Framework for Personalised Online Education based on Learning Analytics through the use of Domain-Specific Modelling and Data Analytics

Sofia Meacham
Computer and Informatics Department
Bournemouth University
Poole, UK
smeacham@bournemouth.ac.uk

Detlef Nauck
Business Modelling and Operational
Transformation Research
British Telecommunications, Adastral
Park
Ipswich, UK
detlef.nauck@bt.com

Han Zhao
Computer and Informatics Department
Bournemouth University
Poole, UK
s5086079@bournemouth.ac.uk

Abstract—**This paper contains a framework for the design and development of adaptive Virtual Learning Environments (VLEs) to assist educators of all disciplines with configuring and creating adaptive VLEs tailored to their needs. The proposed work is performed in three stages: In the first stage, development of an adaptive VLE that collects learning analytics and enables the educator to parametrize (turn on/off) their collection. The output of these analytics gets stored for further processing. In the second stage, data analysis and processing has been performed for the collected information. In the third stage, the results have been used as an input to adaptive VLE to enable informed personalisation of the student learning path and other adaptations. In this paper, we have proposed the combined use of two different environments for the different stages to achieve the most from their specialisation. For the first and the third stage, the *MPS Jetbrains* environment for the development of a domain-specific language (DSL) for adaptive VLEs was utilised. This development environment assists the creation of a new DSL that will enable educators to focus on the domain aspects and configure their adaptive VLE implementation to their needs. For the second stage of data analysis, the *weka* library was used to process the data, apply a range of classification algorithms and produce/store results that can then be used as an input to the adaptive VLE DSL. Overall, the proposed framework-system is anticipated to isolate the domain problem from the corresponding implementation details of web development and data analysis and give the adaptive VLE developer a seamless environment to experiment with a very quick turn-around time with ideas in their domain. More automation and integration between the VLE and the data science algorithms utilised for learning analytics data are part of our future plans towards the greater vision of more autonomic and personalised VLEs.**

Keywords—eLearning, adaptive VLE, DSL, classification algorithms, weka library

I. Introduction

research for several application domains. The idea of huge data collection, data processing and adaptations/decision making based on the above is the direction in design and development of the systems of the future. Applications ranging from the obvious category of smart cities and Internet of Things to business applications such as big-data to support decision making and intelligent business processes, require efficient data collection, processing, and adaptations.

Online education is no exception. A major ingredient of modern education is Virtual Learning environments (VLEs) and their usage in modern institutions ranges from blended-courses to fully online courses such as Massive Open Online Courses (MOOCs). A huge amount of information is collected through these systems that could be used to inform the educators' decision making and therefore provide improved student experience. Learning analytics has emerged as an area of data analytics in this case and adaptivity is increasingly being added to VLEs.

In this paper, we have developed a framework to enable educators from several disciplines to design, develop and use adaptive VLEs utilising the power that learning analytics can offer. First, a new domain specific language (DSL) was developed to enable educators to design quickly an adaptive VLE that will collect a range of information. Using this DSL, the educators would be able to configure the learning analytics that they think more important for their domain and adapt the learning path of each individual student according to the results of processing these learning analytics information. The processing of collected data will also be done using a range of algorithms through the user-friendly environment of *weka* library. Overall, an integrated solution to the adaptive VLE design problem will be given with clear instructions of how to reproduce this work.

A background study on adaptive VLE development and learning analytics will create the requirements for this research and is presented in Section II. In Section III, a block diagram of the framework design will be introduced, and the three stages of the process will be given. Stage one of the process will be presented in Section IV where our adaptive VLE DSL for collecting learning analytics data developed in *MPS Jetbrains* is detailed. The second stage of data analysis and processing using the *weka* library and its results will be then presented in Section V. The third stage of the process is the adaptation stage where the outcome of the second stage will be used as an input and will inform decision making for [...] in Section VII, an initial domain users (educators in this case) evaluation will be presented and our future plans will be detailed.

