Computer Aided Exercise Generation
A Framework for Human Interaction in the Automated Exercise Generation Process

Valentin Nentwich, Nicolai Fischer, Andreas C. Sonnenbichler and Andreas Geyer-Schulz
Information Services and Electronic Markets, Karlsruhe Institute of Technology, D-76128 Karlsruhe, Germany

Keywords: Automated Exercise Generator, Generation, Process.

Abstract: Creating exercises for tutorials and exams is part of lecturers' daily business. Manually creating new exercises for each course is very time expensive. Literature shows many technical solutions in terms of online learning platforms supporting exercise generation, answer checking and students' progress follow up. Most approaches solve specific problems and cannot be generalized in a way that a pattern for system design can be deducted. In this paper we point out a generic model for human interaction in the automated exercise generation process by identifying general constructs of exercise generation and by applying them on the outlined user interaction pattern. The application of the findings in a prototype for multi-criteria decision analysis indicates viability of the explained concept.

1 INTRODUCTION

Reaching and offering high quality mentoring and educational content is a desirable goal in the teaching sector of a university. The time consuming process of repetitive exercise generation captures avoidable ressources and therefore leads to bad quality of education. Otherwise, one could maintain high-quality tasks like promotion of talent programs or assistance of the gifted students.

An information and communication technology (ICT) driven approach promises to reduce required manual efforts thus freeing resources and enables teaching which contains offering educational content and supporting students' learning process in a radical new fashion. Exemplary, students can be offered an interactive learning infrastructure where self-directed learning is supported. Despite of learning in a self-service manner a direct or indirect tutoring system can be applied. Using ICT in the process of education requires a high automation level. In particular, system components have to be taken into account: the generation of exercises is one of the greatest problems identified by (Sadigh et al., 2012).

In terms of the problem of automatic exercise generation literature shows several specific approaches for automating task creation itself or parts of the generation process. In section 2 we examine existing concepts which serve as a basis for the in section 3 figured out constructs of exercise generation. In our understanding constructs are reusable components that can be used as building blocks for educational systems. Building blocks keep the system maintainable and enrich an automated exercise generation process.

In section 2 identified implementations for different education systems show a deep variation in its focus on system aspects and a lack of a common way in system design. Our main contribution is a framework for human interaction in the automated exercise generation process. The developed process closes the gap of interaction between lecturer and educational system and consequently serves as conceptual basis for the design of an education system.

The introduction of the exercise generation process in section 4 contains a precise explanation of the four process steps. The identified constructs of exercise generation (section 3) are associated with each corresponding process step. The exercise generation process serves as a basis for the design of our education system prototype. The implemented prototype in the field of use multi-criteria decision problems indicates viability and serves as proof of concept to show that the general idea works.

2 RELATED WORK

Gonzales and Munoz developed a web-based educa-

Nentwich, V., Fischer, N., Sonnenbichler, A. and Geyer-Schulz, A.
Computer Aided Exercise Generation - A Framework for Human Interaction in the Automated Exercise Generation Process.
DOI: 10.5220/0005947700570063
In *Proceedings of the 13th International Joint Conference on e-Business and Telecommunications (ICETE 2016) - Volume 2: ICE-B*, pages 57-63
ISBN: 978-989-758-196-0

tional software for providing students with statistical and mathematical exercises. Further features include evaluations of students answers as well as data management and support for exercise generation to teachers (Gonzalez and Munoz, 2006).

Aldabe et al. present a question generator which automatically creates exercises for students. Automation is reached by making use of natural language processing (NLP) techniques and corpora. As concrete results exercises for fill-in-the-blank, word formation as well as multiple choice and error correction are constructed (Aldabe et al., 2006).

Sadigh et al. show a template-based approach to establish automation in the teaching life cycle addressing creation of problems and corresponding solutions as well as grading (Sadigh et al., 2012).

