A Web-based Recommendation System for Engineering Education e-Learning Systems

Thorsten Sommer, Ursula Bach, Anja Richert and Sabina Jeschke
IMA - Institute of Information Management in Mechanical Engineering
ZLW - Center for Learning and Knowledge Management
IfU - Institute for Management Cybernetics
Faculty of Mechanical Engineering RWTH Aachen University, Aachen, Germany

Keywords: E-Learning, Recommendation System, Agile Process, Teachers, Professors, Web 2.0, Software Engineering, Open Source.

Abstract: Today there is a flood of e-learning and e-learning related solutions for engineering education. It is at least a time consuming task for a teacher to find an e-learning system, which matches their requirements. To assist teachers with this information overload, a web-based recommendation system for related e-learning solutions is under development to support teachers in the field of engineering education to find a matching e-learning system within minutes. Because the e-learning market is subject of very fast changes, an agile engineering process is used to ensure the capability to react on these changes. To solve the challenges of this project, an own user-flow visual programming language and an algorithm are under development. A special software stack is chosen to accelerate the development. Instead of classical back-office software to administer and maintain the project, a web-based approach is used – even for a complex editor. The determining of the necessary catalog of related solutions within "real-time" is based on big data technologies, data mining methods and statistically text analysis.

1 INTRODUCTION

To help teachers with their different challenges about finding an e-learning solution, a web-based recommendation system for e-learning systems is under development. This recommendation web-based service enables teachers to choose an engineering education e-learning system, which matches her or his requirements.

The term "e-learning" is often used in different matters. Therefore, this definition is chosen: "E-learning is an approach to teaching and learning, representing all or part of the educational model applied, that is based on the use of electronic media and devices as tools for improving access to training, communication and interaction and that facilitates the adoption of new ways of understanding and developing learning." (Sangr et al., 2012) This definition includes any computer- and web-based tool, which is related to the education context.

A variety of e-learning systems and environments (Mayer, 2003) are observable and the amount is continuous growing: From classical computer-based training (CBT), web-based training (WBT) (Schoen and Ebner, 2013), wikis and blogs (Schoen and Ebner, 2013), podcasts (Cebeci and Tekdal, 2006) respectively educasts (Schoen and Ebner, 2013) and game-based learning (Schoen and Ebner, 2013) up to massive open online courses (MOOC) (McAuley et al., 2010; Schoen and Ebner, 2013).

To illustrate the amount of related and available resources: A simple web-search for "e-learning" ends with over one billion results, a web-search for "e-learning system" with over 480 million results! Nearly 80 unique e-learning systems can be found in short time. This fact demonstrates the problem: The interested teacher must investigate this amount of information to find an e-learning system, which matches their personal requirements. Another problem is: The teacher might not be able to choose a system, which matches their requirements, because it is not trivial to understand all the technologies and differences between the unique systems.

Sommer T., Bach U., Richert A. and Jeschke S..
A Web-based Recommendation System for Engineering Education e-Learning Systems.
DOI: 10.5220/0004930203670373
In *Proceedings of the 6th International Conference on Computer Supported Education* (CSEDU-2014), pages 367-373
ISBN: 978-989-758-020-8

2 REQUIREMENTS AND CHALLENGES

The main precondition of the desired recommendation system is that the related e-learning systems are comparable. Moreover, a catalog of related solutions must exist. The approach to reach the comparable state is to find a set of necessary attributes that describes the characteristics of engineering education e-learning systems. These common e-learning characteristics must base on a broad scientific consensus: To realize this, the input of many experts is acquired.

The collected data about each e-learning system, together with the e-learning characteristics, results in a comparable data sheet about each e-learning system. This data sheet is subject of continuously changes to ensure that the data sheet is up-to-date and represents the current state of each system. Also the e-learning characteristics are subject of changes to cover all related kinds of e-learning.

The traditionally approach to determine the catalog with the solutions uses a lot of resources (time, staff and money): Pay and get every e-learning system, prepare a server environment and install all systems. Then investigate the system (as teacher and student) and fill-up the data sheet. It is possible to speed-up this by using virtualization environments (Rosenblum, 2004) like e.g. a type 1 hypervisor (Fenn et al., 2008) with templates for the required environments, system snapshots and derivation between them.

