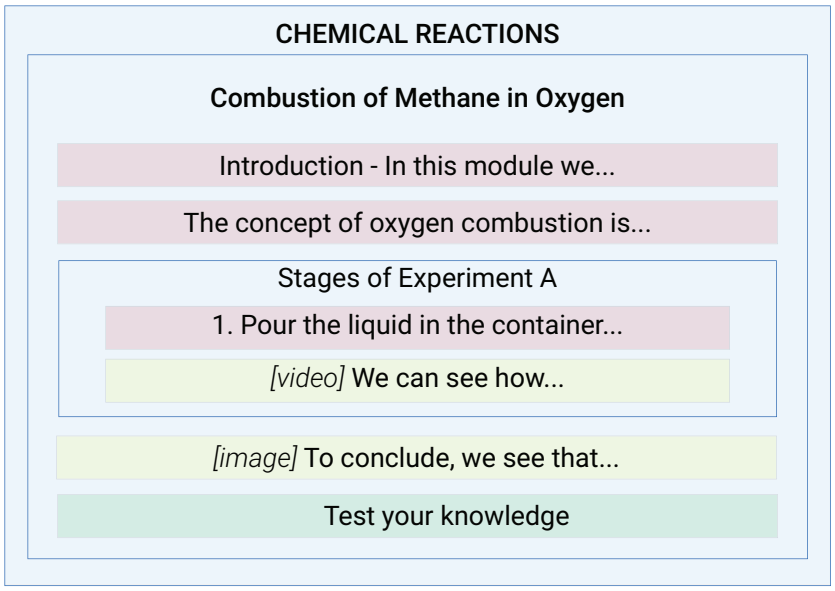

Fig. 3. Third Phase: Document model used in the instantiating process of our illustrative document modeling, from [4].

Choosing a document model over another reflects the intentions of the author. Moreover, when creating his or her document, each choice regarding each block used also represents the intention regarding its *content*: he or she decides it is time to introduce a "practical activity" to check the understanding of the new "concept" just introduced — and not simply: (s)he adds a quiz after a subsection.

Thus, 1/ the author must choose the model corresponding to the needs of his or her profession (or type of course, in our case) and 2/ this will also have consequences in trace analysis phase, in particular regarding semantics analysis.

Fourth Phase – The Learner Finally, learners access the published content, open pages in whatever order they want, scroll through them, take quizzes, etc. See Fig. 4 for an example of a published document.

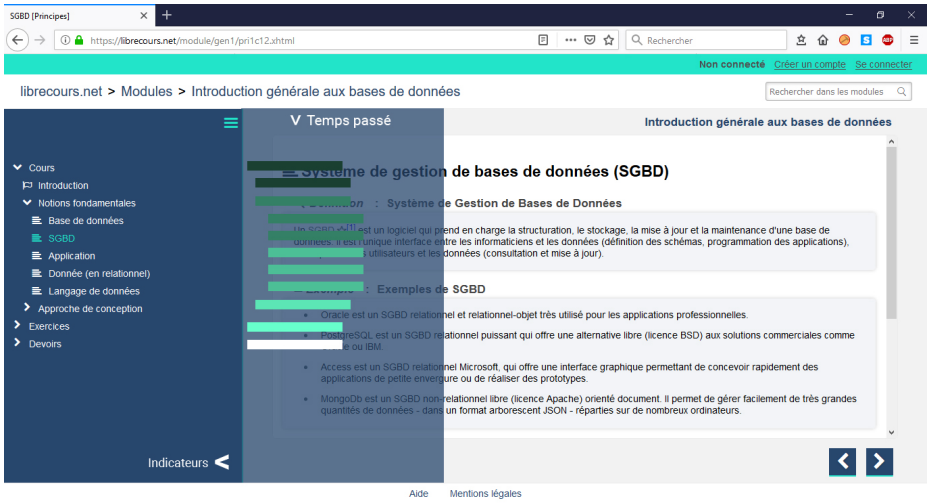

Fig. 4. Fourth Phase: Document transformed into a Web format, ready to be used by learners. We added as an example the visualization of an indicator following the menu (structure) of the course, from [4].