Merceron and Yacef explain an approach for a web-based tutoring system giving feedback for students and teachers. Students are provided with a step by step feedback during solving exercises. A mining component establishes the opportunity for analysing students answers to feedback teaching methods (Merceron and Yacef, 2003).

Chrysafiadi and Virvou present different approaches for student modelling. Student modelling includes initial assessment, tracking and adapting of the student knowledge level and preparation of exercise presentation for individuals (Chrysafiadi and Virvou, 2013).

Lopez et al. designed a coached problem solving system for teaching the simplex algorithm. The implementation contains a problem generator, a student model based exercise adaptation and a dynamic help system for assisting students in improving their mistakes (Lopez et al., 1998). Volodina and Borin use a similar approach for teaching languages (Volodina and Borin, 2012).

Devedzic describes an intelligent web-based education system by clustering basic components of today's educational technology into a framework. For higher reusability and flexibility the educational content is described by semantic aspects (Devedzic, 2003).

Melis et al. use a knowledge base for dynamically generating individual, mathematical tutoring content. Semantic methods serve as a main building block for the realisation. Additionally, a interface for connecting stand-alone service systems opens the tutoring tool in terms of public educational material (Melis et al., 2001). Almeida et al. propose a specific language for designing more complex exercises than multiple choices, data type or file comparisons (Almeida et al., 2013). Further ontology based approaches can be found in (Holohan et al., 2006).

Adamopoulos' research paper identifies key characteristics influencing the long term affinity of users to web based learning environments. The most important key characteristics are involved professors, the students' attitude towards teaching contents as well as difficulty, workload and duration of a lesson (Adamopoulos, 2013). Workman depicts how computer-based education and computer-aided education influence students' ability to learn (Workman, 2004).

Vossen and Westerkamp propose to publish learning content through web service interfaces. A learning content database consisting of e-learning courses and learning objects serves as a basis for web service methods. The web service interface implements, inter alia, content presentation, user tracking and content creation (Vossen and Westerkamp, 2003).

3 CONSTRUCTS OF EXERCISE GENERATION

In this section we present constructs that are able to enrich an automated exercise generation process. According to (Gregor and Jones, 2007) constructs describe entities of interest related to a specific design theory which can be simple terms and mathematical symbols or complex sub-systems. Identification of these constructs is based on section 2.

The following constructs were identified: ontologies, student modelling, feedback mechanisms, web service interfaces and a help suggestion system.

Gruber describes ontologies as a formal specification for classes, relations as well as functions and objects in a shared domain of interest (Gruber, 1993). Formal specification makes ontologies machine-understandable which is a key functionality for automation.

A student model is the result of the process for describing and analysing cognitive skills of a specific student (Self, 1990). Chrysafiadi and Virvou mention this characteristic as a base for personalisation and state that it is a decisive factor for adaptive educational systems (Chrysafiadi and Virvou, 2013).

Franklin et al. delineate a feedback mechanism as a dynamic system where measured output is used for determining the system's future behaviour (Franklin et al., 1994). Such feedback mechanisms are the foundation for assisting in educational systems as portrayed by (Merceron and Yacef, 2003).

Literature shows that the term web service is ambiguous. For our work we refer to Papazoglou who outlines service interoperability, service orchestration, service transaction management and service coordination as characteristics for web services (Papazoglou, 2003). These enable simple network access on teaching platform components and therefore reusability.

Lopez et al. consider student assistance by solving exercises as a necessary element of an educational system (Lopez et al., 1998). With regard to this fact the exercise generation process has to provide a functionality for suggesting capabilities to support students – a help suggestion system.

The identified constructs can be used to enrich automated exercise generation whereas each component is part of a specific solution and realises different tasks. Today, there is a lack of design method for building exercise generation systems on top of multitude system components. In the following sections, we introduce a generic process for educational system design to face this challenge.

4 EXERCISE GENERATION PROCESS

In this section, we introduce our exercise generation process referring to figure 1. We explain every process step and associate the constructs identified in section 3 giving a short example. Furthermore, we point out the necessary technical aspects for the realisation as a web educational system serving as a foundation for our prototype.

