Adaptive Testing in E-Learning – the Veval Approach

David Ahlström[1], Martin Hitz[2], Karin Hodnigg[3]

University of Klagenfurt, Department of Informatics-Systems

Key words: *web based examination, self assessment, test generation, test evaluation*

Abstract:

This paper presents Veval, an Internet based assessment tool under development. It is designed for integration into Velo, a virtual electronic laboratory system. Veval helps to utilize the manifold possibilities of self assessment and examination via Internet. It provides a test generating module for the tutor and an examination or self assessment environment for students as well as a component responsible for statistical evaluations. Based on an analysis of several existing assessment tools, the requirements for Veval are discussed. An overview of the flexible and portable system architecture and a short presentation of the graphical user interface are provided.

1 Introduction

Why considering an online assessment tool? In [1], McCormack and Jones present several advantages of using a web based assessment tool. Next to aspects of time (time to prepare a test as well as time to evaluate it are remarkably reduced) and convenience, the reductions of costs and resources are pointed out as major benefits. Moreover, the information gained by executing a test can be used to further tailor and adapt the contents of a course (electronic or traditional) according to students' needs. Nevertheless, there are some critical factors for success: The assessment environment has to provide a thread running through the whole assessment process. Basically, students have to be prepared for the application of such a tool – both comprehensiveness and a well-defined feedback are necessary for acceptance among students. Also, a comfortable and concise authoring tool to generate tests has to be available for tutors.

Based on an analysis of several commercially available assessment tools, this paper presents the concepts behind Veval (Velo Evaluation Tool), an assessment tool under development. Veval aims to help utilize the manifold possibilities of self-assessment and examination via Internet. It provides both, a test generating module for tutors and an examination or self-assessment environment for students.

Veval is a part of a larger e-learning system developed in the Velo (Virtual Electronic LabOratory) project [9], funded by the Austrian government. Velo was initiated to improve the situation at technical universities and universities of applied sciences. For students of engineering sciences, laboratory exercises are important in order to transfer acquired theoretical knowledge into practical skills and thus to obtain a more profound knowledge. Unfortunately, the setup, administration and usage of educational laboratories are often very cost and time expensive. Furthermore, they can only be used by a limited number of students under a limited time and therefore laboratory time becomes rare and treasured. The Velo project aims to improve this situation by developing an "electronic laboratory", which can be used any time and anywhere from an Internet access point. By simulating electronic experimental equipment, Velo makes it possible to substitute certain real laboratory exercises. Although Veval is designed to primarily meet the requirements for assessment in the area of electronic engineering, the design is made flexible so that adaptations to other areas are possible.

The following section presents a brief overview of some available assessment tools and their functionality. In Section 3, the most important functional and non-functional requirements of Veval are described. Section 4 deals with implementation issues and system architecture and also presents the graphical user interface. The paper ends with a brief description of the future work to do.

2 Survey of Existing Assessment Tools

E-learning and thus online testing have not yet undergone a standardisation process. So a variety of different tools and applications with different features, different source and destination applications and different benefits is available. As the research area of e-learning emerges, we are confronted with very different ways of assessment and evaluation of a student's progress. Tutors as well as their students prefer a variety of methods and techniques. Consequently, there are many applications and e-learning suites that try to meet these requirements. The following presentation of some available assessment tools is extracted form an analysis conducted at the beginning of the development of Veval in fall 2001, as a part of the specification process and serves as an overview.

Hot Potatoes [3] is an authoring tool to create different types of interactive web-based exercises with underlying concepts like HTML and JavaScript. This tool consists of six different applications, which can provide six different types of questions. To the disadvantage of this tool, the only feedback provided is an e-mail message either to the student or the tutor. A question can be answered multiple times – with no adverse effects to the resulting score. Moreover, Hot Potatoes has the answers in the HTML/JavaScript source code accessable for the user - this tool is just for personal use and cannot be used as a serious test environment.

