A Rule-Based Indicator Definition Tool for Personalized Learning Analytics

Arham Muslim
Learning Technologies (Informatik 9)
RWTH Aachen, Germany
muslim@cil.rwth-aachen.de

Mohamed Amine Chatti
Learning Technologies (Informatik 9)
RWTH Aachen, Germany
chatti@informatik.rwth-aachen.de

Tanmaya Mahapatra
Learning Technologies (Informatik 9)
RWTH Aachen, Germany
tanmaya.mahapatra@rwth-aachen.de

Ulrik Schroeder
Learning Technologies (Informatik 9)
RWTH Aachen, Germany
schroeder@cil.rwth-aachen.de

ABSTRACT

In the last few years, there has been a growing interest in learning analytics (LA) in technology-enhanced learning (TEL). Generally, LA deals with the development of methods that harness educational data sets to support the learning process. Recently, the concept of open learning analytics (OLA) has received a great deal of attention from LA community, due to the growing demand for self-organized, networked, and lifelong learning opportunities. A key challenge in OLA is to follow a personalized and goal-oriented LA model that tailors the LA task to the needs and goals of multiple stakeholders. Current implementations of LA rely on a predefined set of questions and indicators. There is, however, a need to adopt a personalized LA approach that engages end users in the indicator definition process by supporting them in setting goals, posing questions, and self-defining the indicators that help them achieve their goals. In this paper, we address the challenge of personalized LA and present the conceptual, design, and implementation details of a rule-based indicator definition tool to support flexible definition and dynamic generation of indicators to meet the needs of different stakeholders with diverse goals and questions in the LA exercise.

CCS Concepts

•**Information systems** → **Information systems applications;** *Data analytics;* •**Applied computing** → **E-learning;** •**Human-centered computing** → *Visual analytics;*

LAK '16, April 25-29, 2016, Edinburgh, United Kingdom
© 2016 ACM. ISBN 978-1-4503-4190-5/16/04...$15.00
DOI: http://dx.doi.org/10.1145/2883851.2883921

Keywords

Learning analytics, open learning analytics, personalized learning analytics, indicator

1. INTRODUCTION

Learning analytics (LA) represent the application of "big data" and analytics in education [13]. LA has been defined at the first international conference on learning analytics and knowledge (LAK11) as "the measurement, collection, analysis and reporting of data about learners and their contexts, for purposes of understanding and optimizing learning and the environments in which it occurs" [13]. Chatti et al. presented a reference model for LA and provided a systematic overview based on four dimensions: *What* kind of data does the system gather, manage, and use for the analysis, *Who* is targeted by the analysis, *Why* does the system analyze the collected data and *How* does the system perform the analysis of the collected data [6, 5].

A particularly rich area for future research is open learning analytics (OLA) [13]. In general, OLA encompasses different stakeholders associated through a common interest in LA but with diverse needs and objectives, a wide range of data coming from various learning environments and contexts, as well as multiple infrastructures and methods that enable to draw value from data in order to gain insight into learning processes. The various objectives in OLA (e.g. monitoring, analysis, prediction, intervention, tutoring, mentoring, assessment, feedback, adaptation, personalization, recommendation, awareness, reflection) need a tailored set of questions and indicators to serve different stakeholders with diverse goals [5]. An indicator can be described as specific calculators with corresponding visualizations, tied to a specific question [9]. Current implementations of LA rely on a predefined set of indicators. This is, however, not helpful in the case of OLA where the set of required indicators is unpredictable. Thus, it is crucial to follow a personalized and goal-oriented LA model that supports end users in setting goals, posing questions, and self-defining the indicators that help them achieve their goals. In this paper, we address the challenge of personalized LA and present the conceptual, design, and implementation details of a rule-based indicator definition tool to support flexible definition of indicators to meet the needs of different stakeholders with diverse goals.

The remainder of the paper is structured as follows: In Section 2, we introduce the concept of personalized LA and its associated challenges. Followed by the review of related work in Section 3. In Section 4, the design and implementation of a Rule-Based Indicator Definition Tool is presented. The results of evaluation are discussed in section 5. Finally, Section 6 gives the concluding remarks and the future work of this paper.

2. PERSONALIZED LEARNING ANALYTICS

The concept of Open Learning Analytics (OLA) was introduced in 2011 by a group of leading thinkers on LA in an initial vision paper published by the Society for Learning Analytics Research (SoLAR) [13]. A first summit was then held in Indianapolis, Indiana in March 2014 to promote networking and collaborative research [1]. Building on the first summit, the Learning Analytics Community Exchange (LACE) project organized in December 2014 the Open Learning Analytics Network Summit Europe to develop a shared European perspective on the concept of an open learning analytics framework [7].

So far, from the initial vision paper through the last summit, the development of the concept of open learning analytics was restricted to a discussion on the need for open source software, open standards, and open APIs to address the interoperability challenge in this field as well as how important tackling the ethical and privacy issues is becoming for a wide deployment of LA. Recently, Chatti et al. point out that OLA refers to an ongoing analytics process that encompasses diversity at all four dimensions of their LA reference model [5]:

– What? It accommodates the considerable variety in learning data and contexts. This includes data coming from traditional education settings (e.g. LMS) and from more open-ended and less formal learning settings (e.g. PLEs, MOOCs).

– Who? It serves different stakeholders with very diverse interests and needs.

– Why? It meets different objectives according to the particular point of view of the different stakeholders.

– How? It leverages a plethora of statistical, visual, and computational tools, methods, and methodologies to manage large datasets and process them into metrics which can be used to understand and optimize learning and the environments in which it occurs.