II. Background study-Adaptive VLEs with learning analytics

There are several attempts around personalised learning environments. In [1], the main aim of the research was to take a model for personalised learning environments and extend it to incorporate adaptability to several conditions such as mobile devices, adapting learning objectives, etc. In [2], a framework of two models for assessment-based personalisation was suggested. In [3] a personalised VLE based on different conceptual models was presented. In [4],

the research concentrated on the progress tracking and peer performance comparison which are new ideas, but the implementation demonstrated only two features without enough explanations for the student.

There is also considerable work around learning analytics which originally focused on clicking behaviour proxies leading to more complex behaviour models over time [5].

These research works presented some useful suggestions and models towards personalisation such as the use of student profile to collect information, progress tracking and peer performance comparison and new conceptual models.

However, common denominator in most of the above research is a weakness into putting into implementation theoretical education related ideas. The underlying problem, that also exists in other application domains, is that the domain experts of the field (educators) lack the technical knowledge to put these ideas into practice and influence the development of appropriate and effective adaptive VLEs.

Another noticeable work that includes good implementation is the work by Open University [6]. However, this work is not easily reproducible and is not accessible to a wide range of educators.

All this research led to the need to create a framework for adaptive VLE development that will enable educators to focus on domain issues leaving implementation details hidden and automatically provided.

In the following section, an overview of the framework-system design, the process and how they relate to the general MAPE-K adaptivity model will be given.

III. FRAMEWORK SYSTEM DESIGN AND PROCESS

An adaptive VLE belongs to the general category of adaptive systems which is becoming the norm for most systems of the future [7].

There are three important parts in all adaptive systems: *data collection*, *data processing* and *adaptation mechanisms* as stated in [8].

In our proposed system design for adaptive VLEs, we addressed the *data collection* and the *adaptation mechanisms* in the domain-specific language (DSL) implementation and handled the *data processing* part with a dedicated external environment, the *weka library* as shown in Fig. 1.
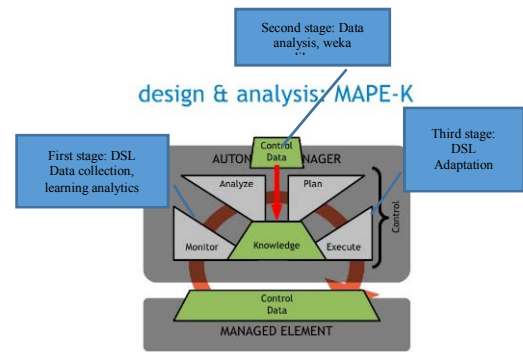

Fig. 1. MAPE-K and Adaptive VLE system design

In the following sections, the individual parts of the system will be detailed. The process to be followed and its three stages will be also be presented.

IV. FRIST STAGE: DOMAIN SPECIFIC MODELLING FOR ADAPTIVE VLE WITH LEARNING ANALYTICS

Research on domain-specific modeling such as domain-driven design [9] and domain-specific languages [10] has been increasingly recognised for being able to isolate the domain from the implementation details. It is exactly this type of framework environment that will enable a wide range of educators to "try out" their ideas without having to be involved in VLE implementation details.

A. Adaptive VLE DSL

The *MPS Jetbrains* environment was used for developing our domain-specific language.

Domain-specific languages developed through MPS Projectional editing environment consist of several main parts: structure, behavior, constraints, actions, editor.

The constructs in domain specific languages are quite similar to object-oriented design using modeling languages such as UML, SysML, etc. The level of abstraction is higher than the implementation level and implementation details are added gradually to the system. Both methods claim to provide tooling and automatic code generation. An attempt for automation, validation and verification is made by both approaches.