QuestionMark Perception [4], consisting of various applications, helps the tutor to develop and deliver tests. With its wizards, this tool does not require programming knowledge on the part of the tutor. A question can hold information in a variety of data types, including images, hyperlinks or multimedia. Moreover, QuestionMark Perception offers a very flexible feedback- and reporting unit. Since it offers a so-called *Perception Secure* Browser and a flexible session manager for adapting the tests to the author's needs, it is a very substantial product.

Macromedia Coursebuilder [5] allows the user to define a complete web-based learning interaction. It offers a so called *Action Manager* to subsume the results, a *Tracking System* with a *Knowledge Track* that supports the evaluation of a student's progress by offering valuable information about a test: the time needed to react and the time needed to answer a question, the number of trials etc. One can save the test – with all its additional information – in a database. Yet, programming knowledge is required to be able to take advantage of all its features.

WebCT [6] is an Internet course management software that also offers the possibility to create and attend tests in a password protected Internet environment. Three different test modes are supported: self test mode, quiz mode (which results in detailed reports and statistics for the tutor) and poll mode (which allows anonymous surveys). WebCT provides a flexible scoring system.

Quandary [7] is an application for creating action mazes, which are a kind of interactive case study. The user is confronted with a situation upon which he/she has to react and has to choose one of several given alternatives. Based on his/her decision, the next situation is presented, and so on. In this sense, only multiple choice questions which can be dependent on preceding question(s) are possible. The fact that this is the only supported question type considerably limits the scope of the application and makes it an inadequate tool for generating self assessment tests and quizzes with varieties.

e-Cadamy [8] is an emerging platform for creating tests online. It offers multiple possibilities regarding creation, execution and analysis of tests including several question types, with the options to include images, hyperlinks or other multimedia content. This environment supports the grouping and reuse of questions and tests, making it possible for every tutor to keep track of his/her data.

2.1 Deficiencies

The analysed tools are listed in Table 1 that compares them with respect to some characteristic features of assessment tools.

Features	***Hot Potatoes***	***Question-Mark Perception***	***Macromedia Course-builder***	***WebCT***	***Quandary***	***e-Cadamy***
authoring environment	stand-alone application	application	application/ Dreamweaver	browser	application	browser
examination environment	browser	browser/ application	browser	browser	browser	browser
types of questions	- MC[*] - text entry - mixed words - crossword - cloze - matching	- essay - cloze - hotspot - matrix - MC - numerical - matching	- various MC - drag'n'drop - button - slider	- MC - text entry - matching - short essay	- MC	- MC - hotspot - cloze - drag'n'drop
results/feedback	yes	yes	yes	yes	no	yes
persistence of results	no	yes	yes	yes	no	yes
statistics/analysis	no	yes	yes	yes	no	yes
tracking a student's progress	no	no	yes	yes	no	yes
user administration	no	no	yes	yes	no	yes
import of questions	no	yes	yes	no	yes	yes
security	none	high	medium	high	none	medium
group questions by (any) criteria	no	yes	no	no	no	yes
integration of an external database possible	no	yes	yes	no	no	no
flexible test generation (with randomized questions)	no	yes	no	yes	no	no
grading	static	flexible	static	flexible	none	flexible

[*] multiple choice question

Table 1: Comparative survey of web based assessment environments.

For a self assessment tool, a dynamic test generation would be appreciable, so a student can test himself multiple times on a subject, without being confronted with the same questions again and again, and without forcing the tutor define multiple tests. This demand results in the following requirement: it should be possible to group questions concerning the same topic and pose them at random. Moreover, the tool should support self assessment as well as examinations of students in a monitored test environment, so a very flexible and adaptable grading system should be offered. To ease the tutor's work, question sharing between tutors should be made possible. E.g. tutors define some questions concerning some topic, and they would probably like to exchange them among each other with minor effort. To be able to track a student's progress, some sort of user administration is needed. A tool used for self assessment as well as for examination should also provide a mechanism for data persistence. Having persistent data, statistical evaluation and reporting is possible.

3 System Requirements

The examined and previously described tools partly offer features which are required for an assessment and evaluation tool in the domain of electronics. Because none of them meets all of the requirements identified so far and detailed below and fulfil the desired functionality, the development of Veval was initiated. For sake of brevity, the requirements are listed below in indicative mood.