5 Models for Learning Analytics

We firstly decided to have indicators as the core of the LA metamodel. So this concept was isolated and defined as "a learning indicator used in a learning analytics approach is a calculated measure [computability property] linked to a behavior or an activity instrumented by digital technology [traceability property] of one or more learners, given to a user [visibility property] and which can be used in the calculation of other indicators."

More specifically, according to this definition, the following elements are not learning indicators:

- Page X was seen or not by learner Y [not respecting the property of computability, as it is directly traced this way].
- The gender of a learner [not respecting the properties of traceability and computability].
- The motivation if it comes only from declarative data, such as a questionnaire [not respecting the property of traceability].
- The number of pages viewed by a learner in a system using this data to calculate a dropout risk indicator, if only the latter is reported to the instructor (because the number of page views is then only a variable internal to the system, not accessible by a user) [not respecting the visibility property].

These three key properties will impact the proposed metamodel below, as it represents the first element of it from which the others derive. Additionally, we used this definition to conduct a systematic review of indicators used by the learning analytics community [to be published soon]. The goal of this work was to ensure that our metamodel could indeed be used to transpose the most commonly used indicators from the LA community, and to avoid defining a metamodel that is too specific to some particular indicators. Indeed, the metamodel proposal must make it possible, once instantiated, to provide the essential specifications to the various transformation and generation processes. The designed metamodel is made up of three main dimensions:

- Interactions with the document: these are the potential sources of traces of interactions that can be recorded.
- Indicators: the core of the metamodel, with the primitives used to define an indicator.
- Visualization modes: allowing to decide how the visualization of the results obtained will be offered to the different stakeholders.

The core of the metamodel is clearly the "indicators" dimension, as it is the selection of an indicator which will trigger or not the addition of a trace to trace log (among all the possible ones associated to each interaction) and the visualizations depend on the nature of the selected indicators (e.g. a bar chart can be preferable to a pie chart) and they can be seen as more general parts of a visualization system (e.g. deciding to group all indicators on a dashboard page or embedding them into the learning content as shown in Figure 4). Thus, we will describe in detail here only the "indicators" dimension (see Fig. 5).

The main primitive allowing to model the indicator (`indicator` on Figure 5) has some traditional attributes regarding its identification (`code`, `name` and `description`), as well as the attribute used to indicate who has access to this primitive (the type of `user`). This last aspect is already present and modeled in the system in question, and therefore a link between these primitives will be enough concerning the proposed primitive. The other three attributes refer to the inputs (`inputs`), analyses (`analysis`) and outputs (`output`) required to define

the indicator that will be modeled. The calculation relating to the indicator must also be indicated in this primitive.