However, there are substantial differences between the two approaches. The DSLs are superior to the traditional UML code generation and tooling. The main reason behind it is the fundamental shift in the focus that differentiates the two approaches. The domain-specific languages focus on providing an interface for the domain expert that solves the domain problem and leaves the software implementation/tooling to be provided by the DSL development environment. The end-result is a language for the domain expert that can be used and maintained by the domain experts without the need of mixing software implementation information. The traditional UML descriptions attempt an abstraction from the implementation details. However, this is more general, doesn't get rid of all implementation details and doesn't provide tooling.

Fundamental aspect in DSLs is the concepts. The concepts are similar to classes in object-oriented design but

not quite. The concepts are strictly referring to domain classes and represent the main aspects of the particular domain.

Adaptive VLE language structure

In our adaptive VLE DSL, the following concepts and their structure represented in a UML diagram are depicted in Fig. 2 and Fig. 3.

In Fig. 2, an *AdaptiveVLE* consists of many *BlendedCourse*s. Each *BlendedCourse* contains a set of *WeeklyContents* (categorized in three main categories: standard weekly resources, low achievement weekly resources, high achievement weekly resources), and information related to learning analytics and adaptation such
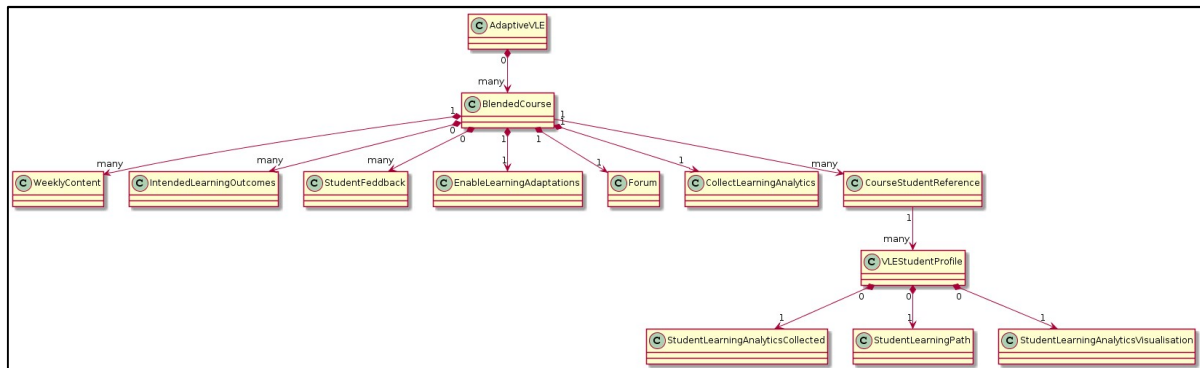
as: *CollectLearningAnalytics*, *EnableLearningAdaptations*, *StudentFeedback*, etc. as well as blocks that are required in order to collect learning analytics such as *Forum* (measuring the number of posts).

For each week also, the educator must provide three types of resources as is depicted in Fig. 3. This categorization is a simplification that addresses the personalization requirements in education environments. Depending on information that comes after the processing of the collected learning analytics and the outcome of the classification algorithm, the educator adds/removes resources from each category (third stage of the process: the adaptation stage).
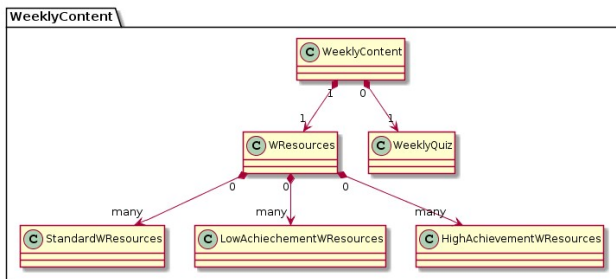

Fig. 2.   Adaptive VLE

Fig. 3.   WeeklyContent with resources for several abilities

AdaptiveVLE Editor

The Editor that is provided to the educator in order to develop an Adaptive VLE is presented in Fig. 4.

