Universal Testing Environment as an External Tool of Moodle

Petr Voborník

Faculty of Science, University of Hradec Králové, Hradec Králové, Czech republic
petr.vobornik@uhk.cz

Abstract

The Universal Testing Environment is a sophisticated electronic cloud online test system, containing features that make it exceptional compared to other test systems. Moodle is a very widespread open-source LMS tool that has a many modules that can be used to implement a full-fledged eLearning teaching. This article will describe how these two systems were connected and tests in the Universal testing environment so they can be run directly from the Moodle course. This connection was realized through an External tool module, which is part of Moodle. Thanks to the Bridge design pattern then it was not necessary intervention into the code of any of both systems. This "bridge" translate LTI (Learning Tools Interoperability) protocol, which communicates Moodle, to the URL in format required by application interface (API) of Universal testing environment. This functionality is currently used at the University of Hradec Králové and the high school Podorlické vzdělávací centrum in Dobruška.

Keywords

Test. Universal Testing Environment. Moodle. External tool. LTI. Application interface. Cloud.

INTRODUCTION

The both linking systems in more details, especially the less known of them will be firstly presented. Its comparison with other test system can be found in Voborník (2011b).

Universal Testing Environment

Universal Testing Environment (hereinafter referred UTE) is an electronic online testing system designed for the creation, operation and administration of the tests. User part of the application (testing and administration interface) is created as the *Rich Internet Application* (RIA) at the *Silverlight*[2] technology. The system originated as a dissertation of the same name (Voborník, 2012b). (Voborník, 2012c). Similar solution for remote contact learning method can be found e.g. in (Hubálovský, 2012; 2013; Milková, 2012).

[2] Silverlight is a software plugin for development lavishly furnished internet applications that run within a web browser. It is developed by Microsoft, executed using the plugin which is a smaller version of the .NET framework and written in various languages supported by .NET (e.g. C#). (Lammarsch, et al., 2008)

ISBN: 978-80-7478-497-2

UTE approaches original way on the issue of electronic testing, both in terms of used technology an in terms of solution of sub-problems. RIA achieves convenience of desktop applications as well as it keeps the advantages of could computing. Any web browser and installed plugin Silverlight 5 is sufficient for using UTE, nothing else needs to be installed on the client computers and the server of organization. (Voborník, 2012c)

The tests consist of individual questions in UTE similar as in the other test systems. However here each question is treated as a separate component. In order the authors were able to fully utilize their creative potential and they needn't limit themselves to a set of predefined types of questions, own language named **QML**[3] was created. This allows to integrate into the questions arbitrary graphics, animation, active and randomly generated objects. Each question is so the independent the graphical and functional unit, which could cover completely thematic area of the curriculum if the random elements are appropriate used. (Voborník, 2011d)

Each question can be any **graphic** processed. This can be not only by embedded raster images, but QML directly supports vector rendering of standard graphical elements as line, rectangle, ellipse, polygon, path, etc. (see Figure 1). The code for the definition of standards was inspired SVG[4] and XAML[5]. (Voborník, 2011b)

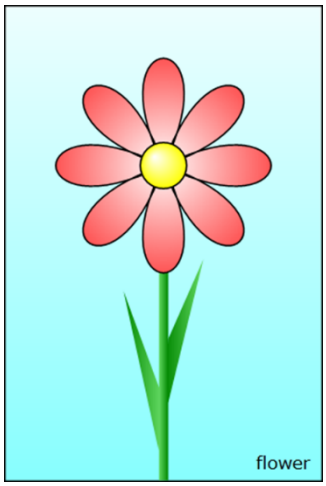

Figure 1: Sample of vector graphics by QML

Transformations are also supported. They allow to objects or groups of object located on the positional elements (containers) change the scale or rotation. **Animations** can be understand as one of the advantages which the test can never have on the paper. Therefore QML supports it too. Currently it is possible to animate to scale, rotation, movement, transparency and colour. (Voborník, 2011d)

In addition to static (or animated) graphical objects QML also support the **active elements**. They provide interactivity to the tested users, i.e. the possibility to intervene into the question and do something as them answers. There are several types of active elements, and each of them has different properties:

- **Switch** - it have several versions of its appearance (content) which is cycled changed by mouse clicking. The challenge is to choose the right variant. Among other things it can to replace the checkboxes or radio buttons.
- **Sorting** – this element contains a set of items (text or graphic) which are shuffled in the question and the tested user has the task to organize them into the correct order by using drag&drop[6] method.