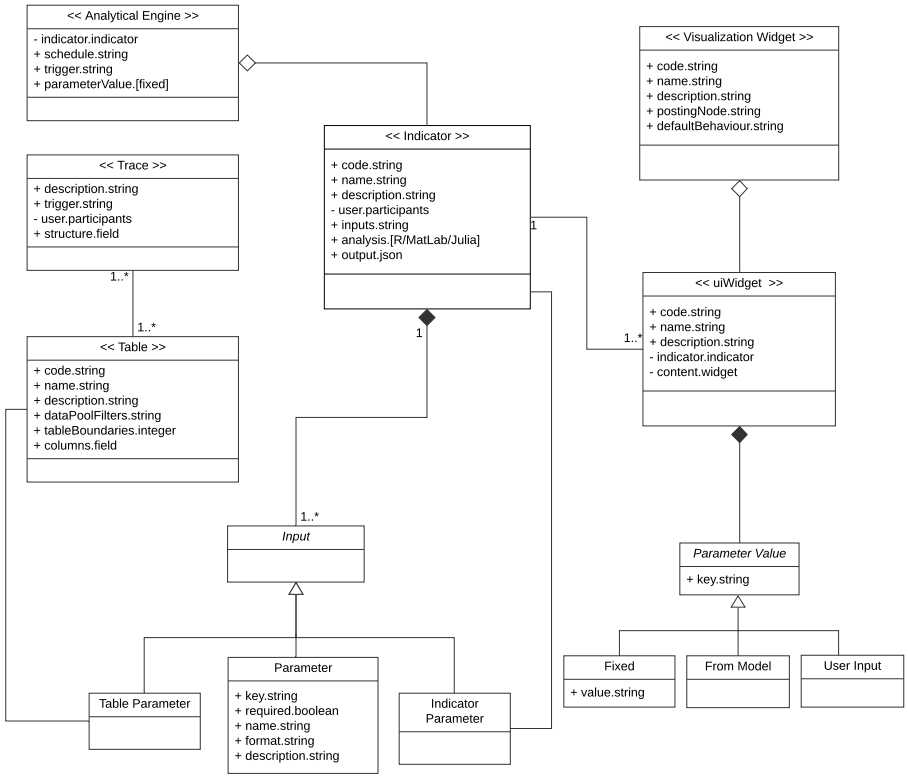

Fig. 5. Metamodel: creating indicators.

Several types of inputs can be provided:

- A parameter: this is to define a necessary entry that is flexible enough for the modeler, for example a start and end date that will be used to perform the analysis from the log traces, or a number which will be used as a threshold to trigger a message to a user, etc.
- A table: an entry can also be a table with data more or less ready to be used by the analysis in question.
- An indicator: some indicators can have as input another indicator calculated beforehand. For example, an indicator aiming to display the involvement of a learner could have as input the time spent on content or the number of connections, etc.

The traces that will be produced from the interaction with the documents are defined in the corresponding primitive (`trace`). It is above all a matter of de-

termining the type of interaction (opening a page, clicking on the play button of a video, etc.) triggering the tracing, but also the documentary components to which this rule must be applied, such as "trace all openings of pages that include a "conclusion" tag". This is done through the documentary model, via an identification key. Note that some information for each trace will be added by default and therefore does not need to be modeled each time. These are mainly elements either not necessarily related to the analyses but technologically necessary or a convention always needed, for example the address of the page, project identification on the server, or the timestamp.

Another primitive (`table`) aims to allow modelers to define tables from raw traces' logs, which would be ready to use for the calculation of certain indicators. This primitive must make it possible to determine the filtering of the traces taken into account to fill the table in question. A Boolean system allows this selection, for example by choosing (`typeBlock = concept AND (focus = started OR focus = ended)`). It is also possible to define calculations to be performed during filling, for each column, if necessary. Then, the modeler must be able to define the columns of this table and the information available in the traces which will fill each of these them, as well as the corresponding type of data. It is also possible to set the retention time of such data, for technical purposes (freeing up memory space) but also for compliance to data protection regulations, such as the European GDRP[4]).

The `Analytical Engine` primitive is used to model the triggers for indicator calculations. It is with this primitive that the modeler determines the scheduling of the actual calculations for each indicator. Parameters, such as start and end dates, can also be defined here. This should allow, for example, to gain in performance for indicators where an incremental calculation avoids recalculating it from all the data (e.g., an indicator showing the number of new concepts seen daily by a learner could be computed every day at midnight and only the traces for the last 24 hours would need to be analyzed and added to the existing indicator).

6 Testing and Evaluation

Once the primitives of the metamodel have been implemented in the tool dedicated to modeling (an example is given in Fig. 6), we sought to verify the correct understanding and handling on the part of its future users, the modelers. Note that the different transformations were not all yet implemented at the time, so the study focuses on getting started with the metamodel. However, understandability by the first-level users (modelers) is key to ensure the proposed LA metamodel will be actually used.