The new and promising approach to determine the catalog with the available solutions is based on text mining: Crawling and parsing the public vendor and community information about the e-learning systems and store the raw data. Next, the raw text data is able to get analyzed to find out about the textual context. A half-automated algorithm suggests then a value for each characteristic, to assist the employee. Such a process is able to get executed e.g. every quarter to ensure that the data sheets are up-to-date.

To provide a convenient tool to develop and maintain the questionnaire, a new visual user-flow programming language is defined. This language is linking the catalog of solutions, the questions with further explanations for the teachers, the e-learning characteristics and the model of the user-flow together. Compared to existing survey solutions, the visual model and the deep integration are new.

Another challenge is that teachers expect a current, modern, responsive (Mohorovicic, 2013) and accessible user interface (UI). This is comprehensible, because it allows any teacher with any device and any handicap to use this web-based recommendation service. A responsive UI saves also time, because there is no need for an additional mobile and tablet websites (Mohorovicic, 2013). Some web frameworks (e.g. Bootstrap[1]) assist the developer in these fields.

For further research (e.g. about e-learning systems and the teachers requirements to these systems), it would be helpful to collect some kind of key performance indicators (KPI) from the recommendation system. The data must be anonymous to keep the teachers privacy. Not only the end results must be logged, also e.g. the reaction time per question and – if present – the cancellation point etc. It is also interesting to capture all single decisions of any anonymous teacher to enable research e.g. in psychology fields.

3 USER FLOW LANGUAGE

To enable the scientific assistants to model efficient the user-adapting questionnaire and to provide a convenient tool for maintaining the questionnaire, a new visual programming language (Hils, 1992) is defined.

The language is simple: Different squares – called ”function blocks” – are connected by wires. For different purposes, different function blocks are present: Start, end, question, numeric and range blocks. Every function block has no or one input connector and no, one or three output connectors – this depends on the kind of the block. Behind every block, some data is stored: The reference to the common e-learning characteristic, a question, additional explanation text or just a text message – depends on the kind of the block.

Every program must have exact one start block, and at least one end block. The visual program must read from left to right: From the start block at the left, then follow block by block until reaches any of the end blocks. The concrete questionnaire can then deviated out of the visual program by simple traveling through the blocks. There is no special algorithm required to get or generate the questionnaire.

With this visual language, the visual program for the questionnaire has to be built. Explaining figure 1 as current example – to clarify the visual language: At the left, the start block was placed. The interviewee gets an introduction message to read. The flow is directed to ”Question 1”, a question block. The interviewee gets a question (the question was defined before by a research assistant) and an additional explanation text. Related to the answer, the flow reaches ”Numeric query 1” or ”Question 2”. The ”Numeric query 1” prompts the teacher to answer a numeric question like e.g. ”How many students are in your

[1] http://www.getbootstrap.com

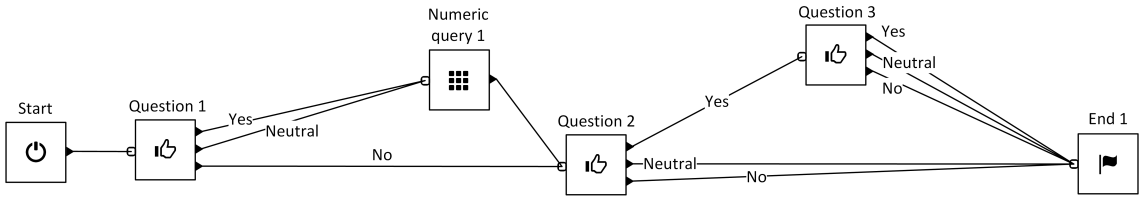

Figure 1: Example of the user flow visual programming language.

class?" From this block, the flow also reaches "Question 2". After "Question 2", the flow is able to reach "Question 3" or "End 1", related to the answer. "End 1" is also reached after answering "Question 3". At the end block, the interviewee is able to read a finished message. By pressing a button, the user submits all their answers to the algorithm.

Each question block has one input and three output connectors for the further flow: A yes-output, a no-output and a neutral-output. This corresponds to the possible answers of the interviewee. "Yes" means that the issue (subject of the question) must be present, "no" means the issue is not present or can be disabled and "neutral" means, that the issue does not matter.