3.1 Functional Requirements

Veval is easy and intuitive to use for students as well as for tutors and provides a broad set of question types in order to create a diversified and stimulating environment that can be used in self-assessment mode as well as for supervised examinations. For clarification purposes, a picture can be added to all different question types. The question types required to emulate conventional (paper) exams and tests in the domain of electronics are:

1. multiple choice question,
2. numerical value question and
3. keyword matching question.

The numerical value question type requires a numerical value as answer. A numerical value question can be evaluated in either of two ways:

1. exact match (the submitted value has to exactly match the value specified as answer) and
2. deviation tolerance (at creation time, an admissible deviation range is specified, answers within this range are considered correct).

In its first application domain (electronic engineering), Veval is employed as part of Velo. Velo represents primarily technical and mathematical modules, so a specific variant of the numerical value question is provided which allows parametric definition of formulas with variables which are assigned a numerical value during run-time.

To ensure flexible, logical and efficient creation, storage, retrieval and re-use of questions, they are organised in question pools with several classification attributes. Question pools are sets of equivalent (but different) questions concerning a specific topic. The question pools are also used to generate random question combinations for individual tests. At test creation time, the tutor specifies a set of question pools from which the questions for each test are to be taken. For every individual test, at run time, a question from each one of the previously specified question pools is selected randomly.

A student carrying out a test, is presented with all the questions simultaneously (i.e., he/she can skip back and forth between the questions at will). The presentation of the correct answers and the resulting score can take place immediately after finishing a test. If prior at creation time of the test the tutor has chosen to grade the test, the grade, based on a scale predefined by the tutor, is presented as well. Since Veval is not going to be used for self-assessment purposes only but also for supervised examinations, the option of reconstructing the whole test event is necessary. For each individual test taken by a student, the randomly selected questions are saved together with the given answers and their scores. This allows for later reconstruction of the test event and also for statistical evaluations. The latter help the tutor to identify strengths and weaknesses of his or her students and subjects that might need additional elaboration. By providing detailed statistical visualisations, a well-defined feedback loop is established for both students and tutors. An additional and valuable source of feedback is the comments about a test that students can submit after having attended it.

The services provided by Veval can be summarised according to user roles as follows:

1. A tutor can
    - define individual questions,
    - define question pools,
    - assemble tests,
    - query test results (including predefined statistics),
    - reconstruct individual tests,
      and
    - notice comments from the students about a test.
2. A student can
    - attend tests,
    - query his/her own results,
    - query predefined statistics (aggregated information only) and
    - comment tests.

3.2 Optional and Non-functional Requirements

Functionality planned to be implemented in later versions of Veval include:

1. support for Internet Browsers other than Internet Explorer 6.0,
2. complementing the current German version of the graphical user interface with an English version,
3. a utility to import/export questions and their answers from/to other applications and
4. embedding the current statistical functionality into a more complex and expressive report generation component.

In its current version, Veval cooperates with HIS (“Hyperwave Information Server”) [2], an object oriented web server and is integrated into the HIS-based e-learning platform eLS (“e-Learning Suite”) [2]. But to meet the high prioritised requirement of high portability, the coupling to HIS and eLS is fairly loose, therefore, the implementation of Veval is based on a flexible three pillar architecture described in Section 4.

4 System Architecture

The implementation of Veval is based on three pillars: the Web based user interface, a database system as a persistence layer as well as an interface between the database and the Web GUI as sketched in Figure 1.

As Hyperwave and its e-Learning Suite have been chosen as software platforms for Velo, Veval's user interface needed to be integrated into the e-Learning Suite. Although this could be achieved in several different ways, we opted for a servlet-based Web GUI.

In order to keep data management as independent as possible from the persistence layer chosen, a middleware layer responsible for the communication between the GUI and the data management layer was introduced. It implements Veval's entire main functions as well as data transportation between the user interface and the data storage based on the Hyperwave Information Server. This layer ensures that

- a different persistence platform (in particular, a relational database system such as Oracle).
- a standard Web GUI (outside the Hyperwave eLS)

can be employed in the future.