A key challenge in OLA is to serve different stakeholders with diverse goals and questions. It is essential to see end users as the central part of the LA practice. In fact, user involvement is the key to a wider user acceptance, which is required if LA tools are to serve the intended objective of improving learning and teaching. Thus, it is important to follow a personalized and goal-oriented LA model that tailors the LA task to the needs and goals of different stakeholders. There is a need to adopt a user-in-the-loop LA approach that engages end users in a continuous inquiry-based LA process to define the right Goal / Question / Indicator (GQI) triple. Giving users the opportunity to set their goal, pose questions, and specify the indicator to be applied is a crucial step to achieve personalized LA results. This would also make the LA process more transparent, enabling users to see what kind of data is being used and for which purpose.

From a technical perspective, most LA tools currently available are designed in such a way that a user can select from a predefined catalogue of static questions and indicators with predetermined data source, analytics method, and visualization type to be displayed in a dashboard. If a user needs a new Indicator which is not available in the catalogue, she has to request it from the tool's developer. To achieve personalized LA, there is a need for an indicator definition tool that enables end users to dynamically define new questions and indicators that meet their goals in an efficient and effective way. This tool should provide a user-friendly interface that supports an interactive, exploratory, and real-time user experience to enable a flexible data exploration and visualization manipulation.

3. RELATED WORK

In the LA literature, there is a wide range of tools that use dashboard to graphically show different indicators. These tools, however, use predefined set of indicators and do not allow end users to define new indicators. To the best of our knowledge, there is no research which has been conducted to achieve personalized learning analytics that enables end users to dynamically define new indicators based on their goals. However, related to our research, there are tools available which support users in representing and executing analysis workflows. For instance, Göhnert et al. propose a web-based analytics workbench to support researchers of social networks in their analytic process. The workbench offers a visual representation of the data flow, using the pipes-and-filters design pattern. Following a multi-agent approach, the workbench also allows an integration of additional analysis components [10]. Similarly, the Konstanz Information Miner (KNIME) offers a modular environment, which provides a graphical workbench for visual assembly and interactive execution of data pipelines. Further, KNIME enables integration of new algorithms as well as data manipulation or visualization methods in the form of new modules or nodes [3]. The two data pipelining environments presented above can be used to model workflows which visualize every single step of the analysis process as one node and make the flow of data explicit by connecting those nodes. These environments offer users powerful mechanisms for visual representation of multi-step analysis workflows as well as data manipulation. However, such workflow and pipelining tools have two major limitations when used in the context of personalized learning analytics. First, these tools are data-driven rather than goal-oriented and thus cannot be used to support LA users in setting goals, posing questions, defining indicators, and relating them to questions. Second, these tools are designed to support specialists in the respective domains. In KNIME, for instance, a user needs to interact with a large variety of nodes, from which she can select data preprocessing steps, data mining and machine learning algorithms along with their parameters, as well as visualization tools, and connect them in the right way to generate a valid data workflow. The complexity of these tools might hinder non-technical user to adopt them. It is widely recognized in the LA literature that LA tools should be designed in a way that can help learners, teachers, and institutions to achieve their analytics objectives without the need for having an extensive knowledge of the techniques underlying these tools. In particular, educational data mining tools should be designed for non-expert users in data mining.

In the next section, we present the Rule-based Indicator Definition Tool (RIDT) which offers an easy to use and effective environment to define the Goal / Question / Indicator (GQI) triple and leverages rule engines to reduce the complexity of the indicator definition process.

4. RULE-BASED INDICATOR DEFINITION TOOL

The aim of RIDT is to achieve personalized and goal-oriented LA by enabling end user to define their own set of goals, question and indicators. In the following sections, we present the design and implementation of RIDT through detailed discussion of possible user scenario, systems requirements, conceptual architecture, implementation details, and evaluation.

4.1 User Scenario

Ayesha is a lecturer at RWTH University where she uses the university learning management system L^{2}P to administer her courses. She uses RIDT to monitor her eLearning class using various predefined indicators such as presence of students in class, most viewed/downloaded documents, and the progress of students in assignments. Recently, Ayesha came up with the requirement to see which students are active in her new Web Technologies class. She looked in the list of available indicators but did not find all the indicators she was looking for which can define the activeness of students in a class. She uses the RIDT interface to define a new analytics question "how active are my students in the Web Technologies class". She reuses the existing indicator "presence of students in class" and updates the filter to get data related to the Web Technologies class.

After that she starts creating a new indicator named "post rate in discussion forums". She chooses the indicator type (statistics), selects L^{2}P as a data source, selects discussion forum as a category, applies a filter to get data related to the Web Technologies class, and chooses to visualize the results as a bar chart. Similarly, she creates another indicator named "post rate in wikis". Finally, she creates a new composite indicator "collaboration rate in class" by combining the two indicators "post rate in discussion forums" and "post rate in wikis" and choosing stacked bar chart to visualize the combined results. The newly created question and associated indicators are then added to the list of available questions and indicators for future use by other users.

4.2 Requirements

Developing a tool to allow dynamic definition of new indicators and managing them is a challenging task. The main requirements for RIDT are described below.

4.2.1 Personalization

It is necessary to provide users full control in the whole indicator definition process which is key to a wider user acceptance of the tool.

4.2.2 Transparency

The whole indicators definition process should be transparent for users to see what kind of data is being used and for which purpose. This is vital to drive forward the acceptance of the tool.

4.2.3 Usability

RIDT should provide simple, intuitive, and consistent User Interface (UI) to easily guide users through the indicator definition process.

4.2.4 Reusability

It is necessary to design an architecture that supports the reuse of existing questions and indicators by other users as well as the reuse of existing indicators in the generation of new ones.

4.2.5 Modularity, Flexibility and Extensibility

RIDT should have a modular and flexible architecture to support faster, cheaper, and less disruptive adaptability of new indicators types due to ever changing requirements of the users.

4.2.6 Performance and Scalability

Scalability and performance should be taken into consideration when designing the architecture of RIDT in order to support growing number of end users and indicators of different types.

4.3 Conceptual Approach

In the following sections, we discuss the design of RIDT with a focus on the underlying data models, the abstract architecture, and a system workflow.