[4] `https://gdprinfo.eu/`

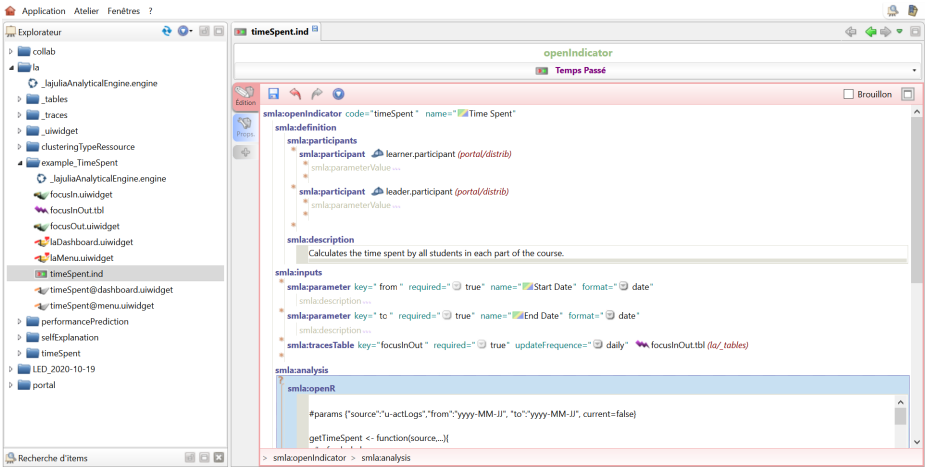

Fig. 6. Metamodel: initial implementation in the dedicated tool, from [4].

6.1 Statement of the problem

The main issue is to understand the usage by the modelers of the proposed LA metamodel by focusing on how they go about creating indicators using the proposed primitives. It entails understanding several aspects — utility, advantages/disadvantages, understanding, sticking points, etc. — of the use of the metamodel by experienced modelers. We were particularly interested in observing the following points:

- How they get started: what actions are taken, with what goal and in what order (planned vs. actual actions).
- Blocking or improvement points: anything that could lead to a hesitation or possible improvements when modeling an indicator.
- Their perceptions on several aspects: the advantages/disadvantages of our approach.

Research questions More precisely, the questions we seek to answer with this study are:

1. How is the handling of the metamodel by modelers?
2. Are the primitives used as expected?
3. What are the blocking points during this first usage?
4. Are there any improvements to be expected, which ones?
5. Is the vocabulary used clear and understandable to a modeler?
6. What is the perceived usability of the metamodel by modelers?
7. How are the proposed primitives used to encourage maintainability, reuse, and customization of indicators?

Delimitations and limitations Modelers are experts in document modeling and so quite familiar with metamodels in general, however they are not necessarily experts in LA. The first phase of the study therefore had to be conducted with care in order to help them understand these new concepts and vocabulary.

Qualitative research approach used This study is based on a triangulation comprising three phases:

1. Presentation of the model with a concrete example of an indicator.
2. Modeling (by experienced modelers) of an indicator in autonomy and response to a questionnaire (System Usability Scale and phrase completion).
3. Explanatory interview on the modeling of an indicator in autonomy (done in phase 2). At the end, modelers are asked to change the indicator that was modeled, creating a new one.

Researcher's role The role of the researcher in this study is that of a guide.

Sampling method A targeted sampling method [15] is used, that is to say the sampling by criterion — which consists in selecting cases who satisfy a predefined criterion — in our case, experienced modelers.

6.2 Results: interviews with modelers

Each interview with the two participating modelers lasted an average of one hour. They did the exercise of telling their actions, primitive by primitive, in order to create the model of the proposed indicator. During the interview, the discussion was based on this modeling, with screen sharing.

The first modeler started by setting the primitive allowing to define the traces to be recorded (`Trace`) and the corresponding table (`Table`). He stated he understood how to define what will be added into the published document and then will feed the trace logs. He had questions about some information, such as the time stamp, which was not to be modeled, since it was part of the information automatically inserted in the traces. The researcher confirmed that some information were automatically collected for all traces, information transmitted during the model presentation session but not stored. A list with this information should be provided in a guide to modelers to help them better understand this point and which information does not need to be modeled. Other than that, the modeler claimed to be confident in what he was doing. A suggestion made by the modeler consisted in adding the possibility of indicating on which element(s) of the interface the traces should be created. For example, if we want to target only the clicks on one button of a page and not all, by indicating the identifier of this button coming from the documentary model (at the moment there is no unique identifier created automatically for each button, therefore changes would have to be considered in the documentary model first).