[3] QML – Questions Markup Language, based on XML (Voborník, 2011b)
[4] SVG – Scalable Vector Graphics (Eisenberg, et al., 2002)
[5] XAML – Extensible Application Markup Language (Macvittie, 2006)
[6] drag&drop is a method in which the objects are moved on the desktop by the mouse so that the cursor are placed above the object, the object are "grabbed" by pressing the mouse button, then the object are dragged to the desired position, where it is "released" so that the pressed mouse button are released (Voborník, 2012b)

ISBN: 978-80-7478-497-2

- **Text input** – the edit box into which the tested user has the task to type the correct text answer.
- **Combo box** – it is an openable box with many options from which the tested user has to the task to choose the right variant.
- **Items and targets** – the items can be moved over the entire question's area by using drag&drop method. The task is to place them into the correct targets (see Figure 3a), i.e. areas into which items "bog down" by defined rules (Figure 2).

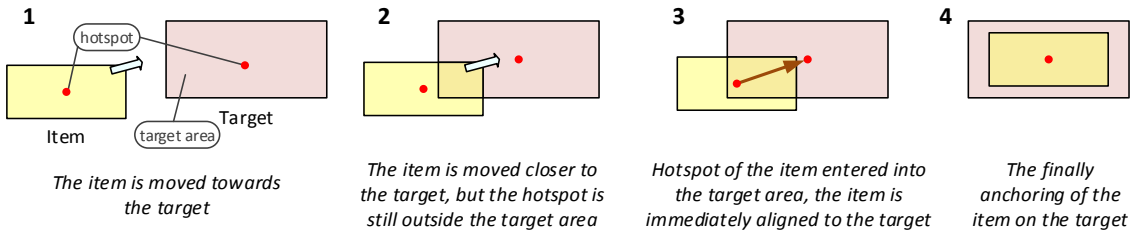

Figure 2: Illustration of anchor item to the nearby target (Voborník, 2012b)

The **random elements** are most extraordinary at UTE. They allow in a question generate random numeric values and also texts (see Figure 3b). Both can then be further combined in internal calculations, and commands for branching, cycles, etc. Thanks to the questions can be used repeatedly without risk that the tested user automatically interpret the solution without the knowledge of curriculum included in question. (Voborník, 2012c)