4.3.1 Data Models

Different data model specifications have been introduced in the LA literature [8]. RIDT uses the data model called Learning Context Data Model (LCDM) suggested by Thüs et al. [14] in the frame of the Learning Context Project[1]. To note, that the concepts discussed in this paper are valid if other data models are used.

As compared to other candidate data models for LA, such as Experience API (xAPI), IMS Caliper, Activity Streams Contextualized Attention Metadata (CAM), Learning Registry Paradata, and NSDL Paradata, LCDM is a compromise between completeness, flexibility, and simplicity. It is a user-centric, modular, and easy to understand data model that holds additional semantic information about the context in which an event has been generated [14, 12]. As illustrated in Figure 1, LCDM associates users with an arbitrary amount of events. The elements of each event are described as follows:

- *Source:* This model gathers data from different applications. To preserve this context information, the name of the application that generated this event is stored as source, such as, L^{2}P, Moodle, etc.

- *Platform:* Since the centralized data model hold data from different kinds of devices, this attribute specifies the platform from which the event originated, such as mobile device, web-based, or stationary device.

[1]http://www.learning-context.de/

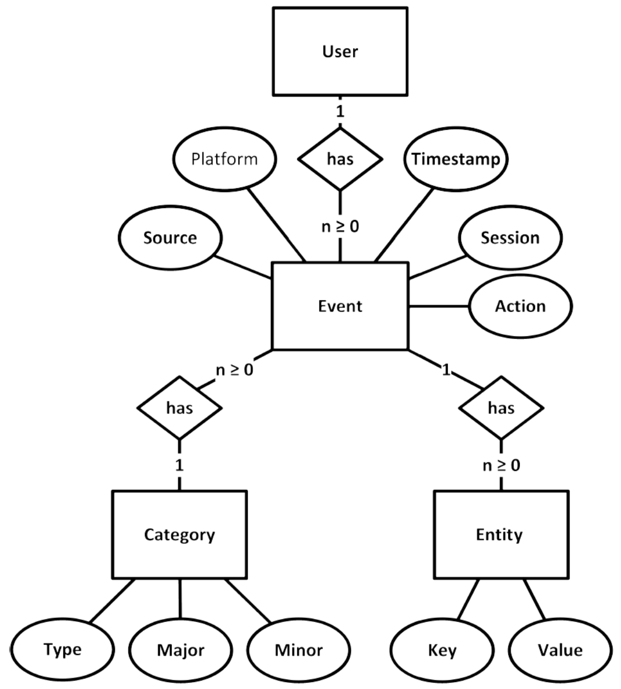


Figure 1: Learning Context Data Model (LCDM) Data Model (adapted from [14])

- *Timestamp:* This attribute indicates when the event has been created to preserve the sequence of events.
- *Session:* This attribute is used to group different events together.
- *Action:* This attribute defines what kind of action happened in the event, such as, update, delete, start, end, etc.
- *Category:* This attribute comprises of three parameters, namely type, major and minor. Type is an optional parameter which defines if the event is of private, professional, or academic nature. Major and minor describe the broader semantic information of the event. Major defines the group (environmental, bio or activity) and minor gives the kind of the event (Wiki, Discussion forum, Calendar, etc.). This is a predefined list but it can be extended based on new requirements.
- *Entity:* An event can be described by an arbitrary amount of key-value pairs. This could e.g. be the longitude and latitude of a position with their respective values or this could be the title and the resource of a learning material.

Besides using LCDM to represent learning events, RIDT uses an additional data model to manage and store questions and indicators, as shown in entity relationship diagram in Figure 2. Every question can have multiple indicators associated with it and an indicator can be associated with multiple questions. Different properties are associated with the question and indicator entities. For example, the *Query* property stores the data query for the Indicator to fetch data from the database every time the indicator is executed. All the properties of the indicator defined by the user are stored

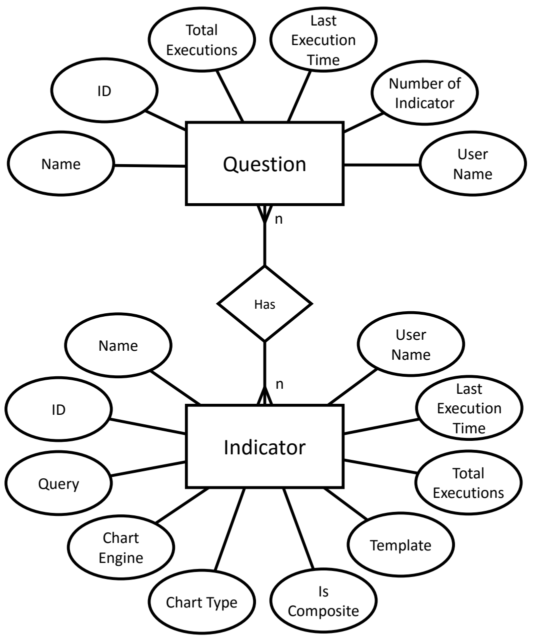


Figure 2: Question & Indicator Entity Relationship Diagram

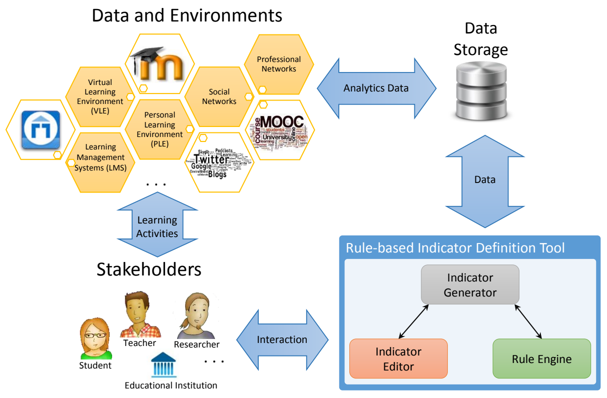


Figure 3: RIDT Conceptual Architecture

in the *Template* property in a JSON format to be reused for the generation of a new indicator based on an existing one.