Then the first modeler affirmed to have set the primitive corresponding to the (`Table`) that is going to be created from the traces: "That, too, is a part that I have fully understood". Regarding the filters making it possible to choose the information from the traces in order to fill the rows and columns that correspond to the need, the logic seemed to be well understood and easy to use. Here, often, the information inserted automatically in the traces must be used, which reinforces the previous remark. The modeler advised that these options could be offered by the system in the form of lists, so that the modeler would only choose from the ones that already exist — the benefit being that the information of what exists by default would also be presented that way.

The next primitive by the modeler was the one that allowing to set information about the indicator itself (`Indicator`). The modeler linked the table necessary as input to the computation carried out subsequently. The output is conditioned by this calculation, and it should be noted that the corresponding typology should result in a defined list, the values of which have not yet been decided and that therefore this field remained open for the moment. The objective is to have, in this field, the type of output expected, which will be used as information for the visualization chosen below (a list, a matrix, etc.). Because of that open field, the modeler asked for details, but understood the purpose of the field without problem.

The widget which allows for preparing the visualization of these results (*Visualization Widget*) was used later. For the indicator that served as an example, the input parameters (start date and end date) came from the instructor, but this could also be set by the modeler at the time of modeling, etc. The modeler did not understand that this is where the origin of this information is defined. With the explanation in this interview, the modeler understood how to define these inputs in several ways using this primitive.

Then, the (`uiWidget`) primitive was used by the modeler, for modeling in terms of the chosen visualization means (dashboard, alongside the menu, as a prompt message, etc.) for each type of user. The modeler suggested changing the naming of the primitive in order to have the term "visualization" appearing in it, making it easier to find/understand among all the available primitives. As a reminder, during this study, the details concerning visualizations type of graph, etc. were not yet implemented.

Finally, the modeler used the (`Analytical Engine`) primitive in order to define the periodicity of the calculations.

Overall, the modeler mentioned that the use of the primitives of the metamodel required some back and forth actions for certain parts, where he realized a need and had to return to a previous primitive before continuing. These actions are natural and even expected, they prove a step-by-step understanding on the one hand of the metamodel, and on the other hand of the indicator itself.

Nonetheless, according to the first modeler, the part of the metamodel corresponding to the visualization seemed a little more complicated, in particular the definition of the inputs as the parameters, but has nevertheless been understood.

This difficulty could be caused by the fact that the metamodel corresponding to this dimension could not be implemented in its entirety at the time of the tests.

Regarding the naming, the other primitives — apart from (`uiWidget`) already mentioned — were easy to navigate. Finally, the modeler said:

[...] I found it to be concise as a metamodel anyway, there weren't that many [primitive] items and that's not bad.

About the change requested by the client (last task in phase 3), the modeler was able to make the necessary changes: "I would start by changing the traces". Subsequently, he affirmed:

[...] If I have to change from one event to another, yes, the fact that it's taken care of by the metamodel is pretty quick.

The second modeler reported that he started with the (`Indicator`) primitive. Using it, he realized which inputs he was going to need. Thus, he then took in hand the primitive relative to the traces (`Trace`), so that these entries were traced. He suggested changing the internal term for this primitive from (`fromModel`) to (`callModel`) in order to keep the nomenclature already existing in the tool regarding these situations.

Next, he determined the table (`Table`) built from the traces. The modeler noted that this primitive is not explicitly related to the traces. This type of link is common among the primitives of the tool used, making it possible to determine the network of the different components and thus to not forget them (the information is also present in the item networks, which makes it possible to check these links at the end of the modeling).

When defining the elements necessary to insert data into tables, the modeler questioned whether a primitive can have one or more values (`focus in` and `focus out`) or that these two values are each set in a component. This implies that the link between primitives just mentioned is made in two ways, as needed. We noted that a reflection on this topic should be carried out to see whether the complexity brought by this flexibility is justified — which could be the case for certain indicators — or whether only one value can be assigned to this primitive. This thought led the modeler to make two possible changes:

1. having two items from this primitive, each with a unique value: then link each one into the component to create the table (and create these links);
2. having an item from this primitive with several values: link the item in question, then choose the value(s) concerned.