A range block (this kind of block is not part of the example at figure 1) contains a text and an explanation text for the interviewee. To represent a range, this block needs two references to the corresponding parts of the common e-learning characteristics. This kind of block has one input and output connector.

While this project grows, the amount of different kind of function blocks will increased as necessary. For any new kind of block, also the algorithm (see section 5) must be extended to cover the new functionality. New types of blocks are caused by changes at the common characteristics, to cover new requirements on the e-learning market.

For convenient usage, a simple web-based editor is under development. Thereby, it is possible to maintain the questionnaire without any programming skills. The whole life-cycle of the editor and the language is also convenient: There is no installation required, updates are only a server-side deployment and any kind of maintenance just occur on the server-side.

4 ENGINEERING

4.1 Process

To be able to react on new requirements on the whole process (software engineering, determining catalog of solutions, development at the algorithm etc.), agile best practices are chosen (Wolf, 2011; Fowler and Highsmith, 2001; Poppendieck and Cusumano, 2012):

- Use cases: Define a few use cases by drawing diagrams or by writing small so called "user stories" (Wolf, 2011).
- Simplicity: Develop just necessary parts and leave anything which is not required (Poppendieck and Cusumano, 2012; Wolf, 2011).
- Fast: Release fast and as many as possible to be able to get feedback from others (Poppendieck and Cusumano, 2012; Wolf, 2011).
- Communication: Get early and continuously feedback from the customer, to ensure the project fits the requirements (Wolf, 2011).

Additionally, not manageable challenges are divided into smaller – but manageable – challenges (Kuster, 2011; Wolf, 2011). It is also necessary to choose the right tool for right purpose: Do not use the same tool for anything – there are right purposes for any tool, but not all tools are convenient for all problems.

Even though different changes at the last three months, it was possible to reach the current state after just eight weeks of work with just one person: This is perhaps related to the agile process. The process is promising for the further work and research.

4.2 Client-Side: Web-UI Approach

At least two tools are required for the back-office: The editor for the visual language and the product editor. While the product editor is quite simple, the editor for the visual language is even more complex. Anyhow, a new approach is tested: Instead of developing the website for the teacher's questionnaire and additional software for the back-office (with Java or .NET), anything will be developed as web-application.

For the web-application (for the back-office tools and also for the teacher's questionnaire) just HTML5, CSS and JavaScript with Bootstrap[2] and jQuery[3] is used. Thereby, it is also useable on tablets like e.g. iPads – independently of the operating system. Further, also the final client performance is fine.

[2]http://www.getbootstrap.com
[3]http://www.jquery.com

4.3 Server-Side: Software Stack

On the server-side, a wide range of options are available. The best fit is a ”BGMR stack”: FreeBSD[4], nginx[5], MongoDB[6] and Ruby[7]. FreeBSD (Niessen, 2012) is very robust, secure, uses small resources and with the concept of ”Jails” (Kamp and Watson, 2000; Kamp and Watson, 2004) a powerful virtualization (Rosenblum, 2004) environment is provided.

The web-server nginx (Nedelcu, 2013) is famous as powerful proxy, load-balancer and fasted server for static content (Dabkiewicz, 2012). Nginx delivers all necessary static data for this project (images, CSS, JavaScript, fonts etc.) But nginx is the wrong tool to deliver dynamic content, because the handling of external processes like e.g. the common PHP[8] is less efficient (Dabkiewicz, 2012). Therefore, non-static requests are passed-through to an application server.

MongoDB (Chodorow, 2013) is a robust, fast (Boicea et al., 2012; Wei-ping et al., 2011) and scalable database. For the most Web 2.0 projects, MongoDB fits perfect, because the database is document-based and uses JSON (Crockford, 2006). If the project grows, the database is also able to scale.

Ruby is an object-oriented programming language (Flanagan and Matsumoto, 2008) that provides a very convenient and fast way to develop web applications: On top of Ruby, the Sinatra[9] framework (Harris and Haase, 2012) provides an own web-centric DSL (domain-specific language). This enables a strong focus to web-oriented programming and reduced the necessary overhead a lot. While the project runs on the development environment with Ruby, for the production environment it is designated to run it with JRuby (Nutter, 2011) on JavaEE with the JBoss application server (Kutner, 2012).