4.3.2 Abstract Architecture

Figure 3 shows the abstract architecture of RIDT. It is composed of three main components: *Indicator Editor*, *Rule Engine*, and *Indicator Generator*. The *Indicator Editor* provides a user-friendly interactive interface to set the LA goal, formulate questions, and define indicators associated with those questions. The *Rule Engine* is responsible for managing indicator types and their associated queries. The *Indicator Generator* is responsible for the generation of new indicators by executing the queries and rendering the indicator.

4.3.3 System Workflow

RIDT follows the GQI approach to allow users to define new indicators. The user starts the indicator definition process by interacting with the *Indicator Editor* and defining the LA goal such as monitoring, analysis, prediction, intervention, feedback, adaptation, recommendation, awareness, reflection, etc. Based on the defined goal, the user asks the LA question and associates indicators to answer this question. Some questions can be answered using a single indicator whereas complex questions would require multiple indicators to fully answer them. For instance, the question "how active are students in class" can have "presence of students in class", "post rate in discussion forums", and "post rate in wikis" as associated indicators to answer it properly.

To define a new indicator, the *Indicator Editor* communicates through *Indicator Generator* with the *Rule Engine* to obtain the list of possible indicator types and with the data storage to get possible data objects based on the data model used in RIDT, as discussed in Section 4.3.1. Taking the example of the indicator "post rate in discussion forums in L^{2}P courseroom Web Technologies", the user first selects the indicator type "number of X in Y". Then she specifies the data *Source* as "L^{2}P", the *Action* as "post", the *Category* as "discussion forum". Additionally, she applies a filter on the coursename *Entity* as "Web Technologies". The *Indicator Generator* communicates with the *Rule Engine* to get the query template related to the selected indicator type and generates the query based on the specified data objects and filters. In our example, in SQL terms, the generated query "SELECT COUNT(*) FROM table_event WHERE Source='L2P' AND Action='post' AND Category='discussion forum' AND Entity.coursename='Web Technologies';" will be associated with the indicator "post rate in discussion forums in L^{2}P courseroom Web Technologies". After defining the indicator properties and filters, the user selects the appropriate visualization technique to visualize the indicator such as bar chart using D3.js.

The user can create new indicators by following the same flow, load an existing indicator as a template and modify its properties, or define a new composite indicator by combining two or more existing ones. Finally, the user can approve the question and its associated indicators which are then stored in the question / indicator database along with the related query and visualization for each indicator.

4.4 Implementation

RIDT is a Web application based on the Java Spring Framework[2] that follows the Model View Controller (MVC) design pattern [11]. It is developed using the Test-Driven Development (TDD) approach [2]. The application makes extensive use of Web technologies, such as Servlets, JavaServer Pages (JSP), JavaBeans at the server side and jQuery[3], jQuery UI[4], AJAX[5], and noty[6] at the client side. In the following sections, we discuss the technical and implementation details of the different components of RIDT, as depicted in Figure 4.

4.4.1 Indicator Editor

The *Indicator Editor* is the collection of MVC Views providing an interactive user interface (UI) for the RIDT. As

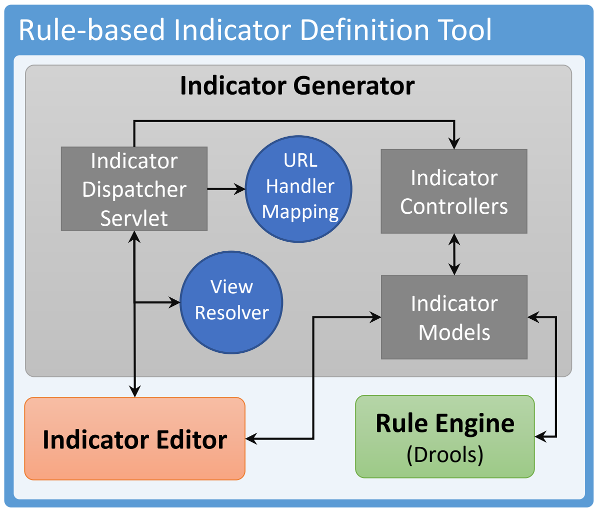


Figure 4: RIDT Technical Architecture

shown in Figure 5, the main UI consists of four main sections:

Question Information

In this section the user starts the indicator definition process by entering the LA question. After that, the user can associate a set of possible indicators with the question by loading an existing indicator and modifying its properties or defining a new indicator. For every indicator, the user has to enter a unique name and select a type. Furthermore, in this section the user can visualize the question which opens a new UI 'Question Visualization' to render all the associated indicators in a dashboard form, as shown in Figure 6. From this UI, the user has the possibility to combine two or more indicators and generate a new composite indicator which is associated with the question as an independent indicator. Composite indicators have their own name and visualization and they continue to function even if the source indicators are deleted. In Figure 6, the indicator "collaboration rate in class" is a composite indicator created by combining the two indicators "post rate in discussion forums" and "post rate in wikis". After defining all the associated indicators, the question along with its indicators can be saved in the question / indicator database.