Using this editor, an educator of any discipline (without any programming knowledge) can concentrate on configuring the important aspects from the education perspective such as the collection of learning analytics and the upload of several types of resources. This editor appears "bare" and "minimal". However, the functionality and tooling behind it is very powerful. First of all, the advantages of projectional editing versus the traditional text-based editing are applicable as is detailed in [11]. Projectional editing enables the educator with restrictions and guidance through the fact that "coding behind the scenes" allows only the correct notation to be permitted and the user can edit only the parts that require input from him/her. It takes some initial time to get used to and obviously programmers are more likely to get familiarized quicker with this type of editing. However, the non-programming users such as educators of other disciplines are offered an environment/editor that

focuses only on the domain information making it easier for them to adapt.

The domain information focus is depicted by the fact that all elements in Fig. 4 are education-related. For example, learning analytics configuration and weekly contents. VLE implementation information such as web interfacing, adaptation mechanisms for the learning path (through editor actions), are hidden from the domain expert.
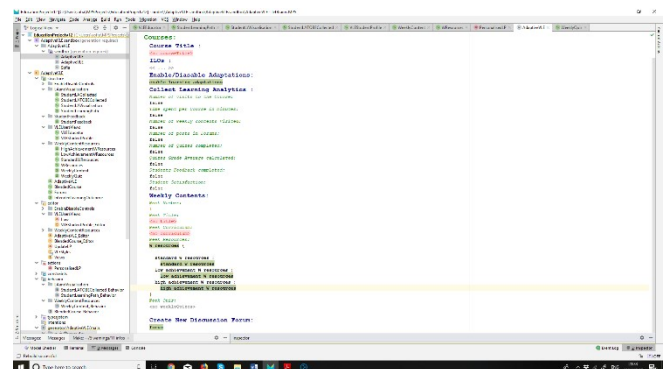

Fig. 4.   Editor for AdaptiveVLE concepts

V. SECOND STAGE: WEKA LIBRARY IMPLEMENTATION FOR LEARNING ANALYTICS PROCESSING

A. Data Processing

This part of the work starts after the data collection stage. Its main objective was to predict students' performance levels based on the learning analytics collected by the adaptive VLE at the previous stage. The data collected were learning analytics data such as number of students' visits to specific

weekly content, students' grades, etc. that were processed in order to extract valuable information regarding student performance levels (*target_performance* feature). This was achieved using another environment than stage one's DSL, that specializes in data analytics, the *weka* library.

Data collection

According to authors' experience in education environments, common logic at points, related research (presented in the background section) and University's VLE system-Brightspace [12] and its learning analytics provision, the data collected were selected to show the factors that could influence students' learning performance. Typical examples of relevant data were how many students used the platform, how long students spent on learning, the number of discussions they posted, how many quizzes they completed, their average grade of the quizzes, student satisfaction (regarding to students' sentiment conditions during learning), etc.

Data storage

Subsequently, the data was stored in the database automatically and was able to be retrieved by machine learning software *weka* library. *Weka* is a suite of machine learning software written in Java, developed at the University of Waikato. It contains a collection of data visualization tools and algorithms for automatic machine learning classification, clustering, association analysis etc. [13].

Feature selection

Nine features (see Fig. 5) were selected as key features with 2500 observations (sample records) in order to train machine learning models and deliver predictions. The prediction result is in the form of three categories: Low, Median, High, which is indicated as target class for machine to conduct prediction.

student_id	num_visited_course	time_spent_mins
num_topic_visited	num_discussion_posts	num_quizes_completed
quizes_grade_avg	student_feedback_complete	student_satisfiction

Fig. 5. Key features for machine learning

The 2500 observations were manually labeled for supervised learning.

B. Data Pre-processing

Before implementing machine learning algorithms, data pre-processing was required to transform raw data into a structured format and cleanse the noise amongst instances and features. Firstly, we delivered exploratory data analysis by data visualization for basic understanding and trends discovery (Fig. 6).