This approach is very promising, because it speeds up the development, keeps the code clear – which makes the maintenance easier – and provides later with JRuby the power and scalability of JavaEE and the JBoss application server (Nutter, 2011; Kutner, 2012).

5 ALGORITHM

To get the expected results out of the teacher's answers, an algorithm is under active development (see figures 2, 3, 4 and 5). The current state of the algorithm is constructed and validated with dedicated test data: As developing and testing environment, a normal spreadsheet application is chosen. A table with test data is representing the results of the questionnaire (the teacher's answers).

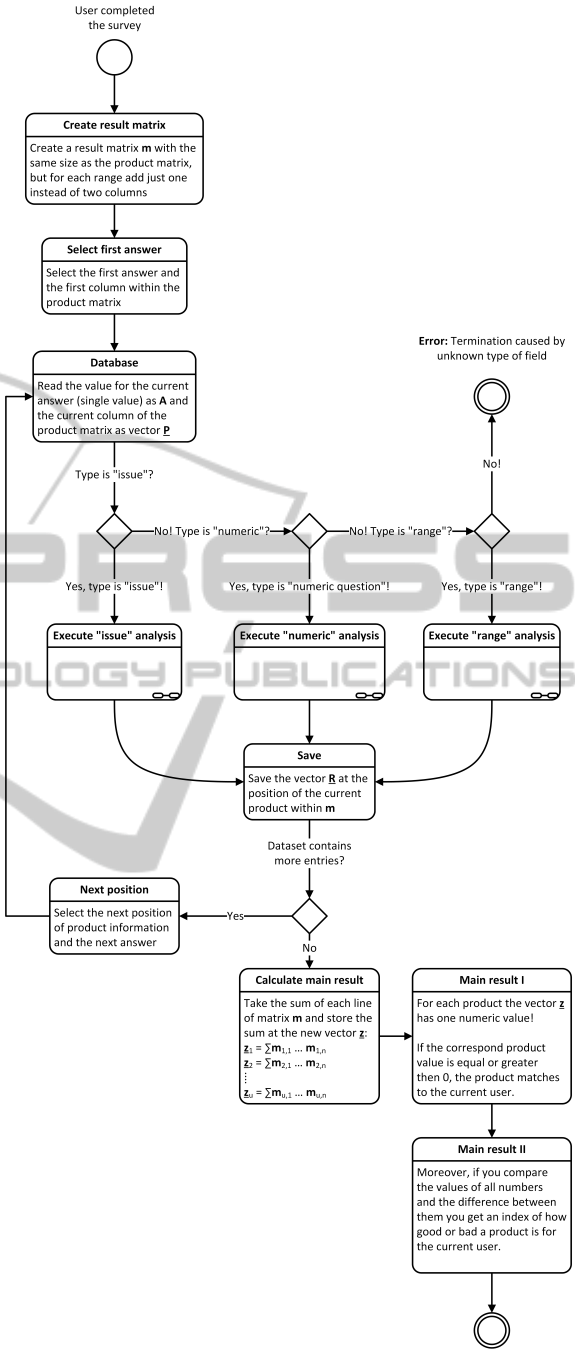

Figure 2: The recommendation algorithm to suggest e-learning systems.

[4]http://www.freebsd.org
[5]http://www.nginx.org
[6]http://www.mongodb.org
[7]https://www.ruby-lang.org
[8]http://www.php.net
[9]http://www.sinatrarb.com

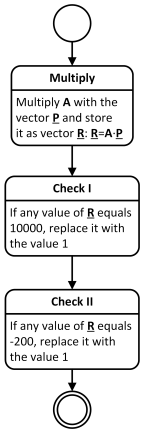

Figure 3: The sub-algorithm for the ”issue” analysis.

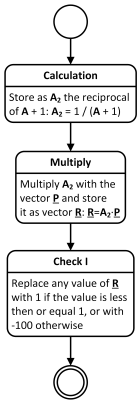

Figure 4: The sub-algorithm for the ”numeric question” analysis.

acteristics, e.g. a product function, product behavior or a didactic method, but not a numeric or range question (see section 3 for the differences). As input, the algorithm expects the static catalog of solutions as matrix: Horizontally the columns with the product issues, numeric questions and ranges etc. and vertically the rows with the products. The possible types of columns are corresponding to the available types of function blocks (see section 3).