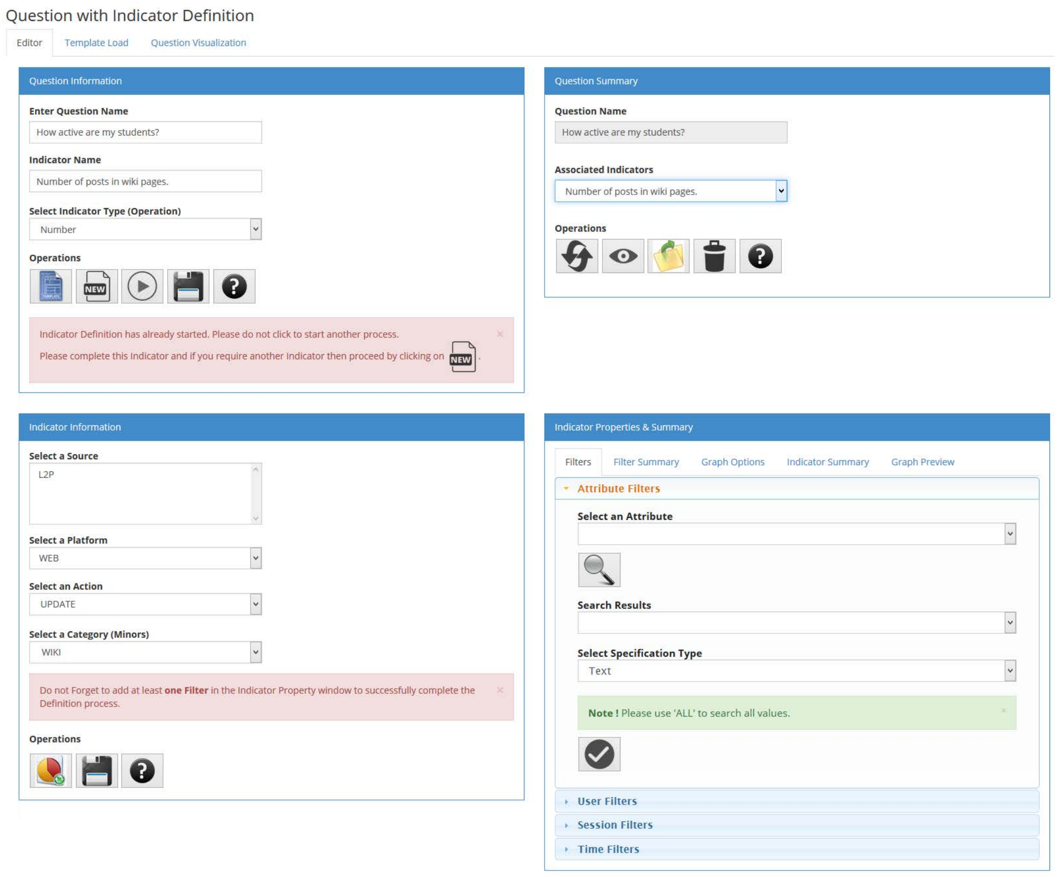
Indicator Information

In this section the user can specify basic filtering parameters of the indicator, such as from which *Source* data should be used (L^{2}P, Moodle, etc.), data related to which *Action* type should be used (add, update, delete, post, begin, end, etc.) and which *Category* should be used (Wiki, Discussion forum, Assignments, etc.). These parameters represent the elements of the LCDM data model, as discussed in Section 4.3.1.

Indicator Properties & and Summary

In this section the user can refine the data fetched from LCDM event store based on the basic filtering parameters specified in the 'Indicator Information' section by specifying additional filtering parameters of the indicator. The user

[2]http://projects.spring.io/spring-framework/
[3]https://jquery.com/
[4]https://jqueryui.com/
[5]http://www.w3schools.com/ajax/
[6]http://ned.im/noty/



Enter Question Name: How active are my students?
Indicator Name: Number of posts in wiki pages.
Select Indicator Type (Operation): Number

Indicator Definition has already started. Please do not click to start another process.
Please complete this Indicator and if you require another Indicator then proceed by clicking on NEW.

Question Name: How active are my students?
Associated Indicators: Number of posts in wiki pages.

Select a Source: L2P
Select a Platform: WEB
Select an Action: UPDATE
Select a Category (Minors): WIKI

Do not Forget to add at least **one Filter** in the Indicator Property window to successfully complete the Definition process.

Select Specification Type: Text

Note ! Please use 'ALL' to search all values.

Figure 5: RIDT Main User Interface

can add filters to specify *Entity* attributes, get data for a particular *User*, in a particular *Timestamp* range, for a specific *Session*. The summary of all applied filters can then be viewed and edited under the 'Filter Summary' tab. Under the 'Graph Options' tab the user can define properties related to the visualization of the indicator, such as graph type (e.g. bar chart, pie chart) and the visualization library to use for rendering the graph (e.g. D3.js, Google charts, JGraph). The 'Indicator Summary' tab shows a comprehensive summary of the selected properties for the current indicator. After all the properties are defined the indicator can be previewed under the 'Graph Preview' tab.

Question Summary

This section displays a detailed summary for the current question and its associated indicators that have already been defined. Furthermore, the user has the option of loading the indicators to edit their properties or deleting them before finalizing the question.