4.1 Select and Modify Exercise Context

First step of the exercise generation process is the selection and modification of the exercise context.

The exercise context specifies the exercise framework and thus builds the association to a certain field of use. For example we have the problem to purchase a car. In this situation the purchaser has to choose between several alternatives and selects the one which meets best his requirements. This example shows a specific exercise context for the abstract problem of decision making. Further problems of this category could be an investment decision or choosing the right university.

For building an exercise context two different approaches are conceivable: manually or automated. A naive method is to manually build up and maintain a context database by lecturers. Automated context generation can be reached by making use of the construct ontologies. Technically, we call this component a context generator.

Applying these approaches on the sub process ”Select and Modify Exercise Context” both human and machine have different tasks to complete. In case of manually created exercise contexts lecturers have to imagine and construct fictitious fields of use to feed an exercise context database. For automated generated exercise contexts machines take this part. However we assume, that an exercise context database exist and for the manually as well as the automated approach the lecturers simply choose a suitable context for building up his exercise during the exercise generation process. In this step, modification gives the opportunity to correct mistakes and to apply lecturers' own style.

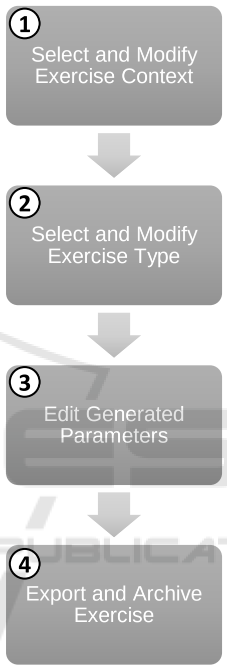
1
Select and Modify Exercise Context
2
Select and Modify Exercise Type
3
Edit Generated Parameters
4
Export and Archive Exercise

Figure 1: Exercise generation process.

4.2 Select and Modify Exercise Type

Second step of the exercise generation process is the selection and modification of the exercise type.

An exercise type defines a general structure of an exercise. For the mentioned example of purchasing a car as a multi-criteria decision problem multiple strategies for making the decision exist. Exemplary one could sum up the values-in-use for each alternative and choose the best one. Another example could be choosing the alternative which has the highest score for the leading goal. Each decision strategy leads to a different general exercise type which can be implemented by different instances. Instances of the same general exercise type can differ in their specific structure. Exercise types serve as input for the automated creation of students' exercises by the component exercise generator.

