DaskaL – a web-based application for foreign-language teaching

Kjetil Raa Hauge – University of Oslo
Svetla Koeva – Bulgarian Academy of Sciences
Emil Doychev, Georgi Cholakov – Plovdiv University

Abstract

The DaskaL web application is a framework for the creation and interactive use of foreign language exercises for beginners and advanced students. It supports the construction of different types of tasks: grammar drills; element order drills, and free-style drills (essays, dialogues etc.). Several options for interaction (keyboard entry, menu choice, single or multiple correct answers) are provided. Inflectional dictionaries may be plugged in to speed up development of exercises.

1 Introduction

The DaskaL web application is a framework for the creation and interactive use of foreign-language learning exercises for beginners and advanced students (http://daskal.net).[1] It is being developed with language data from Bulgarian, Serbian, Czech and Polish, but other languages can easily be added, as the system architecture is designed to be language-independent. DaskaL provides possibilities for educators to make several types of exercises and present them to students. The exercises may relate to different levels of language units – sound, morpheme, word, phrase, sentence, or text.

The application is based on three conceptual units labelled: task, exercise and test. A **task** is seen as the actual linguistic knowledge to be attained by an exercise. **Exercises** are the concrete realizations of the tasks and are the basic units of the system. A **test** is an ordered group of two or more exercises.

DaskaL offers several exercise patterns covering different types of tasks: **grammar drills**; **word order** (element order) **drills** (where scrambled units on different levels should be put into correct order); and **free-style** (semantic) **exercises** (essays, dialogues etc.). Different types of exercises may be combined into tests. The exercises can be designed to accept either type-in answers or a

[1] The DaskaL application is developed by the Department for Computational Linguistics at the Institute for Bulgarian language at Bulgarian Academy of Sciences. The project is financed by the Program for Cooperation between the Faculties of Letters at the University of Oslo and Gothenburg University, and by the Program for Flexible Learning at the University of Oslo.

selection from a menu.
When several exercises are combined into a test, they may be assigned varying point weights. The weights may also be assigned by default and adjusted by the educator after real-life testing. The student may be given the opportunity to show and hide the correct answers. For essay-type exercises a model answer or teacher's hint may be shown and hidden again in a similar fashion.
DaskaL's advantage over similar systems is that the educator may either specify correct answers on an ad hoc basis (by typing or pasting) or may specify that the correct answer(s) should be retrieved automatically from a database according to given criteria. The database will include lexica of the above-mentioned languages (presently only Bulgarian and Czech are available) in which each word form is associated with its lemma and its grammatical features. Thus the educator may specify that a given slot in each question in an exercise should be filled with masculine singular animate noun, generated automatically from different lemmas. The results from these queries may be used for populating a drop-down menu, for providing the correction to a typed-in answer from the student, or for providing part of the stimulus.

2 Application functionalities

DaskaL is a web-based application. It distinguishes three classes of users:

- Educator – with permission to create, edit and delete exercises and tests and specify access levels for them.
- Student – with access to a number of exercises and tests specified by his or her educator.
- Guest – with access to a number of exercises that are made available to any user.
- Administrator – who manages the users.

2.1 Functions for Educator users

A user logged in as Educator will meet a window for selecting tests and a window for selecting exercises. Making a selection in either of these windows will start the Test Editor or the Exercise Editor respectively.

2.1.1 Developing and editing exercises

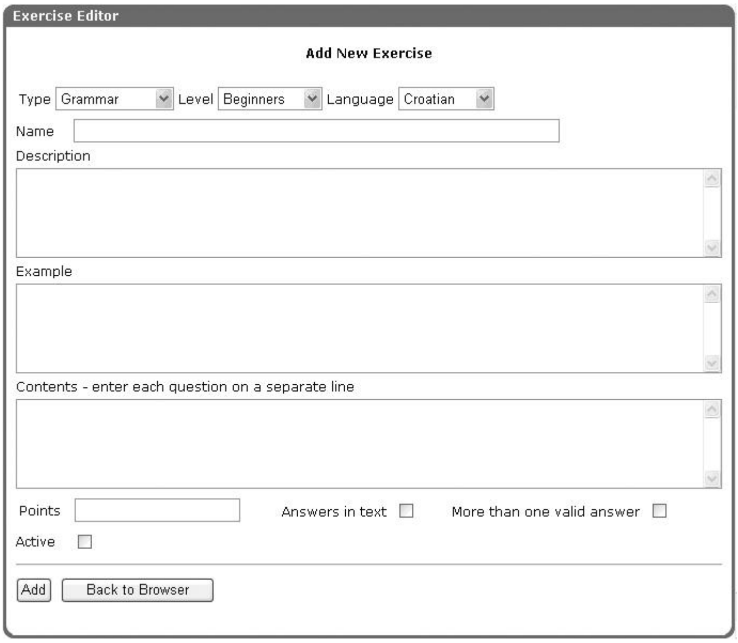
Selecting exercises
Checkboxes and menus in the Exercise Browser allow the Educator to select a list of exercises according to a number of parameters:

- Exercise Type – with the following alternatives: All Types, Grammar Exercises, Element Order Exercises, Semantics Exercises, and Mixed Exercises.
- Exercise Level – with the values: All levels, Beginners, Intermediate, and Advanced.
- Status – with the following alternatives: All, Active, and Inactive (active exercises are exercises that are accessible to Student and Guest users as part of a test, while inactive exercises are accessible only to the Educator (and Administrator.
- Language – the choice relates to the language that is taught (not the language of the menus) with the following alternatives: All Languages, plus an option for every installed language.
- Name - option for search by name (full or in part) of a given exercise.

Exercise list
The result from the exercise search appears in the Exercise browser as a list with three columns:

- The column "Exercise" contains a short description of the exercise. For inactive exercises, the text is greyed out.
- In the "Status" column active exercises are indicated by a green icon, while the column is empty for inactive exercises.
- The third column "Operations:" provides two options: Edit exercise and Delete Exercise.

At top and bottom of the list are links named "New Exercise", which lead to the form for creating exercises.
Creating an exercise
New exercises are created by filling in the fields of the form of the Exercise Editor (Figure 1). The filter fields reflect the basics of the chosen methodology. They are: **Type** (with the options Grammar, Element Order, Semantic, and Mixed), **Level** (with the options Beginners, Intermediate, and Advanced), **Language** (the language under study), **Name** (max. 255 characters), **Description** (a brief description of the task of the exercise), **Example** (an example of the type of questions the exercise contains), **Contents** (the content of the exercise itself), **Points** (the number of points awarded for a correct answer), **Answers in Text** (this option indicates that the exercise contains text with gaps to be filled in by the student and that the Contents field consequently should contains special codes for specifying the positions of the gaps), **More Than One Valid Answer** (this option allows the educator to specify several correct answers), **Active** (active exercises are those that are included in some test, while inactive exercises are not assigned to any test). When the fields are filled in, clicking the button Add will enter the exercise into the database.



Exercise Editor

Add New Exercise

Type Grammar Level Beginners Language Croatian

Name

Description

Example

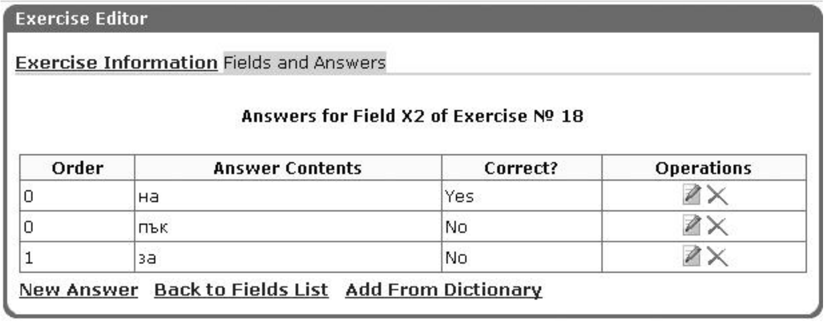
Contents - enter each question on a separate line

Points   Answers in text □   More than one valid answer □

Active □

Figure 1 The Exercise Editor

From the same window the link **Fields and Answers** opens a tab page with options for further specification of answer fields. The answer fields are shown in a table with the following columns providing different options for specification: **Field** an explicit link to the gap position code already assigned in the Contents field on the general descriptive page; **Type** the answer field can be specified either as a Type-in field (keyboard entry) or Choose from List; **Operations** – edit and delete operations for the given field.
If the field is of the type Choose from List, the full list of alternative answers from which the student should select the correct one should be entered (Figure 2).