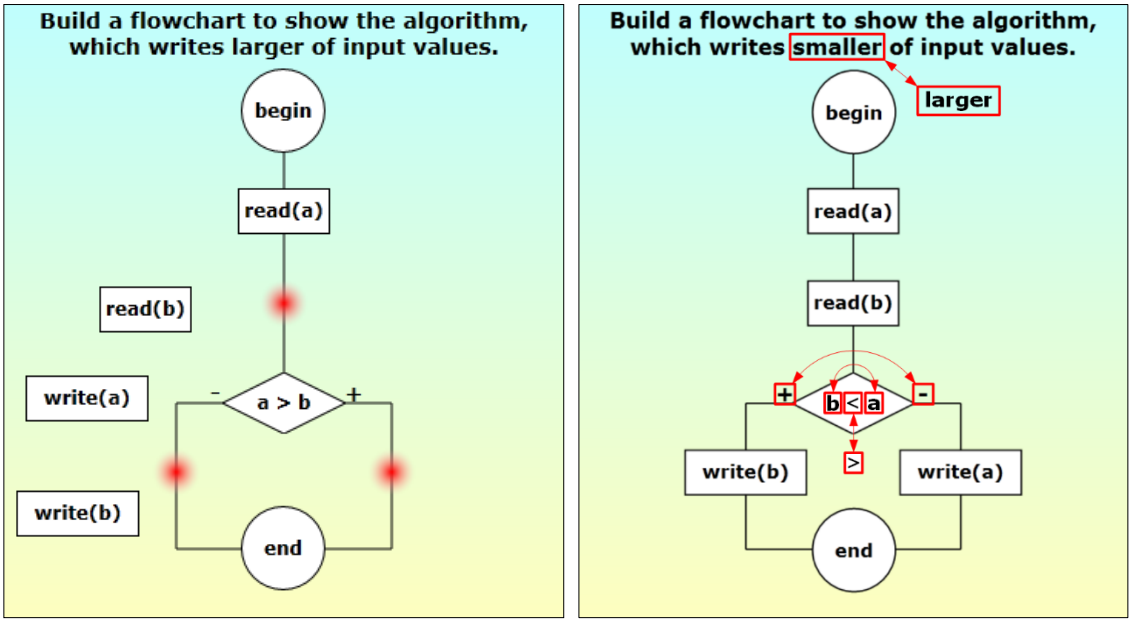

Figure 3: Sample of the question with the items and the targets in a default state (left) and resolved state (right), where is highlighted the parts that are randomly changed in the task and how (Voborník, 2011c, 2011d)

Automatic assessment is also sophisticated. For each part of each active element can be set to any **fuzzy score scales** for evaluation. The scope and scale can be chosen arbitrarily for each question. The system itself calculates the maximum possible score while the questions are generated, and the obtained score divides by this maximum, thus resulting score is converted back into a single range <0, 1>. Similarly works well overall

ISBN: 978-80-7478-497-2

evaluation of the test, where the result is the weighted average of the results of individual questions. (Voborník, 2011d)

For the evaluation of written text answers is also available possibility to determine the **degree of similarity** of the answer with the pattern. For this purpose, UTE has implemented 18 algorithms, which according to various aspects can express similarity between two texts as a percentage. Recognition of score in the question does not need to be subject only to absolute exact wording of the answers, but a certain tolerance may be allowed (e.g. typos or other form of the word). It is also possible to gain score value directly derived from the percentage of similarity between the two texts.

In addition to the evaluation of the test that accurately determines the percentage result, UTP can also provide detailed **analysis of the test**. Here is at all active elements indicated whether they were resolved correctly, partly correctly or completely wrong. For each variant, then can be also add additional information to further explain the obtained evaluation (see Figure 4).

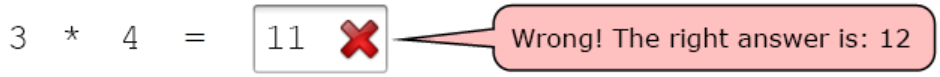

Figure 4: Example of automatically generated feedback

QML allows you to create graphically stunning and functional extensive questions, however taxes for this versatility is sometimes unnecessarily lengthy code. In order the same code in a recurring type of question was not necessary to write (copy) ever again, the ability to define an arbitrary transformation **template** (XSLT) was added. This from a brief entry in a few trivial XML elements generates a comprehensive and fully functional question in QML with all appurtenances (see Figure 5). (Voborník, 2011d)

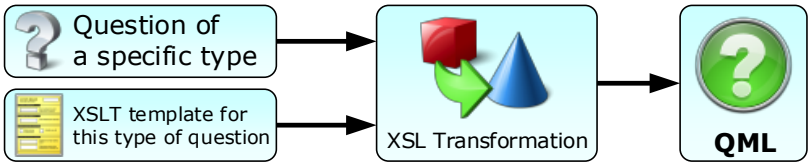

Figure 5: Scheme of the transformation simplified XML code of the question into the QML using the XSLT (Voborník, 2012c)

Questions of the specific tests before each launch may not only **mix**, but also randomly questions are **selected** from a larger database divided to the multilevel hierarchical structure. In this case, can be activated function for the **smart choice** of the questions. This function tries to maximize learning and testing effect of the test when it is launched repeatedly by the same user. This is achieved by an algorithm based on evolutionary principle which takes into account the previous results of the test in each of the possible questions. This principle was described in (Voborník, 2012a).

There is also a **protection against copying** from the internet or other applications. If the browser window or the current browser tab switched during test, the test is covered by a red panel and countdown of the automatic termination is launched. It can be stopped only by clicking on the *Continue* button. Time for this countdown and the number of tolerated switch can be set for the each test separately. (Voborník, 2011b)

ISBN: 978-80-7478-497-2

UTE can be used independently, or it can be interfaced with other systems via the **application interface** (API). UTE can be integrated into any web application, including modular or open-source LMS[7] and through the component *WebBrowser*, in to desktop applications too. The condition is that the systems used the UTE, must comply with the communication interface.