With this architecture the system remains fairly flexible with regard to further use within different environments, to maintenance and to modification.

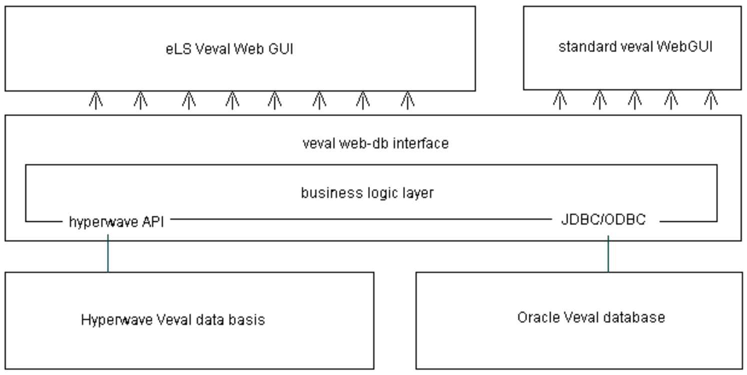


Figure 1 A simplified architecture of Veval.

4.1 Data Management

The central components of the Veval data model can be divided into three main parts:

- a test generation part, consisting of the question and its answer, the question pool and the test itself,
- a test execution part, which consists of evaluation records and the test configuration and
- a test analysis part, consisting of the concrete tests already executed by students and their answers, potential feedback and statistical information.

In Figure 2 we show an excerpt of the UML class model focusing on the test generating part.