Out of the product point of view for the product data: Each issue can obtain the value -100 (the issue is not present), 1 (the issue is present and cannot be disabled) or 2 (the issue is present and can be disabled).

A range, e.g. the possible amount of students (from/to), is provided as two columns inside the matrix. Moreover, numeric fields are possible for e.g. the price, which are provided as one column inside the matrix. At the moment, numeric fields are able to hold only positive numbers (include 0).

Another input for the algorithm is the teacher's answers, provided as vector. Out of the teacher's point of view: Each issue can obtain the value -100 (issue is not present or can be disabled), 0 (issue does not matter) or 1 (issue must be present).

For each range, the answer can be a positive number (include 0) to represent e.g. the amount of students – or -100 if this does not matter. Finally for each numeric field, the answer can be a positive number (include 0) to represent e.g. the teacher's budget – or -100 if this does not matter.

The start point (see figure 2) is the end of the teacher's questionnaire. As first step, the result matrix $\mathbf{m}$ is created, with the same amount of rows as the product data and the nearly the same amount of columns (but for each range only one instead of two columns).

Important to know: The first question from the questionnaire corresponds to the first answer and this corresponds to the first column within the product matrix and also to the first column within $\mathbf{m}$! Furthermore, a teacher receives might not all questions: The questionnaire is dynamic and user-adapting – the teacher gets only necessary questions. Any skipped question results implicit in the answer -100 which means ”does not matter”.

The next steps are repeated for each pair of an answer (the answer as a single value, here called $\mathbf{A}$) and corresponding column out of the product matrix as vector $\underline{\mathbf{P}}$.

The algorithm is now branching out by the column type (for now: issue, numeric or range). If an unknown type of column is found, the algorithm gets unexpected terminated. The sub-algorithms for the different types are found at figure 3 (type is ”issue”), figure 4 (type is ”numeric question”) and figure 5 (type is ”range”).

- In case of ”issue”, multiply $\mathbf{A}$ with $\underline{\mathbf{P}}$ and store the result as vector $\underline{\mathbf{R}}$: $\underline{\mathbf{R}} = \mathbf{A} \cdot \underline{\mathbf{P}}$. Check then, if any value of $\underline{\mathbf{R}}$ equals 10000. If so, replace it by 1. If any value of $\underline{\mathbf{R}}$ equals -200, replace it also with 1.
- If the type of the current column is ”numeric question”, store as $\mathbf{A}_2$ the reciprocal of $\mathbf{A}+1$: $\mathbf{A}_2 = \frac{1}{\mathbf{A}+1}$. Now multiply $\mathbf{A}_2$ with $\underline{\mathbf{P}}$ and store the result as vector $\underline{\mathbf{R}}$: $\underline{\mathbf{R}} = \mathbf{A}_2 \cdot \underline{\mathbf{P}}$. Replace all values of $\underline{\mathbf{R}}$ with 1 if the value is less or equals 1 or otherwise replace it with -100.
- The ”range” type is represented by two columns so instead of one $\underline{\mathbf{P}}$ this type has two vectors $\underline{\mathbf{P}}_{\text{min}}$ and $\underline{\mathbf{P}}_{\text{max}}$. If $\mathbf{A}$ equals -100, create the zero vector $\underline{\mathbf{R}}$ with the size of $\underline{\mathbf{P}}_{\text{min}}$ and this sub-algorithm is done. Otherwise, store as $\mathbf{A}_2$ the reciprocal of $\mathbf{A}+1$: $\mathbf{A}_2 = \frac{1}{\mathbf{A}+1}$. Next, multiply $\mathbf{A}_2$ with $\underline{\mathbf{P}}_{\text{min}}$, $\underline{\mathbf{P}}_{\text{max}}$ and store the result as $\underline{\mathbf{R}}_{\text{min}}$ and $\underline{\mathbf{R}}_{\text{max}}$. Replace now any element of $\underline{\mathbf{R}}_{\text{min}}$ with 1 if the value is