The *student_id* feature is a sequence of characters that is randomly assigned to each student, which would not provide any information assisting the machine learning algorithm. Thus, we decided to remove it in order to avoid interfering with the learning process of the algorithm.
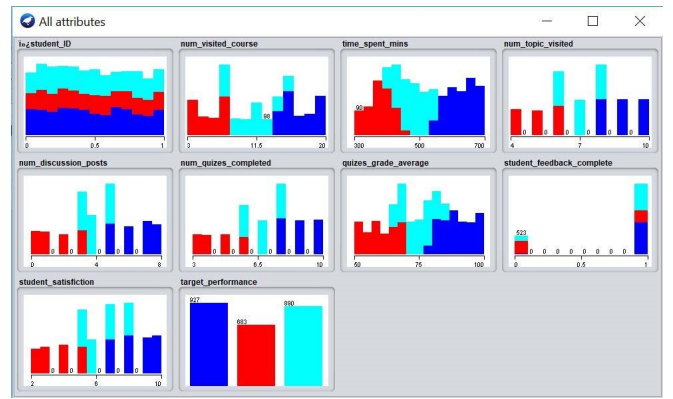

Fig. 6. Exploratory Data Analysis

Red and *light blue* and *dark blue* represent the sample data labeled "Low", "Median", "High" respectively in target class. There are some indicative patterns within the dataset that the samples in *light blue* are mostly lying between samples in *red* and *dark blue*, while the boundaries are not explicit. From the visualization, there is no obvious outliers, neither missing value based on *weka* automatic checking.

Although the entire dataset is in numerical form, the type of all the data is mixed, which consist of discrete data and binary data. The *student_feedback_complete* is the only feature that has binary values. The ranges of the data values are widely different from one feature to another. This might create some problems during machine learning process that misleading the classifier to over-weight the features with higher values and under-weight features with lower values. The issue might further lead to overfitting.

In order to avoid the above problems, we utilised standardisation technique. Standardization is one of the techniques for feature scaling, which can benefit machine learning and prediction process such as reducing training time, preventing overfitting, reducing prediction error, etc [14]. It can rescale the data value to fall into the range of [-1,1], with 0 for mean and 1 for standard deviation.

The transforming equation is shown in (1) below:

$$x_{new} = \frac{x - \mu}{\sigma} \qquad (1)$$

After pre-processing, the dataset was clean and fully structured and in a form that was ready to be used for model training.

C. Algorithm Selection

Since the target feature is in categorical form with three unique values, we identified the prediction task as a multiclassification classification task that would be using machine learning multiclass classification algorithms to solve the problem. According to the paper from Aly [15], several classification algorithms can be naturally extended to the multiclass case such as Decision Trees, Naïve Bayes, Neural Networks, K-Nearest Neighbor, Support Vector Machines, and Random forest. A multinomial logistic regression algorithm is also generalized for multiclass problems [16]. Therefore, these seven multiclass classification algorithms were implemented and trained using our dataset. The results

of their prediction performances were evaluated and compared with statistic metrices in terms of accuracy, recall, precision, and confusion matrix.

To prevent overfitting and to validate the machine learning models, K-fold cross-validation was applied by adjusting Test Options in *weka*. The 10-fold option was chosen. The dataset was shuffled randomly and split into 10 groups, then one group was taken as a test set whilst the remaining groups were used as training sets. Then selected classifiers were fitted and evaluated automatically by *weka*.

The whole process was then conveyed and implemented in java code using *weka* library APIs.

D. Results and Evaluation

The results of model prediction performance are shown below (Fig. 7). Random Forest was outperforming other classifiers with 99.04% accuracy, nearly 99% on both recall and precision scores. Only 24 samples were classified incorrectly out of 2500, which is the least amongst 7 classifiers. Seven algorithms were all providing good performance, and all achieved over 90% accuracy. The main reason behind this result was more likely to be that our dataset was ideally simple and structured, with distinct characteristics and patterns between each class. Therefore, further optimization is required by using more data with more noise and wider variety.