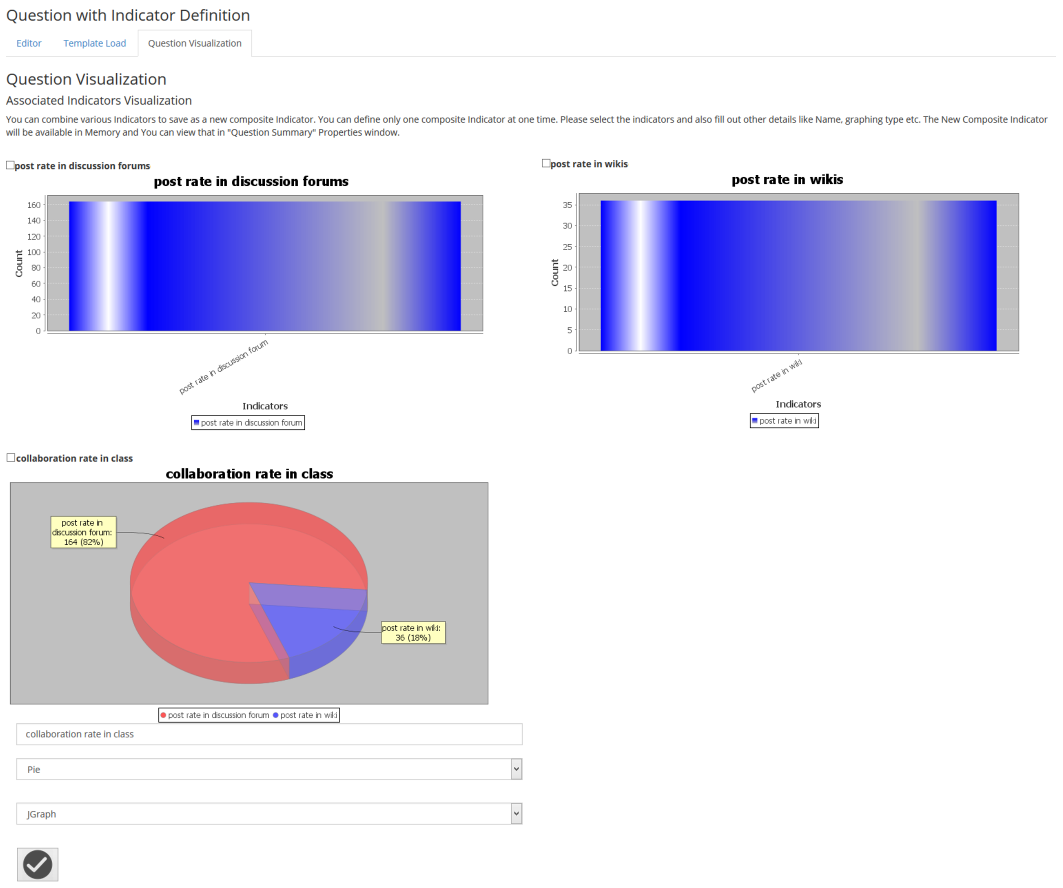
4.4.2 Rule Engine

This component is responsible for managing the rules for the different indicator types and their associated queries. In RIDT, Drools[7] is used as a rule engine which stores rules using MVFLEX Expression Language (MVEL)[8] in a file. Each rule in the file reflects a combination of selected additional filtering parameters and maps it to an appropriate query. Every Drools rule have three main sections, a 'rule' part which contains a phrase to serve as the human identifiable name of the rule, a 'when' part where the condition of the rule is specified and a 'then' part to describe the actions which are to be taken if the specified conditions are satisfied. Listing 1 shows a Drools rule for the scenario when there are no additional filters specified by the user while defining the indicator.

```
dialect "mvel"
rule "When no additional filters are present"
when
  EntitySpecification (
    userSpecifications.isEmpty() == true,
    timeSpecifications.isEmpty() == true,
    sessionSpecifications.isEmpty() == true,
    entityValues.size()== 1,
    entityValues.get(0).geteValues() == "ALL")

  $pFilter : ProcessUserFilters()
  $userParameter : EntitySpecification(
    $source : selectedSource,
    $platform : selectedPlatform,
    $action : selectedAction,
    $major : selectedMajor,
```

[7]http://www.drools.org/
[8]https://github.com/mvel


Question with Indicator Definition

Editor | Template Load | Question Visualization

Question Visualization

Associated Indicators Visualization

You can combine various Indicators to save as a new composite Indicator. You can define only one composite Indicator at one time. Please select the indicators and also fill out other details like Name, graphing type etc. The New Composite Indicator will be available in Memory and You can view that in "Question Summary" Properties window.

☐ post rate in discussion forums

☐ post rate in wikis

☐ collaboration rate in class

collaboration rate in class

Pie

JGraph

post rate in discussion forums
Count
Indicators
post rate in discussion forum
post rate in wikis
post rate in wiki
collaboration rate in class
post rate in discussion forum: 164 (82%)
post rate in wiki: 36 (18%)

Figure 6: RIDT Question Visualization User Interface

```
      $minor : selectedMinor,
      $type : selectedType,
      $key : entityValues,
      $pobject: persistenceObject,
      $filter : filteringType,
      $robject : retrievableObjects)
  then
    $userParameter.setHql("SELECT " + $robject
    + " FROM" + $pobject + " WHERE Action='"
    + $action + "' AND Source IN "
    + $pFilter.processSource($source, $filter)
    + " AND Platform = '"+ $platform
    + "' AND Category.Major = '" + $major
    + "' AND Category.Minor = '"+ $minor
    + "' AND Category.Type = '" + $type
    + "' AND Key = '" + $key.get(0).getKey() + "'");
  end
```

Listing 1: Default Drools rule when no additional filters has been added.

After the user has defined all the parameters of the indicator using *Indicator Editor*, the *Indicator Generator* initiates a unique Drools instance in which it inserts the rule file related to indicator type, the indicator filtering parameters entered by the user, and other necessary java objects to parse user data and generate the hibernate query. Every time the indicator is executed, the *Indicator Generator* uses this hibernate query to fetch the relevant data for visualizing the indicator.

The benefits of following a rule-based approach in RIDT are twofold. First, it is possible to replace LCDM with other data models (e.g. xAPI, IMS Caliper) by altering the 'when' and 'then' parts in the rule to accommodate the attributes of the new data model. Second, it is easy to use a different database technology (e.g. NoSQL) by defining appropriate actions in the 'then' part.

4.4.3 Indicator Generator

The *Indicator Generator* is the backbone of RIDT which acts as a mediator between different components in the tool. It is responsible for generating the data queries by communicating with the *Rule Engine*, executing the queries to get data for visualization, and finally rendering the indicator to be displayed on the *Indicator Editor*. The different components of the *Indicator Generator* are described below:

Indicator Dispatcher Servlet

The Indicator Dispatcher Servlet (IDS) is responsible for handling all incoming requests and returning the responses. Based on the request, the IDS consults the 'URL Handler Mapping' component to resolve which controller is appropriate for this request and dispatch the request to the returned 'Indicator Controller' for further processing.