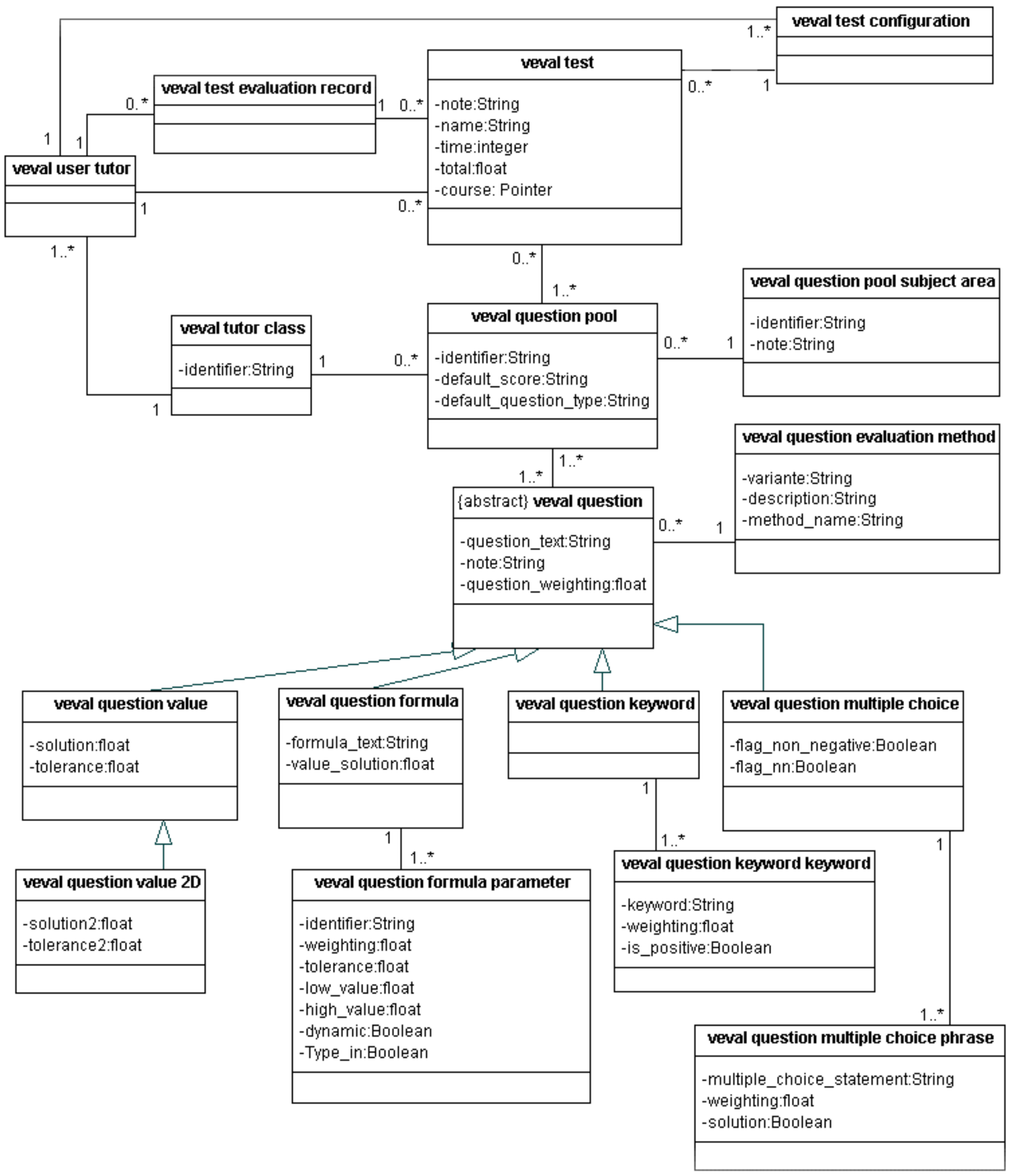


Figure 2 Extract from the conceptual data scheme.

A *test* is defined by a tutor. It holds parameters like configuration details or scoring information and authoring information of the test. In fact, test refers to a test frame, not to the executed test, which is represented by a class *test_completed*. *test_evaluation_record* consists of attributes describing the grading scale, *veval_test_configuration* holds configuration information for a test, e.g. flags which indicate the time the score is to be presented to the user, or if a feedback text field should be shown.

A test consists of various question pools, with a *question_pool* finally keeping the questions of a test. This class describes the basic information of a question pool, e.g. its name, its default question type and its default score. All questions belonging to one question pool have to have the same score, but the score itself can be changed (e.g. by multiplication), if the tutor decides to reduce a question’s significance in a test. A question pool is defined by a *tutor_class* that allows the sharing and reuse of questions among a well-defined group of tutors (and only among them).

Finally, a *question* is the object that holds the inquiry information and its solution. There are four different basic types of questions, numeric value, formula value, keyword and multiple choice questions. From the numeric value question type a subclass is derived to implement questions with imaginary numbers. Within a *question_value* the tutor can define a deviation range for the solution value that means a range in which the student's answer is still considered correct (or partly correct). A *question_keyword* consists of different keywords, their relevance and correctness (negative keywords are allowed). A *question_multiple_choice* consists of a set of phrases or statements (*question_multiple_choice_phrase*) whose correctness the student has to determine. A *question_formula* consists of in- and output parameters representing the mathematical variables used in the formula. At runtime, these variables are assigned numeric values within the specified range (between *low_value* and *high_value*).

On the Hyperwave Server, this UML scheme, as shown in Figure 2, has to be adapted to the Hyperwave Information Server object paradigm, which basically relies on the JavaScript object paradigm. There are three types of objects, which can be implemented: *collections* (quite the same as directories), *documents* (containing content like images, texts, presentations, etc.) and *objects* (which are mainly pairs of attribute names and their values). These types of objects can be enriched by parameters and attributes and entailed to the user's needs in so-called Document Classes. Document Classes allow the Hyperwave user to add own functionality to the Server. In fact, without going to deep into the implementation details, tests, question pools and questions are collections. So we can remain quite flexible, with new functionalities added just by re-implementing some functions of the interface without major changes of the data model.