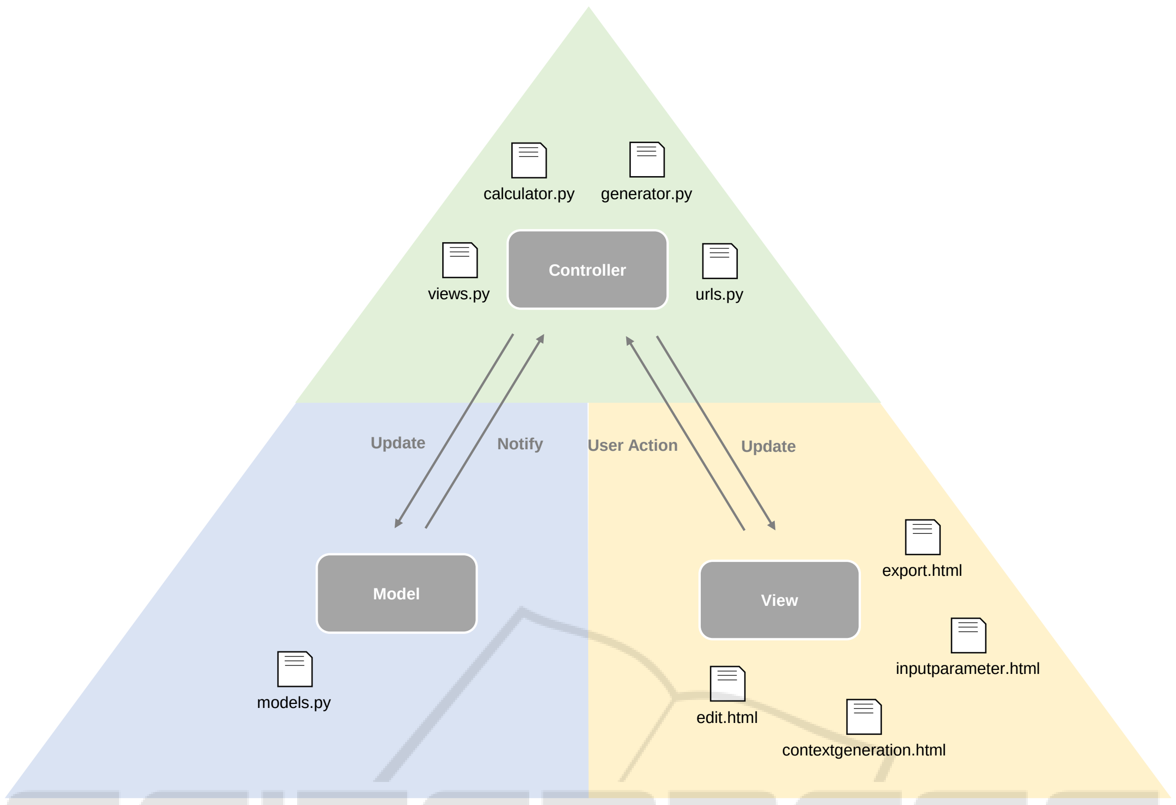

Figure 2: Prototype scheme (MVC as described in (Ullenboom, 2014)).

The characteristics for lecturers' interaction in the sub process "Select and Modify Exercise Type" are selecting an appropriate exercise type and defining its specific structure by establishing individual parameters. Referred to our example, a lecturer can choose target weighting as exercise type and set as parameters the individual weights for each target. In accordance with this input the exercise generator component automatically creates the exercise.

4.3 Edit Generated Parameters

Third step of the exercise generation process is editing the parameters generated by the exercise generator.

The exercise generator's output is a complete task for students including exercise text, questions, hints for solution and solutions themselves. We assume that developers' cognitive skills are limited and even making use of ontologies and natural language processing in automated exercise generation leads to inaccurate or unfair produced exercises, respectively exercise parameters. Considering this, the basic idea of this process step is to give lecturers the chance to adjust parameters.

In this process step it is possible to install three constructs presented in section 3: feedback mechanisms, the help suggestion system and the student model.

As mentioned in (Merceron and Yacef, 2003), stored student answers can be mined to gather information about typical problems or mistakes which occur while solving an exercise. Properly processed, the information can be used as feedback for the lecturer and help him to adjust parameters of a certain exercise type accordingly.

There are two different approaches to work with the help suggestion system. Firstly, it is conceivable that a lecturer is able to turn on or to turn off specific features of the help suggestion system manually – as a consequence the lecturer determines how much students are supported. An automated approach is to support students dynamically according to their knowledge level represented by a student model – as a consequence the students are supported individually by the help suggestion system.

Lecturers interact in this process step by evaluating machine produced exercises and by adjusting phrasing as well as input parameters for calculations. According to the previous example, the exercise generator creates the textual description of the exercise as well as the matrix containing the values-in-use for the target weighting decision problem. Lecturers' are able to edit the exercise text and the generated values of the matrix. If there are changes in input values for calculation the exercise generation component has to automatically adapt proposed solutions and assistance for solving the problem.

4.4 Export and Archive Exercise

Fourth step of the exercise generation process is exporting and archiving the generated exercise.

Basic idea of this step is, that done work has to be saved for further use. The export functionality gives tutors the ability to convert the generated exercise into a specific data format like the portable document format (PDF) or LaTeX. Archiving means to store the generated exercise in a database for future lookup and processing.