Exercise Editor

Answers for Field X2 of Exercise № 18

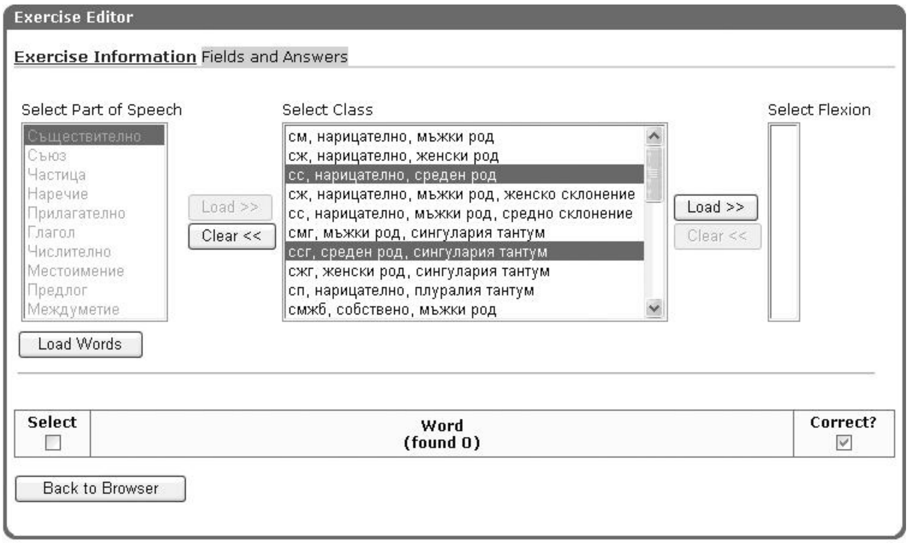
Order	Answer Contents	Correct?	Operations
0	на	Yes	✎ ✕
0	пък	No	✎ ✕
1	за	No	✎ ✕

Figure 2 Fields and Answers

The Educator may specify the order of the possible answers (applicable only for the type Choose from List) together with indication of whether they are correct or not. Lists of answer options may be made either manually or by database search.

Answer options through database search

If an inflectional dictionary for the given language is present in the database, answer options can be retrieved from it. The system presently contains Bulgarian [3] and Czech dictionaries. The form for dictionary access is shown in Figure 3.



Exercise Editor

Exercise Information Fields and Answers

Select Part of Speech

Съществително
Съюз
Частица
Наречие
Прилагателно
Глагол
Числително
Местоимение
Предлог
Междуметие

Select Class

см, нарицателно, мъжки род
сж, нарицателно, женски род
сс, нарицателно, среден род
сж, нарицателно, мъжки род, женско склонение
сс, нарицателно, мъжки род, средно склонение
смг, мъжки род, сингулария тантум
ссг, среден род, сингулария тантум
сжг, женски род, сингулария тантум
сп, нарицателно, плуралия тантум
смжб, собствено, мъжки род

Select Flexion

Select ☐	Word (found 0)	Correct? ☑

Figure 3 Answer options added from the Bulgarian morphological dictionary

In this form selection parameters for answer option retrieval are specified. The three tables: Part of Speech, Class, and Flexion, are hierarchically linked, so that a change of selection in a table on the left will cause the table on its right to be populated with new content. For instance, if a noun is selected from the Part of Speech list, the database relations will allow selection among different classes of nouns: common, proper, masculine, feminine, neuter, singularia tantum and pluralia tantum. The class specification in its turn interrelates with the grammatical features linked with the chosen class, so that for instance Bulgarian common feminine nouns will be specified as either singular or plural and either definite or indefinite. Selection of at least one option in the leftmost table is compulsory. Selecting options from the two other tables is optional and will narrow down the number of word forms returned. Multiple selections are allowed in the two leftmost tables.