Moodle

Moodle (Modular Object-Oriented Dynamic Learning Environment) is a very popular learning management system, provides a highly configurable web-based interface that includes a wide range of activities which are, in general, sufficient for a standard course. (Corbera, et al., 2008)

Moodle includes module for creating, launching and assessment tests already in the basic installation (Voborník, 2014). These tests, although they are very good, they lack several important features that are commonplace in UTE (see previous section). The great advantage would therefore be, if it were possible to link these two systems and the Moodle courses could include the UTE tests.

Because the Moodle is open-source, so there were more possibility of linking. One of possibilities was edit the Moodle code directly, also could be to create a custom module, or add interface SCORM standard to the UTE, or use an existing module the **External tool**. The last option has proved to be most effective, both in terms of workload adjustments, as well as from user's view. External tools can be easy added to the course by any teacher without special privileges or assistance by administrator.

External tools in Moodle

External tool is therefore activity module in Moodle LMS. This can be added to any topic in any course same as other activity modules.

This module allows using an external third party tools which need not be included in the Moodle or its module. Just only the appropriate application interface that communicates based LTI[8] protocol (see Figure 6) and can thus be used as other integrated Moodle activities. (Voborník, 2014)

[7] LMS – Learning Management System, also in Šedivý and Hubálovský (2012)
[8] LTI – Learning Tools Interoperability

ISBN: 978-80-7478-497-2

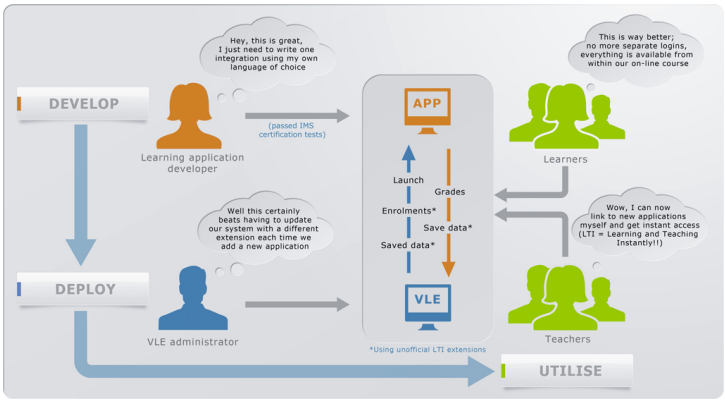

Figure 6: Scheme of the LTI protocol[9]

The basis of this communication is the URL address on which user is redirected by clicking on the name of the activity in the topics. It send by POST method to the external tool a hidden data of the user, such as the ID, login or e-mail address. It also includes authentication information such as a unique key (salt) for the communications, consumer key for identification of the application (Moodle) and checksum (hash) of whole message including the secret consumer password to validate the entire requirement.

Based on these data, the external on-line application can access some its operations, which the student executes and then by a similar way may send a "signed" data (e.g. grade) back into the Moodle. (Voborník, 2014)

IMPLEMENTATION OF LTI PROTOCOL IN UTE

Moodle sends data to the external tool such as a list of parameters and their values by the POST method, i.e. invisibly to the user. For this communication is of particular interest (inter alia) the following parameters:

- oauth_nonce – a unique random code for the requirement (salt)
- oauth_consumer_key – organization identifier (login)
- user_id – ID of the user that is registered in the Moodle
- lis_person_name_full – full name of the user in the Moodle
- lis_person_contact_email_primary – e-mail address of the user in the Moodle
- lis_outcome_service_url – URL address to which can be send back any data to the Moodle
- launch_presentation_return_url – return URL
- oauth_signature – signature (hash) of the whole message for request authentication; the hash is counted with secret password that is not included in the message

Moodle also allows to the setup of an external tool to add additional custom parameters that are sent to the application with the prefix "custom_" (e.g.