In the sub process ”Export and Archive Exercise” lecturers’ interaction in the simplest case is choosing the data format for exporting and to record general data like the lecturer’s name or the description of the semester. Technically, a component for archiving and a component for exporting have to exist to perform machine parts in this process step.

In our example, the lecturer chooses LaTeX as export format. As a result the lecturer gets a LaTeX file containing the exercise description, questions, a matrix with values-in-use and a solution outline adjusted for the case of the target weighting decision problem ”purchasing a car”.

5 PROTOTYPE

In this section we present a prototype which indicates viability of the explained exercise generation process. We fill up the prototype by an example in the field of use multi-criteria decision problems. We explain the technical set up of our prototype pointing out the general as well as the detailed structure. In each step we show how the exercise generation process can be referenced to the prototype in the scope of an economic example.

5.1 General Structure

This section gives a short overview about the general structure of the prototype and its implementation. Vossen and Westerkamp propose web service interfaces for realisation of educational systems (Vossen and Westerkamp, 2003). Furthermore, the model-view-controller pattern (MVC) is a common agreed design pattern for designing a component based application where each component can be developed independently (Ullenboom, 2014). We use Django (Django Documentation, 2015) as a simple web framework based on Python (Python, 2015) which supports MVC and thus reusability.

Figure 2 shows the general scheme of the prototype considering the MVC pattern as described in (Ullenboom, 2014): the model represents the internal state of an object for enabling persistent data storage and manipulation. Views define how data is presented in the user interface. The controller processes the user interaction including model and view updating.

5.2 Model

This section includes a detailed explanation of the model’s structure. Every class describing a model object in Django is placed into the models.py file (Django Documentation, 2015). We use the data model shown in figure 3 to describe these classes.

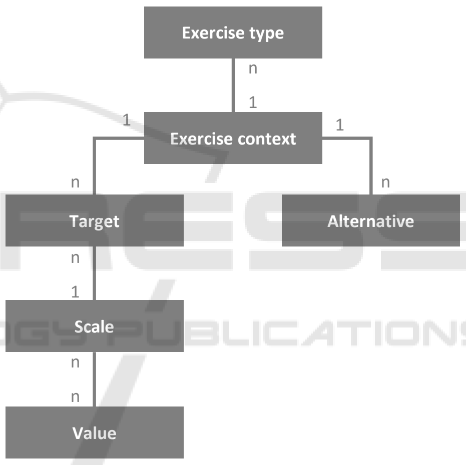

Figure 3: Data model used for implementing models.py.

An exercise type defines the general structure of an exercise. In case of the example ”purchasing a car” as a multi-criteria decision problem the following exercise types are conceivable: target weighting, lexicographical ordering, maximum of the minimal target achievement, goal programming and mixed up exercise type consisting of the types mentioned.

The scenario of an exercise plays in a specific context. Therefore we introduced the generic component exercise context, which can be applied on several exercise types. The scenario in our example plays in the context of ”purchasing a car”. As exercise type target weighting is selected. It is possible to choose e.g. goal programming as exercise type by using the same context of ”purchasing a car”.

A generic exercise context is composed by several targets and alternatives. Targets in the sense of a multi-criteria decision problem display the value proposition of a decision maker for solving the decision problem. The value of a target is expressed by a scale which is implemented by discrete or continuous values. Alternatives represent options that can be chosen by a decision maker.

To complete our example, we add alternatives, targets, scales and values. Alternatives in our case could be the cars ”Audi A3”, ”VW Golf” and ”Scoda Octavia”. Attributes the cars have in common can be viewed as desired goals to achieve. Let’s say we consider ”PS”, ”fuel consumption”, ”mileage” and ”CO2 conformity” as viewed targets to compare the cars. To achieve the processing of given targets we add a scale to each target. Processing means the selection of the best alternatives according to the exercise type’s methods. A scale e.g. can be kilometers with the values 10km, 100km and 1000km in case of the target ”mileage” or boolean (true/ false) in case of the target ”CO2 conformity”.