		NaïveBayes	LogisticRegression	MultilayerPerceptron	SVM	KNN	DecisionTree	RandamForest
Accuracy		93.28	94.04	97.24	94.48	95.16	98	99.04
Precision	Low	0.99	0.961	0.985	0.983	0.995	0.985	0.997
	Median	0.863	0.909	0.958	0.888	0.896	0.969	0.98
	High	0.972	0.956	0.977	0.979	0.981	0.987	0.996
Recall	Low	0.873	0.947	0.974	0.927	0.94	0.977	0.988
	Median	0.972	0.925	0.966	0.966	0.98	0.975	0.993
	High	0.94	0.95	0.977	0.937	0.933	0.987	0.989
Incorrectly classified		168	149	69	138	121	50	24

Fig. 7. Classification performance comparison

VI. Third Stage: Adaptation Process

In the third stage of the process, the adaptive VLE DSL is utilized again in order to present to the educator the outcome of the second stage and assist them into making decisions.

More advanced options of the editing experience were implemented in order to address the Adaptive VLE development application requirements. In Fig. 8, an editor that allows the educator to personalize the learning path for individual students according to information/results that are produced from the learning analytics data processing stage (described in the next section) is presented.
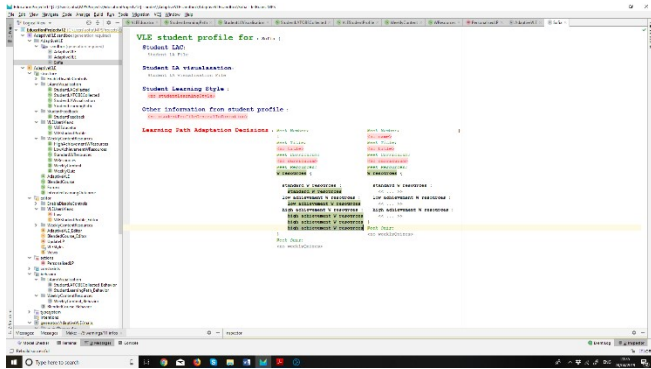

Fig. 8. Editor for adapting/persnolising the individual student learning path

In this editor, more advanced MPS DSL functionality such as *actions* is utilized in order to implement the personalised learning path feature. The educator has to update/synchronize the student learning path from the general course learning path and then is able to add resources according to the student learning analytics picture.

In order to create a copy of a part of a model, in MPS it is as trivial as invoking the "copy" operation on a node:

node<> deepCopy = course.weeklyContent.copy;
student.learningPath.add(deepCopy);

This code was used inside the action that was called from the editor when the educator presses CLICK on a particular part of the screen.

As you can see from Fig. 8, for *Week 1* for student *Sofia* the contents of the course have been copied to the student's individual learning path and additional *High Achievement Weekly Resources* have been added.

Note that the decision making for adding/removing resources is currently done manually by the educator. This could be automated in the future utilizing data science algorithms for decision making from the learning analytics data. This is actually leading to the future direction of this research.

VII. Initial evaluation – Conculsions and future plans

We tested our proposed environment with a limited number of educators (20) and the industrial partner's data science team to get an initial feel of the potential applicability of the design. This will be extended in the new academic year and appropriate plans have been put into place. The results were very positive and encouraging. Some points were made

regarding the collected learning analytics which will be taken into account in our future work.

In our immediate plans, response to the feedback we got from educators regarding the adaptive VLE requirements will be first addressed. For example, it was pointed out that further improvement could be achieved through collecting more learning analytics data and not only from the VLE, such as data from the registration process, or feedback survey, demographic information such as age, gender, study behavior such as study place, note taking behavior, or review behavior, and other relevant features. Consequently, our next plan is to fully-integrate the *weka* library functions in the DSL and add external data sources that were identified from the feedback.