[9] http://www.imsglobal.org/toolsinteroperability2.cfm

ISBN: 978-80-7478-497-2

"custom_test=prg1"). These parameters allow preparation UTE support of several different mutually combinable type of use:

- **User login**
  - *Full-featured log in* – the user has already created and verified account in the UTE, he is a member of the organization under which this approach is established and he allowed self-login through API for this organization. It is important that the email address of the accounts in both systems (UTE and Moodle) was the same, because the user is identified and logged in by this address.
  - *Fictitious user* – users do not need to be registered in the UTE, for access is established a fictional user account under which are logging in and tested all students. In this case, the user is not be found by the email address, but according to the identifier specified in the additional parameter. In this case the email address is stored for each record of the testing in an optional parameter as application identifier. According to it, they are also filtered list of already finished tests for these fictitious users, so everyone sees only his own.
- **Launch application**
  - *Only logged in* – the user is only automatically logged into the system (and into the organization) and an overview of available or previously carried out tests are displayed to him. Here he can run a specific test (if it is currently available), or he can show the analysis of previously realized tests. After the test he is switched back to this overview.
  - *Launch test* – the user is logged into the UTE and the specific test is launched immediately. This test is intended by the identifier in the custom parameter for an external tool. After its completion, respectively after seeing its evaluation and analysis (if enabled), the user is logged out and redirected back into the Moodle.

All these variants of login in the UTE has supported earlier, but only through specific application interface (Voborník, 2012b, p.72). LTI protocol used in Moodle, however contains a different format for communication (e.g. POST method instead of GET, different parameter names, different calculation of authentication hash, etc.). Still, the best variant was to use of the existing API system also for LTI protocol.

For this purpose, was added special intermediate step to the UTE, respectively application webpage that accepts an incoming message LTI (POST), decodes the data and verify its validity. From these data its draw and sign a URL containing the necessary parameters (GET) for an already functioning application interface and it redirects the user on this URL (see scheme on Figure 7). Everything takes place in a fraction of a second, and the testing interface or the specific test is launched to the user, without he can note this step. Everything is maximum security with a disposable valid checksums (hash).

Parameter *oauth_nonce*, containing in each request another string of unique random characters, is used as the *salt* of each request. This key UTE save in the database and checks its uniqueness. This makes each generated URL usable only once (Voborník, 2011a and 2011e). Its validity is also limited to a few seconds, which makes its abuse almost impossible.

ISBN: 978-80-7478-497-2

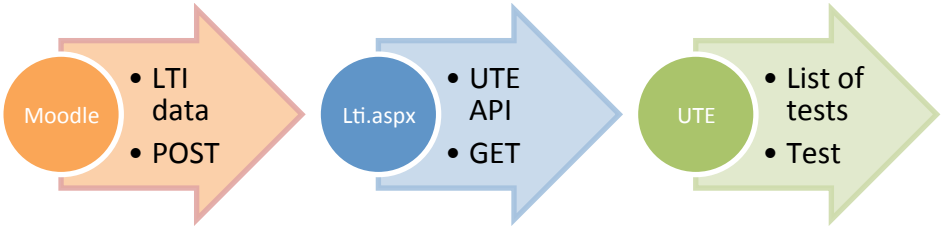

Figure 7: Diagram of sending data from Moodle to UTE

Thanks of creation an auxiliary site for translation of the LTI to the UTE API was not need to interfere with existing code of any of the two systems. Their communication is yet fully secured and allows all of the available variability. This approach of solutions basically applies the *Bridge* design pattern, described in (Bishop, 2007).

Linking UTE with Moodle via an external tool is very easy. Just in the administration of UTE set the test parameters and its availability by the classical way, and enabled access to test by secure API protocol. On Moodle side in the selected course is added external tool and a few parameters is set to (URL for launch, the organization identifier, password, and the parameters for run the UTE).

If there is no need to execute only the specific tests for specific course topics, just one external tool for the entire course is needed. Its only logs a student into UTE and shows him a list of tests, from which he can directly launch any of the currently available tests. In addition, here students can also see to the final analysis of the tests that they had written previously (if enabled).