The modeler stated that the primitive dedicated to the indicator itself is clear, and that the (`Analysis`) part will depend on the technologies chosen.

Regarding the primitive used for modeling the visualization of the indicator, the second modeler questioned the chosen names, but found the primitive clear:

The item itself is super clear, it's just the naming system compared to the existing ones...

Finally, the modeler stated that the primitive (`Analytical Engine`) seemed less clear, because it is "a little more difficult to visualize without the context that might arise").

Regarding the last task of phase 3, the change supposedly requested by the client, the modeler was able to perform it without any issue.

To conclude, the second modeler stated that he would need a practical application and a real context for even more precise opinions, because it is with use and time that we will be able to find other improvements.

6.3 Discussion: summary of the feedbacks

In order to synthesize the results of this analysis, we propose to come back to the research questions asked.

How is the handling of the metamodel by modelers? The modelers seem to handle the metamodel without much difficulty. Some areas for improvement were suggested, some parts were less clear, but in general the primitives were understood and used correctly for a very first modeling of an indicator.

Are the primitives used as expected? Yes, they are used as intended. The order of use of the components is logical and sometimes with back and forth actions, which is expected, especially for first-time use.

What are the blocking points during this first usage? A visualization-related primitive was less clear to one of the modelers; for the other modeler, it was a primitive linked to the calendar of calculations. In the first case, this difficulty could be related to the fact that the visualizations were implemented with less detail at the time of the tests. In the second case, the modeler asserted that the difficulty was linked to the abstract project/need as opposed to a practical situation, in which it would probably be easier to perceive the use of the primitive.

Are there any improvements to be expected, which ones? Yes, the two main improvements are related to the naming system (see below) and the fact that some primitives are not explicitly linked.

Is the vocabulary used clear and understandable to a modeler? Most of the vocabulary seems to be understood without problem, especially the one relating to the field of LA. However, some suggested changes are: 1/ add the term (*Visualization*) to (`uiWidget`), or do not consider this primitive as a widget, because it is too different from the existing ones; 2/ In (`Trace`), change the term of (`fromModel`) to (`callModel`) which already exists, for consistency.

What is the perceived usability of the metamodel by modelers? Perceived usability (via the questionnaire on phase 2) does not have very high scores, but during interviews it was possible to understand that this was due to the fact that it is a metamodel, which is inherently perceived as complex even by experts used to manipulate them. The fact that the modelers were able to model a first indicator autonomously, and the interviews, allow us to put this initially perceived complexity into perspective.

How are the proposed primitives used to encourage maintainability, reuse, and customization of indicators? This aspect was analyzed through the request to change an aspect of the indicator at the end of phase 3. The modelers were able to understand and perform the requested task, creating a new indicator by changing a few aspects of the metamodel. In addition, the maintainability, reuse, and customization of the use of an indicator via the metamodel seem to be well received considering the latest responses to the questionnaire (and certainly also the modeling experience).

7 Conclusion

In this work, we proposed and evaluated a learning analytics metamodel that uses a model-driven approach within digital publishing chains based on the same MDE approach. The metamodel aims at being sufficiently abstract to allow the implementation of the vast majority of learning analytics indicators, and includes the possibility to enrich them natively with the prior knowledge of documents' semantics and structure.

Results show that modelers seem to take in hand the metamodel without significant difficulties: they used the primitives in a logical order, sometimes with back and forth actions, which is expected for a metamodel and especially for a first use.

Most of the vocabulary seems to be understood and some suggestions for changes have been made, which are easy to fix. During the interviews, it was possible to understand that usability was rated low given the inherent complexity of use. It should be noted that the modelers were able to model a first indicator after being shown only one example and that especially some aspects such as maintainability, reuse, and customization of an indicator via the metamodel seemed to be well perceived. This could be verified via the request to change an aspect of an indicator (phase 3), where the modelers could change only part of a primitive to adapt it according to new needs or to create a new one from the primitives modeled previously.