5.3 View

Views define how data is presented in the user interface. For the view component in the prototype we designed different HTML templates (as in figure 2) matching the requirements of the exercise generation process.

The first step of the exercise generation process is realised by the inputparameter.html. Within this template a existing exercise context can be chosen and targets as well as alternatives can be adapted for the specific exercise type. If there is no appropriate exercise context, the user can navigate to contextgeneration.html which is a template for manually creating generic exercise contexts. Moreover, inputparameter.html includes the second step of the exercise generation process. Our protoype contains the following choosable and editable exercise types: target weighting, lexicographical ordering, maximum of the minimal target achievement, goal programming and mixed up exercise type consisting of the types mentioned.

Input parameters are processed by the controller to generate an exercise. The user is navigated to the edit.html where automatically created exercise parameters can be viewed and adjusted by the lecturer. In our prototype the entries of the use matrix can be edited.

After editing users have the opportunity to export and to archive the complete exercise (export.html). Export formats that can be chosen are PDF and $\LaTeX$.

5.4 Controller

The controller navigates the user and processes input parameters. Django realises navigation by the use of views.py and urls.py. Url requests are catched by the urls.py and directed to the mapped view. Views are build by views.py which is responsible for parameter processing. Processing work is assisted by calculator.py and generator.py. HTML templates of the view component are enriched by generated parameters.

The supporting component generator.py automatically creates the exercise including exercise text, input values for calculation, questions as well as a solution outline. Generator.py fills a static exercise text with the values of the selected exercise context. Additionally, a use matrix is set up with random values. To each exercise type static questions are assigned. As a naive approach for the implementation of the help suggestion system, we provide an solution outline automatically composed by generator.py.

The supporting component calculator.py offers different mathematical methods that can be used for calculation in generator.py. Example methods are: generating random values, finding minimum or maximum values with respect to targets’ scales and solving decision problems.

6 CONCLUSION

In this article, we described a framework for human interaction in the automated exercise generation process. We examined current approaches for educational system design (section 2) to identify basic components realising different tasks within the specific solutions reviewed. Our contribution lies in the creation of a generic process as a design method for developing educational systems.

In section 3, we presented constructs that are able to enrich an automated exercise generation process. The following constructs were identified: ontologies, student modelling, feedback mechanisms, web services interfaces and a help suggestion system.

In section 4, we introduced our generic exercise generation process. Furthermore, an associations between each process step and its corresponding constructs is made. Additionally, we pointed out necessary technical aspects for the realisation as a web educational system.

In section 5, we presented our prototype indicating viability of the explained concept. Therefore, we described the prototype’s set up basing on the exercise generation process. Firstly, we pointed out the general structure of our Python/Django implementation. Secondly, we give a detailed insight into the used structure for implementing our protoype: the design according to the exercise generation process is transferred into the technical view of the model-view-controller pattern.

The paper at hand presents a process on a high level of abstraction including general concepts of technical solution for educational systems. Regarding future work, a more extensive analysis by expanding the review of existing theoretical and practical approaches can lead to a more precise shape of our exercise generation process.

REFERENCES

Adamopoulos, P. (2013). What makes a great mooc? An interdisciplinary analysis of student retention in online courses. In *Thirty Fourth International Conference on Information Systems*, pages 1–21, Milan.

Aldabe, I., de Lacalle, M., Maritxalar, M., Martinez, E., and Uria, L. (2006). Arikiturri: An automatic question generator based on corpora and nlp techniques. In *Intelligent Tutoring Systems*, volume 4053 of *Lecture Notes in Computer Science*, pages 584–594. Springer Berlin Heidelberg.