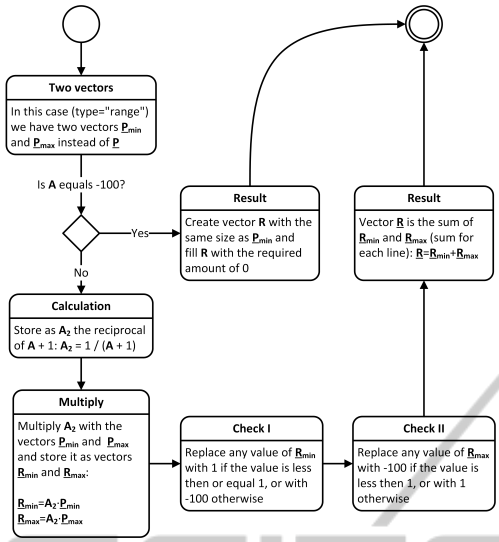

Figure 5: The sub-algorithm for the ”range” analysis.

less or equals 1, otherwise replace it with -100. Also replace any element of $\underline{\mathbf{R}}_{\max}$ with -100 if the value is less than 1, otherwise replace it with 1. The result vector $\underline{\mathbf{R}}$ is now $\underline{\mathbf{R}} = \underline{\mathbf{R}}_{\min} + \underline{\mathbf{R}}_{\max}$.

After the right sub-algorithm, the result vector $\underline{\mathbf{R}}$ must be stored at the corresponding position in $\mathbf{m}$. If not yet all pairs of answers and product data are executed, then select the next pair and repeat the steps above. If no pairs are left, take the sum of each line of matrix $\mathbf{m}$ and store the sum at the new vector $\underline{\mathbf{z}}$:

$$\begin{aligned} \mathbf{z}_1 &= \sum_{n=1}^{v} \mathbf{m}_{1,n} \qquad (1) \\ \mathbf{z}_2 &= \sum_{n=1}^{v} \mathbf{m}_{2,n} \\ &\vdots \\ \mathbf{z}_u &= \sum_{n=1}^{v} \mathbf{m}_{u,n} \end{aligned}$$

$$\underline{\mathbf{z}} = \begin{pmatrix} \mathbf{z}_1 \\ \mathbf{z}_2 \\ \vdots \\ \mathbf{z}_u \end{pmatrix} \qquad (2)$$

The variable v starts at 1 and grows up to the amount of issues, numeric questions and two times the amount of range questions. The variable u starts at 1 and grows up to the amount of products.

Interpretation of the results: The vector $\underline{\mathbf{z}}$ provides for each product one value. If a value is greater or equals 0, the corresponding product matches to the teacher's requirements. Moreover, if the resulting values are descending sorted, this new ordered list represents the teacher's best matching solution down to the worst solution[10].

6 RESULTS

The chosen software stack has already proved its power: The development time has been short and the source code is clear, with less overhead and a strong focus to web development. Thereby, the first prototype is a responsive web solution: This saves time, because the mobile devices are covered without an additional mobile app. The agile process made it possible to react on changes to the subject of e-learning and also to common project changes.

The first technical prototype with the visual programming language and the questionnaire is running without errors: It is possible to write a visual program and test the user flow through the resulting questionnaire. It is possible to change the visual program while users are inside the user flow of the questionnaire: The users in front of the changed function blocks are receiving directly these changes, and users behind the changed blocks are not affected at all. This feature enables to run a long-term system with no or less maintenance impacts.

The visual web-based editor for creating a visual program is convenient and enables also staff without computer science knowledge to create a visual program. The first version of the algorithm is passing all test cases with dedicated test data: The algorithm is working deterministic and the results are correct for any expected input. In case of the comparable characteristics for the related e-learning solutions, which are the precondition for this recommendation system, a first proof of concept exists.

7 CONCLUSION AND OUTLOOK

To meet the precondition of comparable characteristics for related e-learning solutions it is promising to get the input of experts to reach a broad scientific consensus. Because it was possible to build a proof of concept for the characteristics, it is confident to meet this precondition. The recommendation algorithm needs further research to investigate the performance under real conditions with a huge solutions catalog and also huge amount of answers. Therefore,

[10] If the value is less than 0, the product is of course not a ”solution” for this teacher.

the algorithm must be implemented into the prototype.

The new visual user-flow language enables to build user-adapting questionnaires. This saves processing time for the teachers, because their get just the necessary parts of the whole questionnaire. Nevertheless: The user-flow program is able to cover the complexity from unexperienced to advanced teachers, regarding to the e-learning subject.