Indicator Controllers

The Indicator Controllers are designed to handle user requests and invoke the suitable service methods based on the HTTP request type which can be either GET or POST. The service methods process the requested URL and return the logical name of the corresponding view to IDS. Controllers in RIDT can be broadly classified into two major categories: JSON-based controllers and view-based controllers. JSON-based controllers are basically server end points for the client side scripts to interact with the server. For example, when the user types a new name for an Indicator in *Indicator Editor*, the system must adhere to constraints like minimum number of characters in the name, check for duplicate names etc. In order to achieve this functionality, the client side scripts send a HTTP GET/POST request to these controllers with additional parameters. These controllers perform the necessary validation on receiving the request and return the appropriate response in JSON format which enables the system to perform in consistent, responsive and error-free manner. View-based controllers are mainly responsible for validating the data on the page and performing any additional processing in response to the incoming request for the specific page.

Indicator Models

Models are Plain Old Java Objects (POJOs) responsible for holding the core logic of RIDT. Models are divided in to three main categories:

- *Domain Objects:* These objects hold the necessary business logics which are invoked from the indicator controller based on certain criteria.
- *Data Mappers:* These objects are exclusively responsible for retrieval and storage of requisite data from data-source using Hibernate.
- *Services:* These are higher level objects responsible for interaction between the Domain Objects and the Data Mappers.

5. EVALUATION AND DISCUSSION

We conducted a user evaluation of RIDT to gauge the effectiveness and usability of the tool. We employed an evaluation approach based on the System Usability Scale (SUS) as a general usability evaluation as well as a custom questionnaire to measure the effectiveness of indicator definition in RIDT. During the evaluation process, the user was briefly introduced to the concept of personalized LA and the goals of RIDT. The tool was evaluated with two teachers, three research assistants, and seven students. The participants were asked to perform different tasks and subsequently fill out the questionnaire.

5.1 Participants' Background

The first part of the questionnaire captured the participants' backgrounds. Figure 7 shows the user background survey results. About 75% of the users were familiar with LA. All of them supported the idea to have an LA system at RWTH Aachen University which can be accessed by both students and teachers online. But only about 33% of the users had actually used an LA system with real time data. Also, 33% reported that they had prior experience with defining indicators in an LA system.

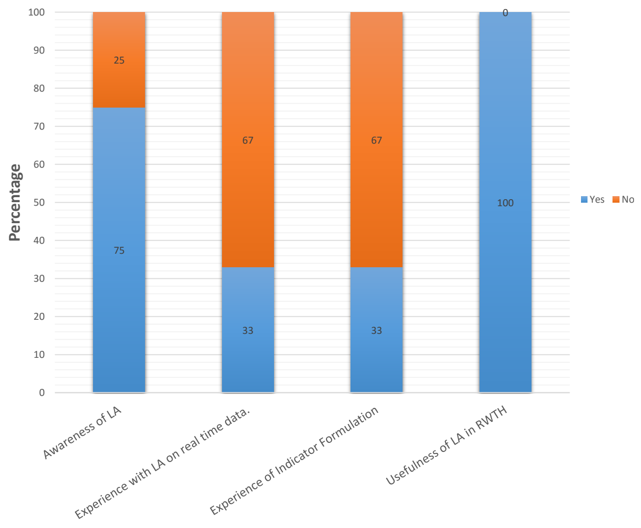


Figure 7: User Background Survey Result

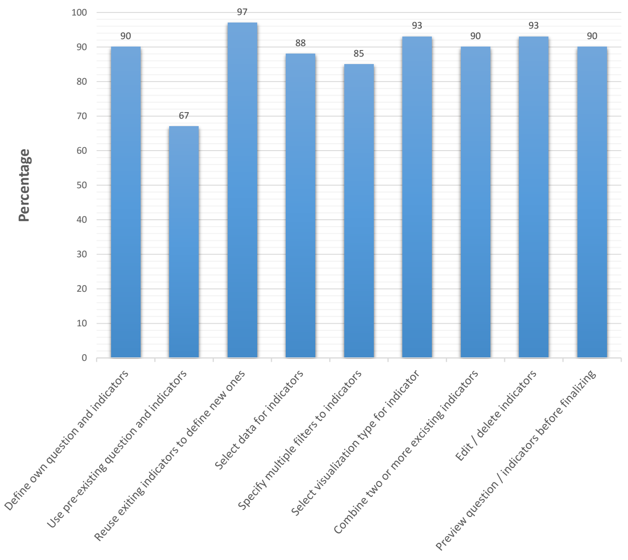


Figure 8: User Expectation Survey Result

5.2 Expectations

The second part of the questionnaire captured the expectations on the features that the users generally would like to have in an LA system. Figure 8 shows the user expectation survey results. About 90% of the users wanted to formulate their own questions and indicators as well as preview them. Over 90% of the users wanted to select different combination of data, add filters, select visualization types, modify existing indicators to create new ones, and delete existing indicators. About 88% of the users wanted to customize existing indicators available in the system. And, 67% of the users wanted to use pre-existing questions and indicators in the system.

5.3 Usefulness

The third part of the questionnaire captured the usefulness of the tool. Figure 9 shows the average percentage of user rating on system usefulness. There were ten questions

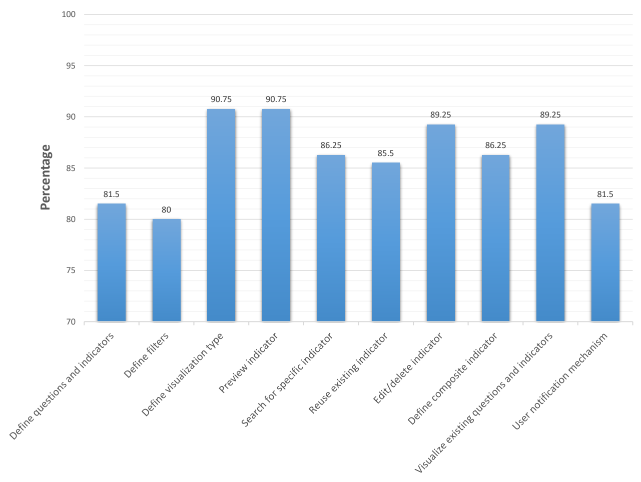


Figure 9: Average User Ratings on System Usefulness

in total which were answered after performing various tasks in RIDT. Overall, in terms of functionality, the response was positive as all the questions rated 80% or above. Top questions, "Define visualization type", "Preview indicator", "Edit/delete indictor" and "Visualize existing questions and indicator" which rated near 90% indicates that the most useful feature to the users was to trying out different visualization techniques and find best visualizations for their indicators. However, some participants suggested to introduce catalog for easy and fast addition of default filters as well as improve the notification mechanism.