The long-term plan of this research is to create a new language for developing "autonomic" VLEs that will be able to adapt automatically using data science algorithms for processing the learning analytics. This will still involve human interaction and the final and critical decisions will still be left for the educators which are the domain experts on this application area. The new language will focus on providing the users with interfaces/editors that focus on the particular domain and all other implementation details and tooling is hidden underneath and provided by the DSL. These implementation details/tooling include web interface implementation details, calling of data science algorithms for the learning analytics data collected and selecting the best algorithm for the corresponding data, automated adaptation of the individual learning path according to achievement, etc. The new language developed will enable educators of all disciplines to focus on their education domain issues and tailor the design of adaptive VLE to their needs. The focus on the domain is a general and very powerful research direction and can be extended to several application areas beyond education such as smart cities and IoT, health information systems, etc. Therefore, we plan to expand this research in more application areas as an enabler towards the next generation computing applications of the future.

REFERENCES

[1] S. Green, L. Nacheva-Skopalik and E. Pearson, "An adaptable personal learning environment for e-learning and e-assessment," in CompSysTech'08 Proceedings of the 9th International Conference on Computer Systems and Technologies and Workshop for PhD Students in Computing, Gabrovo, Bulgaria, 2008.

[2] S. Ong and I. Hawryszkiewycz, "Towards personalised and collaborative learning management systems," in Proceedings 3rd IEEE International Conference on Advanced Technologies, Athens, Greece, 2003.

[3] D. Xu and M. Wang, "A conceptual model of personalized virtual learning environments," Expert Systems with Applications, pp. Volume 29 (3), pg. 525-534, 2005.

[4] D. Verpoorten, C. Glahn, M. Kravcik, M. Ternier and M. Specht, "Personalisation of Learning in Virtual Learning Environments," in European Conference on Technology Enhanced Learning, Nice, France, 2009.

[5] D. Tempelaar, B. Rienties and B. Giesbers, "In search for the most informative data for feedback generation: Learning analytics in a data-rich context," Computers in Human Behavior, vol. 47, pp. 157-167, 2015.

[6] C. Calvert, "Developing a model and applications for probabilities of student success: a case study of predicitive analytics," Open Learning: The Journal of Open, Distance and e-Learning, vol. 29, no. 2, pp. 160-173, 2014.

[7] R. de Lemos and et al., "Software Engineering for Self-Adaptive Systems:A Second Research Roadmap," Lecture Notes in Computer Science, vol. 7475, 2013.

[8] J. Kephart and D. Chess, "The Vision of Autonomic Computing," Computer, vol. 36, no. 1, pp. 41-50, 2003.

[9] V. Vernon, Implementing Domain-Driven Design, Addison-Wesley, 2013.

[10] A. van Deursen, P. Klint and J. Visser, "Domain-Specific Languages: An Annotated Bibliography," SIGPLAN Notices, vol. 35, pp. 26-36, 2000.

[11] M. Voelter, DSL Engineering: Designing, Implementing and Using Domain-Specific Languages, dslbook.org, 2013.

[12] "Brightspace for Higher Education," D2L, [Online]. Available: https://www.d2l.com/en-eu/higher-education/products/. [Accessed 25 04 2019].

[13] I. H. Witten and F. Eibe, Data Mining: Practical machine learning tools and techniques, San Francisco: Morgan Kaufmann, 2011.

[14] J. Grus, Data Science from Scratch, Sebastopol: CA: O'Reilly, 2015.

[15] M. Aly, "Survey on multiclass classification methods, Technical Report," Caltech, USA, 2005.

[16] W. G. Greene, Econometric Analysis (Seventh ed.), Boston: Pearson Education, 2012.

[17] D. Xu and H. Wang, "A concetpual model of personalized virtual learning environments," Expert Systems with Applications, 2005.