Although this work is instanciated in the context of a particular model-driven tool, we believe the work presented here is generic enough to be replicated in a similarly designed environment. Further work would involve a deeper analysis of the modelers' work on a real task once all the elements will be fully implemented into the system. Although a larger sample size would be appreciated, the very

specialized nature of the modelers' work does not make it realistic to imagine having enough participants for a quantitative analysis.

References

1. Agudo-Peregrina, Á.F., Iglesias-Pradas, S., Conde-González, M.Á., Hernández-García, Á.: Can we predict success from log data in VLEs? Classification of interactions for learning analytics and their relation with performance in VLE-supported F2F and online learning. Computers in Human Behavior **31**(1), 542–550 (feb 2014). https://doi.org/10.1016/j.chb.2013.05.031
2. Arribe, T., Crozat, S., Bachimont, B., Spinelli, S.: Chaînes éditoriales numériques : allier efficacité et variabilité grâce à des primitives documentaires. In: Actes du colloque CIDE. pp. 1–12. Tunis, Tunisie (2012)
3. Bachimont, B., Crozat, S.: Instrumentation numérique des documents : pour une séparation fonds/forme. Revue I3 - Information Interaction Intelligence **4**(1), 95 (2004)
4. Canellas, C., Bouchet, F., Arribe, T., Luengo, V.: Towards Learning Analytics Metamodels in a Context of Publishing Chains. In: Proceedings of the 13th International Conference on Computer Supported Education. pp. 45–54. SCITEPRESS - Science and Technology Publications (apr 2021). https://doi.org/10.5220/0010402900450054
5. Choquet, C., Iksal, S.: Usage Tracking Language: a Meta-Language for Modelling Tracks in TEL Systems. International Conference on Software and Data Technologies (ICSOFT) pp. 133–138 (2006). https://doi.org/10.5220/0001312701330138
6. Choquet, C., Iksal, S.: Modélisation et construction de traces d'utilisation d'une activité d'apprentissage : une approche langage pour la réingénierie d'un EIAH. Sciences et Technologies de l'Information et de la Communication pour l'Éducation et la Formation **14**(1), 419–456 (2007). https://doi.org/10.3406/stice.2007.968
7. Combemale, B.: Ingénierie Dirigée par les Modèles (IDM) – État de l'art (2008)
8. Cordier, A., Lefevre, M., Champin, P.A., Georgeon, O., Mille, A.: Trace-based reasoning - Modeling interaction traces for reasoning on experiences. FLAIRS 2013 - Proceedings of the 26th International Florida Artificial Intelligence Research Society Conference pp. 363–368 (2013)
9. Crozat, S.: Scenari, la chaîne éditoriale libre. Eyrolles (2007)
10. Guillaume, D., Crozat, S., Rivet, L., Majada, M., Hennequin, X.: Chaînes éditoriales numériques (2015)
11. Hutchinson, J., Rouncefield, M., Whittle, J.: Model-driven engineering practices in industry. In: Proceedings - International Conference on Software Engineering. pp. 633–640. ACM Press, New York, New York, USA (2011). https://doi.org/10.1145/1985793.1985882
12. Ifenthaler, D.: Are Higher Education Institutions Prepared for Learning Analytics? TechTrends **61**(4), 366–371 (12 2016). https://doi.org/10.1007/S11528-016-0154-0
13. Jézéquel, J.M., Combemale, B., Vojtisek, D.: Ingénierie dirigée par les modèles - Des concepts à la pratique. Ellipses, Paris (2012)
14. Kitto, K., Whitmer, J., Silvers, A.E., Webb, M.: Creating Data for Learning Analytics Ecosystems. SOLAR Position Paper (September), 1–43 (2020)
15. Patton, M.Q.: Qualitative Evaluation and Research Methods. SAGE Publications, 2nd edn. (1990)

16. Siemens, G.: Learning and Academic Analytics (2011), https://www.learninganalytics.net/uncategorized/learning-and-academic-analytics/
17. Wise, A.F., Vytasek, J.: Learning Analytics Implementation Design. In: Lang, C., Siemens, G., Wise, A., Gašević, D. (eds.) Handbook of Learning Analytics, chap. 13, pp. 151–160. SoLAR, first edn. (2017). https://doi.org/10.18608/hla17.013