USE IN PRACTICE

Support LTI protocol to UTE was added in April 2013 and it began to full use immediately. This occurred on the high school named "Střední Škola – Podorlické vzdělávací centrum, Dobruška" (SŠPVC) and on the University of Hradec Králové (UHK).

Střední škola – Podorlické vzdělávací centrum, Dobruška

UTE is used to separately on the SŠPVC before it. There all students were registered their user accounts, where they had a records of all previously passed tests. It was therefore desirable to have their history of tests remained continue available for them. So they can analyses their older tests from previous years to learn for graduation exam for example.

In this case, the approach with a full user login to UTE was provided only. Users, after logging into the Moodle and click on the appropriate action (external tool), has to directly displayed the list of tests in the UTE. From these test they will themselves chose and launch that with which should they work currently.

SŠPVC since February 2012 also uses Office 365 for school where all the staff and students have set up email box with address in a consistent format under the domain of school (surname.firstname@sspvc.cz). This address, which is everybody mandatory registered in the Moodle, is uniquely identifier for each in a form that everyone can easily read. Thanks this was made it possible for the following new classes (1st year in the school

ISBN: 978-80-7478-497-2

year 2013/2014) skip registration in UTE and for access to use only one fictitious user which is differentiate only by email addresses.

University of Hradec Králové

The UTE on the **UHK** was until then used only for unregistered auto tests. The UTE was not used for a rated testing, primarily because was need to register of each user to the system. Given that UHK as a possible support for learning uses the Moodle (http://kurzy.uhk.cz), the implementation of the LTI protocol to the UTE this problem was solved.

UHK is using a similar format of email addresses (firstname.surname@uhk.cz) a long time ago, so it could be access through a fictitious user put into operation immediately. Tests in the UTE accessed through the Moodle have become part of the test of the subjects in Applied informatics 1 and Programming 1-3 (see Figure 8) and also plans to deploy them in other subjects.

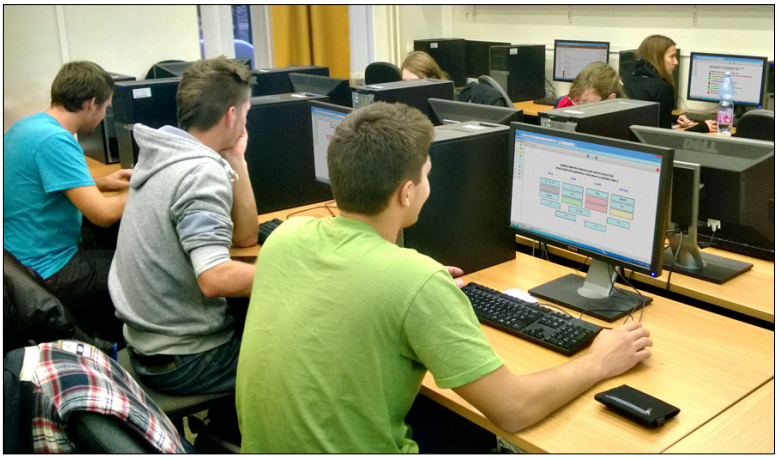

Figure 8: The exam of the subject Programming 3 via UTE

CONCLUSION

Universal testing environment provides a many features that no other electronic test system have. Moodle is a very widespread open-source LMS tool that has a many modules that can be used to implement a full-fledged eLearning teaching. These two independent systems succeeded in linking for cooperate. Extensive Moodle functionality can be extended by the use of tests developed and run in UTE using all its benefits.

Adding the tests prepared in the UTE into the Moodle is due to an external tool module so easy that it can handle any course creator himself without having to install anything else into the Moodle. Implementation of support to this functionality was realized without having to change existing code of the Moodle or the UTE. It was enough to just add a simple compiler of LTI protocol to the UTE API.

Only one-way communication is supported in the current version, i.e. from the Moodle to the UTE. In the future, it should be also added support for reverse flow of information, i.e. the possibility that the UTE sent data from such evaluation of the completed tests back to the Moodle.

ISBN: 978-80-7478-497-2

UTE may be used equally with any other LMS that supports the LTI. Through UTE was carried 7,642 tests until 1st February 2014.

ACKNOWLEDGEMENT

This paper is published thanks to the financial support of the project "VeNaDo".