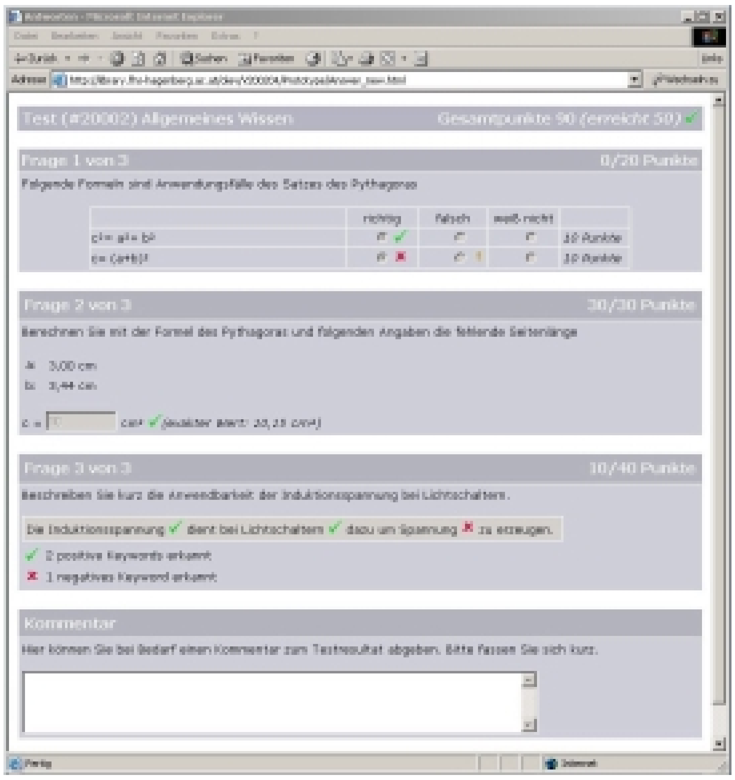
4.2 Middleware

As shown in Figure 1, the middleware mainly consists of two different modules: One provides access to the Hyperwave server, the other one access to an Oracle database. Since Veval should become portable, the decision towards JDBC/ODBC was made. The communication with the Hyperwave server is based on a software programmer's API, which allows object orientation throughout Veval.

The middleware – a Java implemented interface – gets the request form the web based GUI, processes it, puts or gets required data to/from the database, handles necessary transformations and finally returns the result data to the web interface. The business logic layer performs data manipulation e.g. the encapsulation of different information into an object needed by the Hyperwave server. It also implements the main functions of Veval such as test generation, user administration, analysis and grading of tests.

4.3 User Interface

The web based graphical user interface of Veval is implemented by using servlets. It consists of three main modules, an access module for user data, an administration module to provide an interface where a tutor can create tests, as shown in Figure 4, and a module responsible for test execution, shown in Figure 3. Since Veval is intended to be installed at Austrian technical colleges, the first (and current) version of the GUI is only available in German.



Test (#20002) Allgemeines Wissen — Gesamtpunkte 90 (erreicht 50) ✓

Frage 1 von 3 — 0/20 Punkte

Folgende Formeln sind Anwendungsfälle des Satzes des Pythagoras

	richtig	falsch	weiß nicht	
c² = a² + b²	● ✓	○	○	10 Punkte
c = (a+b)²	● ✗	○ !	○	10 Punkte

Frage 2 von 3 — 30/30 Punkte

Berechnen Sie mit der Formel des Pythagoras und folgenden Angaben die fehlende Seitenlänge

a: 3,00 cm
b: 3,44 cm

c = [illegible] cm ✓ (exakter Wert: [illegible])

Frage 3 von 3 — 10/40 Punkte

Beschreiben Sie kurz die Anwendbarkeit der Induktionsspannung bei Lichtschaltern.

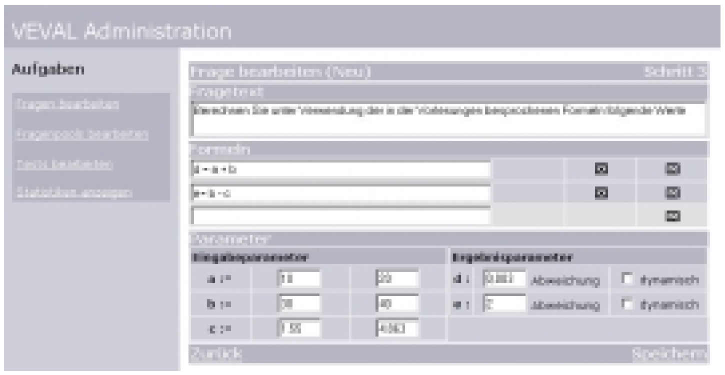
Die Induktionsspannung ✓ dient bei Lichtschaltern ✓ dazu um Spannung ✗ zu erzeugen.