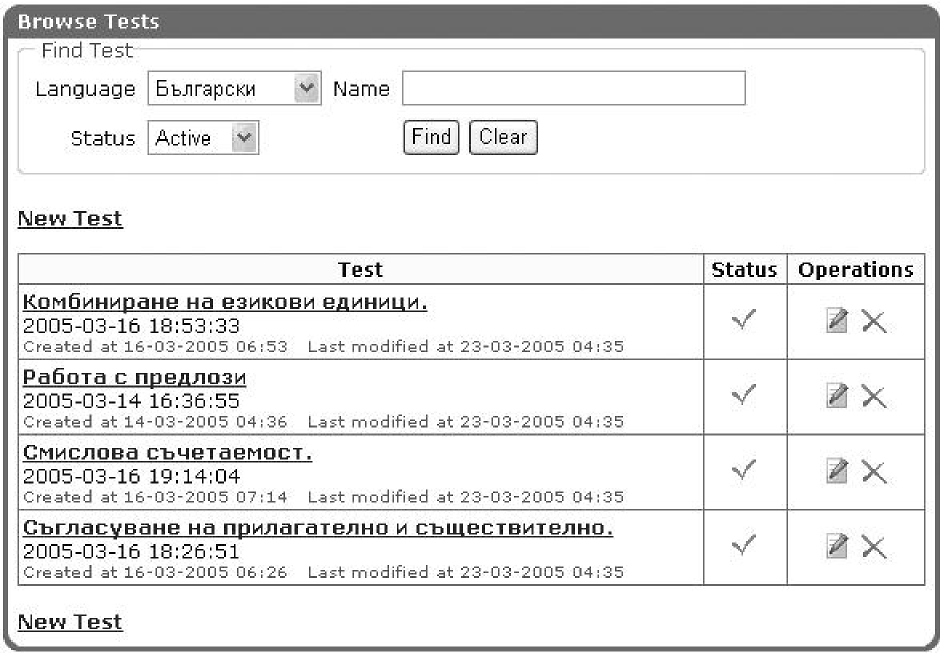
Clicking the button Load Words will load the word forms corresponding to the selected criteria and will show them in the table at the bottom. If, for example, Bulgarian common feminine noun, singular, and definite are the selected features, word forms like *rozata (the rose); zhenata (the woman)*, etc. will appear. The Educator can select one or several word forms from the list and mark them as correct or incorrect candidates for the answers.

For any given field in the exercise, answer options may be added by retrieval

from the database any number of times, with different criteria each time. The method may also be combined with manual editing of answer options. **Editing an exercise**
Editing an exercise works in the same way as creating an exercise, with the exceptions that fields are already filled in and the link Fields and Answers is active.

2.1.2 Developing and editing tests

Selecting tests
Possible search criteria for extracting tests from the database are Language, Status, and Name, corresponding to the same criteria for exercises.
Test list
The list of tests is a table similar to that used for exercises, with search criteria options and a list of test names (Figure 4). Links for "New Test" are at the top and bottom.



Browse Tests

Find Test

Language: Български
Name:
Status: Active

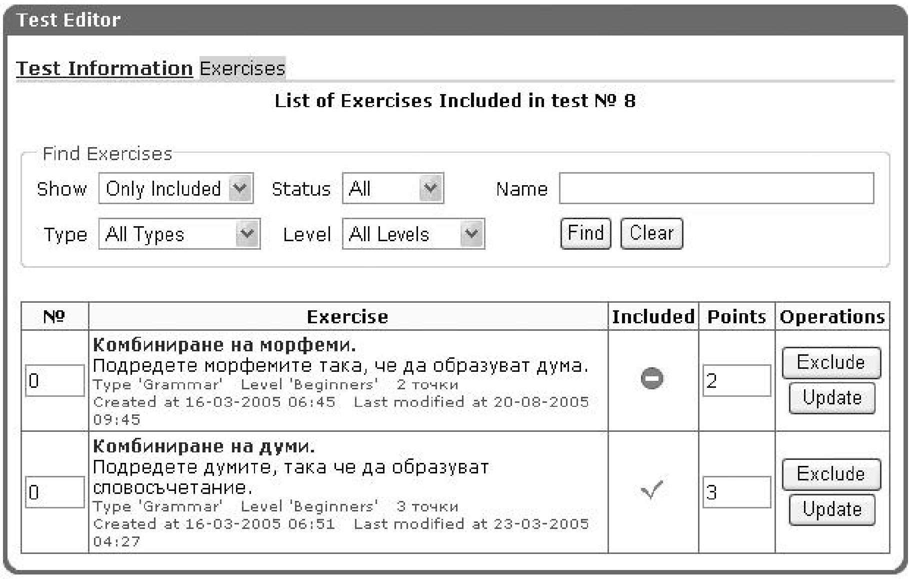
New Test

Test	Status	Operations
Комбиниране на езикови единици. 2005-03-16 18:53:33 Created at 16-03-2005 06:53 Last modified at 23-03-2005 04:35	✓	✎ ✕
Работа с предлози 2005-03-14 16:36:55 Created at 14-03-2005 04:36 Last modified at 23-03-2005 04:35	✓	✎ ✕
Смислова съчетаемост. 2005-03-16 19:14:04 Created at 16-03-2005 07:14 Last modified at 23-03-2005 04:35	✓	✎ ✕
Съгласуване на прилагателно и съществително. 2005-03-16 18:26:51 Created at 16-03-2005 06:26 Last modified at 23-03-2005 04:35	✓	✎ ✕