5.4 Usability

The fourth part of the questionnaire dealt with the usability of the tool based on the System Usability Scale (SUS) which is a simple, ten-item attitude Likert scale giving a global view of subjective assessments of usability [4]. The questions are designed to capture the intuitiveness, simplicity, feedback, responsiveness, efficiency of the UI, user's comfort level in using the UI, and the steepness of the learning curve which a user must go through to successfully use the UI. Based on the results of the questionnaire, the usability scale (SUS score) of RIDT is calculated to be approximately 53, which is below the standard average SUS score of 68. In general, the respondents found the system intuitive, easy to use, and easy to learn. However, they expressed their concerns with inconsistencies in the UI and the lack of feedback and responsiveness of the tool. The majority of the evaluators wanted a more effective way of on-screen guidance during the indicator definition process.

5.5 Discussion of Results

In general, the evaluation results demonstrate that the majority of the users liked and appreciated the concept on which RIDT has been developed, namely to define own indicators in an effective and efficient manner. They liked the features offered by the tool like formulation of questions and indicators, flexibility of visualization type selection, previewing capabilities, and aggregation of simple indicators to build composite ones. However, the majority of the users suggested that the UI should be more consistent and responsive. It is obvious from the evaluation results that future work involving improvements in UI in terms of feedback is necessary through a more effective user notification mechanism which would guide the user through the indicator definition process.

6. CONCLUSION AND FUTURE WORK

In the last few years, there has been an increasing interest in learning analytics (LA) which calls for a movement from data to analysis to action to learning. In this paper, we introduced the concept of personalized learning analytics that engages end users in a continuous inquiry-based LA process, by supporting them in interacting with the LA tool to set goals, pose questions, and self-define the indicators that help them achieve their goals. We presented the theoretical, design, implementation, and evaluation details of a rule-based indicator definition tool (RIDT) to support flexible definition and dynamic generation of indicators. RIDT leverages rule engines to reduce the complexity of the indicator definition process. Users can use RIDT to easily define new indicators by exploring learning event data, selecting the appropriate dataset for the indicators, applying different levels of filters, and specifying which visualization technique should be used to render the indicators. The evaluation results are promising and show that RIDT has the potential to support a personalized learning analytics experience. Future work includes the improvement of the RIDT user interface mainly in terms of consistency and responsiveness, the implementation of new indicator types, and the integration of more visualization techniques.

7. REFERENCES

[1] S. Alexander et al. OLA press release (2014). Retrieved from http://solaresearch.org/initiatives/ola/.

[2] D. Astels. *Test driven development: A practical guide.* Prentice Hall Professional Technical Reference, 2003.

[3] M. R. Berthold, N. Cebron, F. Dill, T. R. Gabriel, T. Kötter, T. Meinl, P. Ohl, K. Thiel, and B. Wiswedel. Knime-the konstanz information miner: version 2.0 and beyond. *AcM SIGKDD explorations Newsletter*, 11(1):26–31, 2009.

[4] J. Brooke. Sus-a quick and dirty usability scale. *Usability evaluation in industry*, 189(194):4–7, 1996.

[5] A. M. Chatti, V. Lukarov, H. Thüs, A. Muslim, F. A. M. Yousef, U. Wahid, C. Greven, A. Chakrabarti, and U. Schroeder. Learning analytics: Challenges and future research directions. *eleed*, 10(1), 2014.

[6] M. A. Chatti, A. L. Dyckhoff, U. Schroeder, and H. Thüs. A reference model for learning analytics. *International Journal of Technology Enhanced Learning*, 4(5-6):318–331, 2012.

[7] A. Cooper. Open Learning Analytics Network - Summit Europe (2014). Retrieved from http://www.laceproject.eu/open-learning-analytics-network-summit-europe-2014/.

[8] A. Cooper. Standards and Specifications - Quick Reference Guide (2014). Retrieved from http://www.laceproject.eu/dpc/standards-specifications-quick-reference-guide/.

[9] A. L. Dyckhoff, D. Zielke, M. Bültmann, M. A. Chatti, and U. Schroeder. Design and implementation

of a learning analytics toolkit for teachers. *Journal of Educational Technology & Society*, 15(3):58–76, 2012.

[10] T. Göhnert, A. Harrer, T. Hecking, and H. U. Hoppe. A workbench to construct and re-use network analysis workflows: concept, implementation, and example case. In *Proceedings of the 2013 IEEE/ACM international conference on advances in social networks analysis and mining*, pages 1464–1466. ACM, 2013.

[11] A. Leff and J. T. Rayfield. Web-application development using the model/view/controller design pattern. In *Enterprise Distributed Object Computing Conference, 2001. EDOC'01. Proceedings. Fifth IEEE International*, pages 118–127. IEEE, 2001.

[12] V. Lukarov, M. A. Chatti, H. Thüs, F. S. Kia, A. Muslim, C. Greven, and U. Schroeder. Data models in learning analytics. In *DeLFI Workshops*, pages 88–95. Citeseer, 2014.

[13] G. Siemens, D. Gasevic, C. Haythornthwaite, S. Dawson, S. Buckingham Shum, R. Ferguson, E. Duval, K. Verbert, and R. S. d Baker. Open learning analytics: an integrated & modularized platform, 2011.

[14] H. Thüs, M. A. Chatti, C. Greven, and U. Schroeder. Kontexterfassung,-modellierung und-auswertung in lernumgebungen. In *DeLFI 2014-Die 12. e-Learning Fachtagung Informatik*, pages 157–162. Gesellschaft für Informatik, 2014.