✓ 2 positive Keywords erkannt
✗ 1 negatives Keyword erkannt

Kommentar

Hier können Sie bei Bedarf einen Kommentar zum Testresultat abgeben. Bitte fassen Sie sich kurz.

Figure 3: Graphical user interface for test execution.



VEVAL Administration

Aufgaben

- Fragen bearbeiten
- Fragenpools bearbeiten
- Tests bearbeiten
- Statistiken anzeigen

Frage bearbeiten (Neu) — Schritt 3

Fragetext: Berechnen Sie unter Verwendung der in der Vorlesungen besprochenen Formeln folgende Werte

Formeln:
d = a + b
e = b - c

Parameter

Eingabeparameter			Ergebnisparameter			
a :=	10	20	d :	0.001	Abweichung	☐ dynamisch
b :=	30	40	e :	2	Abweichung	☐ dynamisch
c :=	1.55	4.563				

Zurück — Speichern

Figure 4: Graphical user interface for test administration.

5 Future Work

The current version of Veval is based on Hyperwave Information Server and is integrated into the Hyperwave eLS. Future versions should provide an independent data source with preservation of the now implemented features such as user administration, content inclusion or information sharing.

In the next few months, Veval will be installed at the Carinthia Tech Institute in Villach. Results form the first test period will show how Veval can be optimised according to students' and tutors' needs. A bug fixing phase where all the software errors should be remedied is planned throughout this first real-time and real-situation test. We will also concentrate on performance and efficiency issues.

The integration of other databases than Hyperwave, primarily Oracle, and the implementation of this module of the middleware, as well as the implementation of some optional modules and features should take place at the same time to further improve Veval. The new modules include a language module, realizing a multilingual Veval, an import/export utility to easyly exchange or import questions or question pools. The current rather rudimentary statistically features, are to be widened by a more elaborate reporting unit, implementing complex and expressive reports.

Although there are several tools available for online assessment, it is hard to find suitable products that support all the necessary features, which make them adequate for extensive use in academic settings. The requirements that result in such an academic setting are particularly considered in the design of Veval. With its test authoring, its test execution and its statistic module Veval has the potential to satisfy tutors' and students' high demands.

Acknowledgement

The authors wish to thank all members of the Velo project team for their fruitful cooperation and Gerhard Jahn and Alois Reitbauer (who are responsible for implementing the user interface of Veval) for kindly providing screenshots of the current release of the GUI for this paper.

References:

[1] McCormack, C. and Jones, D.: Building a Web-based Education System, Wiley Computer Publishing, New York, 1998
[2] http://www.hyperwave.com/
[3] http://web.uvic.ca/hrd/halfbaked/
[4] http://www.questionmark.com
[5] http://www.macromedia.com/software/coursebuilder/
[6] http://www.webct.com/
[7] http://www.halfbakedsoftware.com/index.htm?quandary/
[8] http://www.e-Cadamy.at
[9] http://193.171.119.158:8080/velo/

Author(s):

1
Ahlström, David, Mag.
University of Klagenfurt, ISYS
Universitätsstraße 65-67, 9020 Klagenfurt
david@isys.uni-klu.ac.at

2
Hitz, Martin, Univ.Prof. Dipl.Ing. Dr.
University of Klagenfurt, ISYS
Universitätsstraße 65-67, 9020 Klagenfurt
martin.hitz@uni-klu.ac.at

3
Hodnigg, Karin
University of Klagenfurt, ISYS
Universitätsstraße 65-67, 9020 Klagenfurt
karin@isys.uni-klu.ac.at