REFERENCES

Bishop, J., 2007. *C# 3.0 Design Patterns*. Sebastopol: O'Reilly Media. ISBN 978-0596527730.

Corbera, F., Gutiérrez, E., Ramos, J., Romero, S., Trenas, M. A., 2008. Development of a New MOODLE Module for a Basic Course on Computer Architecture. *ACM SIGCSE Bulletin*, New York: ACM, 40(3). ISBN 978-1-60558-078-4.

Eisenberg, J., 2002. *SVG Essentials*. Sebastopol: O'Reilly Media, ISBN 978-0-596-00223-7.

Hubálovský, Š., 2012. Research of Methods of a Multidisciplinary Approach in the Teaching of Algorithm Development and Programming. In*: 9th International Scientific Conference on Distance Learning in Applied Informatics (DIVAI)*. Nitra: Univerzita Konštantína Filozofa.

Hubálovský, Š., 2013. Remote Contact Learning of Programming in Distance Study. In: *10th International Conference Efficiency and Responsibility in Education (ERIE)*. Praha: Česká zemědělská univerzita.

Lammarsch, T., Aigner, W., Bertone, A., Miksch, S., Turic, T., Gärtner, J., 2008. A Comparison of Programming Platforms for Interactive Visualization in Web Browser Based Applications. In: *International Conference Information Visualisation*. Washington DC: IEEE Computer Society. ISBN 978-0-7695-3268-4.

Macvittie, L.A., 2006. *XAML in a Nutshell*. Sebastopol: O'Reilly Media. ISBN 9780596526733.

Milková, E., 2012. Multimedia Applications – Effective Support of Education. In: *9th International Scientific Conference on Distance Learning in Applied Informatics (DIVAI)*. Edition: Prirodovedec, no. 500, Faculty of Natural Sciences UKF, Nitra, pp. 13–21.

Šedivý, J., Hubálovský, Š., 2012. Mathematical foundations and principles in practice of computer aided design simulation. *International journal of mathematics and computers in simulation*, North atlantic university union, pp.230–237. ISSN: 1998-0159.

Voborník, P., 2011a. Bezpečná autentizace aplikace klient-server v internetu pomocí povinně unikátních saltů. In: *Internet, bezpečnost a konkurenceschopnost organizací*. Zlín: Tomas Bata University in Zlín, pp.347–354. ISBN 978-80-7454-012-7.

Voborník, P., 2011b. Počítačové testovací systémy. In: *Alternativní metody výuky*. Praha: UK, [online] Available at: <http://everest.natur.cuni.cz/konference/2011/prispevek/vobornik_prispevek.pdf> [Accessed 1 February 2014]. ISBN 978-80-7435-104-4.

Voborník, P., 2011c. Teaching algorithms using multimedia tools. In: *Efficiency and Responsibility in Education (ERIE)*. Prague: FEM CULS, pp.312–321.

Voborník, P., 2011d. Univerzální testovací prostředí. In: *Konference eLearning 2011*. Hradec Králové: Gaudeamus UHK, pp.80–85. ISBN 978-80-7435-153-2.

Voborník, P., 2011e. Rychlá multiplatformní autentizace v internetu. In: *IP Networking 1 – Theory and practice (zborník vedeckých prác)*. Žilina: University of Žilina, pp.41-45. ISBN 978-80-554-0494-3.

Voborník, P., 2012a. Výpočetní model inteligentního výběru testových otázek. In: *14th international conference (MEKON)*. Ostrava: VŠB – Technical University of Ostrava. ISBN 978-80-248-2552-6.

Voborník, P., 2012b. *Univerzální testovací prostředí*. Ph.D., University of Hradec Králové, [online] Available at <http://download.petrvobornik.cz/docs/disertace.pdf> [Accessed 1 February 2014].

Voborník, P., 2012c. *Univerzální testovací prostředí*. Web, [online] Available at <http://www.alltest.eu> [Accessed 1 February 2014].

Voborník, P., 2014. *Základní moduly činností v Moodle*. Hradec Králové: University of Hradec Králové, [online] Available at <http://download.petrvobornik.cz/docs/knihy/moodle.pdf> [Accessed 1 February 2014].