Almeida, J. J., Araujo, I., Brito, I., Carvalho, N., Machado, G. J., Pereira, R., and Smirnov, G. (2013). Passarola: High-order exercise generation system. In *Eighth Iberian Conference on Information Systems and Technologies (CISTI), 2013*, pages 1–5, Lisboa. IEEE.

Chrysafiadi, K. and Virvou, M. (2013). Student modeling approaches: A literature review for the last decade. *Expert Systems with Applications*, 40(11):4715–4729.

Devedzic, V. B. (2003). Key issues in next-generation web-based education. *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, 33(3):339–349.

Django Documentation (2015). [online]. available: https://docs.djangoproject.com/en/1.8/. Requestdate: 2015-10-04.

Franklin, G. F., Powell, J. D., and Emami-Naeini, A. (1994). Feedback control of dynamics systems. *Addison-Wesley, Reading, MA*.

Gonzalez, J. A. and Munoz, P. (2006). e-status: An automatic web-based problem generatorapplications to statistics. *Computer Applications in Engineering Education*, 14(2):151–159.

Gregor, S. and Jones, D. (2007). The anatomy of a design theory. *Journal of the Association for Information Systems*, 8(5):312–335.

Gruber, T. R. (1993). A translation approach to portable ontology specifications. *Knowledge acquisition*, 5(2): 199–220.

Holohan, E., Melia, M., McMullen, D., and Pahl, C. (2006). The generation of e-learning exercise problems from subject ontologies. In *Sixth International Conference on Advanced Learning Technologies (ICALT), 2006*, pages 967–969, Kerkrade. IEEE.

Lopez, J., Millan, E., Perez-De-La-Cruz, J., and Triguero, F. (1998). Ilesa: a web-based intelligent learning environment for the simplex algorithm. In *Computer Aided Learning and Instruction in Science and Engineering*, volume 98, pages 399–406, Gothenburg.

Melis, E., Andres, E., Budenbender, J., Frischauf, A., Goduadze, G., Libbrecht, P., Pollet, M., and Ullrich, C. (2001). Activemath: A generic and adaptive web-based learning environment. *International Journal of Artificial Intelligence in Education (IJAIED), 2001*, 12:385–407.

Merceron, A. and Yacef, K. (2003). A web-based tutoring tool with mining facilities to improve learning and teaching. In *Eleventh International Conference on Artificial Intelligence in Education*, volume 97, pages 201–208, Sydney. IOS Press.

Papazoglou, M. P. (2003). Service-oriented computing: Concepts, characteristics and directions. In *Fourth International Conference on Web Information Systems Engineering (WISE), 2003*, pages 3–12, Rom. IEEE.

Python (2015). [online]. available: https://www.python.org/. Requestdate: 2015-10-04.

Sadigh, D., Seshia, S. A., and Gupta, M. (2012). Automating exercise generation: A step towards meeting the mooc challenge for embedded systems. In *Proceedings of the Workshop on Embedded and Cyber-Physical Systems Education*, pages 1–8, Tampere. ACM.

Self, J. A. (1990). Bypassing the intractable problem of student modelling. In *Intelligent tutoring systems: At the crossroads of artificial intelligence and education*, pages 107–123. Citeseer.

Ullenboom, C. (2014). *Java SE 8 Standard-Bibliothek : das Handbuch fuer Java-Entwickler*. Galileo Computing. Galileo Press, Bonn.

Volodina, E. and Borin, L. (2012). Developing a freely available web-based exercise generator for swedish. In *EUROCALL Conference, Gothenburg, Sweden, 22-25 August 2012 – CALL: Using, Learning, Knowing*, Gothenburg.

Vossen, G. and Westerkamp, P. (2003). E-learning as a web service. In *Seventh International Database Engineering and Applications Symposium, 2003*, pages 242–249, Hong Kong. IEEE.

Workman, M. (2004). Performance and perceived effectiveness in computer-based and computer-aided education: do cognitive styles make a difference? *Computers in Human Behavior*, 20(4):517–534.