In the long-term view: If the web-based recommendation system goes online, the service enables teachers to save time and let them focus to the engineering education. Later, the recommendation system can be expanded to other disciplines, beyond engineering education. After the project reaches a more mature stage, it will be accessible as open source under the 2-clause BSD license to enable others to use and modify it.

REFERENCES

Boicea, A., Radulescu, F., and Agapin, L. (2012). MongoDB vs oracle database comparison. In *2012 Third International Conference on Emerging Intelligent Data and Web Technologies (EIDWT)*, pages 330–335.

Cebeci, Z. and Tekdal, M. (2006). Using podcasts as audio learning objects. *Interdisciplinary Journal of E-Learning and Learning Objects*, 2(1):47–57.

Chodorow, K. (2013). *MongoDB: the definitive guide*. O'Reilly, Sebastopol, CA.

Crockford, D. (2006). The application/json media type for javascript object notation (json).

Dabkiewicz, S. (2012). Web server performance analysis.

Fenn, M., Murphy, M., Martin, J., and Goasguen, S. (2008). An evaluation of KVM for use in cloud computing. In *Proc. 2nd International Conference on the Virtual Computing Initiative, RTP, NC, USA*.

Flanagan, D. and Matsumoto, Y. (2008). *The Ruby programming language*. O'Reilly, Beijing; Sebastopol, CA.

Fowler, M. and Highsmith, J. (2001). The agile manifesto. *Software Development*, 9(8):2835.

Harris, A. and Haase, K. (2012). *Sinatra up and running*. O'Reilly Media, Sebastopol, CA.

Hils, D. D. (1992). Visual languages and computing survey: Data flow visual programming languages. *Journal of Visual Languages & Computing*, 3(1):69–101.

Kamp, P.-H. and Watson, R. (2004). Building systems to be shared, securely. *Queue*, 2(5):4251.

Kamp, P.-H. and Watson, R. N. (2000). Jails: Confining the omnipotent root. In *Proceedings of the 2nd International SANE Conference*, volume 43, page 116.

Kuster, J. (2011). *Handbuch Projektmanagement*. Springer, Berlin [u.a.].

Kutner, J. (2012). *Deploying with JRuby: deliver scalable web apps using the JVM*. Pragmatic Bookshelf, Dallas, TX.

Mayer, R. E. (2003). ELEMENTS OF a SCIENCE OF e-LEARNING. *Journal of Educational Computing Research*, 29(3):297–313.

McAuley, A., Stewart, B., Siemens, G., and Cormier, D. (2010). *Massive Open Online Courses. Digital Ways of Knowing and Learning. The MOOC Model for Digital Practice.*

Mohorovicic, S. (2013). Implementing responsive web design for enhanced web presence. In *2013 36th International Convention on Information Communication Technology Electronics Microelectronics (MIPRO)*, pages 1206–1210.

Nedelcu, C. (2013). *Nginx http server 2nd edition.* Packt Publishing Limited, Birmingham, UK.

Niessen, B. (2012). *Der eigene Server mit FreeBSD 9 Konfiguration, Sicherheit und Pflege*. dpunkt verlag, Heidelberg.

Nutter, C. O. (2011). *Using JRuby: bringing Ruby to Java*. Pragmatic Bookshelf, Raleigh.

Poppendieck, M. and Cusumano, M. (2012). Lean software development: A tutorial. *IEEE Software*, 29(5):26–32.

Rosenblum, M. (2004). The reincarnation of virtual machines. *Queue*, 2(5):3440.

Sangr, A., Vlachopoulos, D., and Cabrera, N. (2012). Building an inclusive definition of e-learning: An approach to the conceptual framework. *The International Review of Research in Open and Distance Learning*, 13(2):145–159.

Schoen, S. and Ebner, M., editors (2013). *Lehrbuch fuer Lernen und Lehren mit Technologien*. Second edition.

Wei-ping, Z., Ming-xin, L., and Huan, C. (2011). Using MongoDB to implement textbook management system instead of MySQL. In *2011 IEEE 3rd International Conference on Communication Software and Networks (ICCSN)*, pages 303–305.

Wolf, H. (2011). *Agile Softwareentwicklung: Werte, Konzepte und Methoden*. dpunkt, Heidelberg.