New Test

Figure 4 The test selection window

Creating a test
Tests are created with the Test Editor. It contains fields for Language, Name, Description, and Active. The Educator fills in these fields and enter the test into the database. He can then continue editing it in the Test Editor, and the link Exercises will be active. It opens a page from which exercises are added to the test. This page shows the exercises in a view similar to that of the Exercise Editor, but with some added features (Figure 5).



Test Editor

Test Information Exercises

List of Exercises Included in test № 8

Find Exercises

Show: Only Included

Status: All

Name:

Type: All Types

Level: All Levels

№	Exercise	Included	Points	Operations
0	**Комбиниране на морфеми.** Подредете морфемите така, че да образуват дума. Type 'Grammar' Level 'Beginners' 2 точки Created at 16-03-2005 06:45 Last modified at 20-08-2005 09:45	⊖	2	Exclude Update
0	**Комбиниране на думи.** Подредете думите, така че да образуват словосъчетание. Type 'Grammar' Level 'Beginners' 3 точки Created at 16-03-2005 06:51 Last modified at 23-03-2005 04:27	✓	3	Exclude Update

Figure 5 The Test Editor

This page allows viewing all exercises (which will be most convenient when creating a new test) or only those that are included (for reviewing the contents of an already existing test) and provides buttons for excluding, including or updating an exercise. As it is assumed that educators would not wish to make multilingual exercises, there is no option for Language here, that choice having already been made. The List of exercises has the following columns:

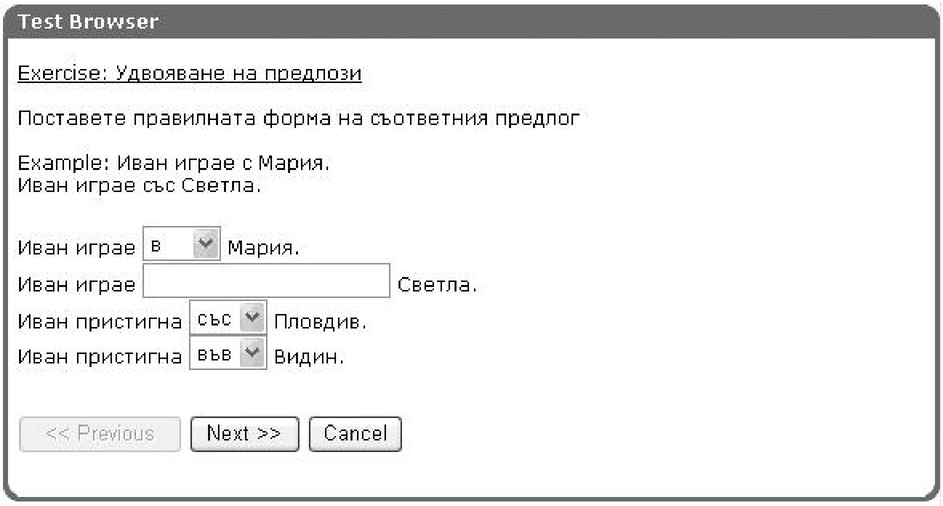
- Number – allowing the Educator to change the ordering of exercises within the test:
- Included – a green icon indicates an exercise that is active and included in the test. Exercises that are not included are greyed out. Exercises can be marked as inactive and will not be shown to student users even if included in the test.
- Points – the number of points awarded for a correct answer. If the exercise has been assigned a default value by the educator, this value will be shown here, but it may also be altered if the purpose of the test calls for it.
- Operations – with buttons for the functions Include (includes exercises in the test) and Exclude (removes the exercise from the test).
- Update – this button is used if the values for numbering and points have been changed.

Editing a test

Editing a test works in the same way as creating a test, except that fields are filled in and the link Exercises is enabled.

2.2 Functions for Student users

Users of the category Student may take any specified active test. Each exercise in the test will be shown on a separate page (Figure 6). These web pages are generated on the fly from the database according to the parameters set by the Educator. For instance, exercises may be generated with a selection of possible answers, or with a field for keyboard entry of the student's answer.



Exercise: Удвояване на предлози

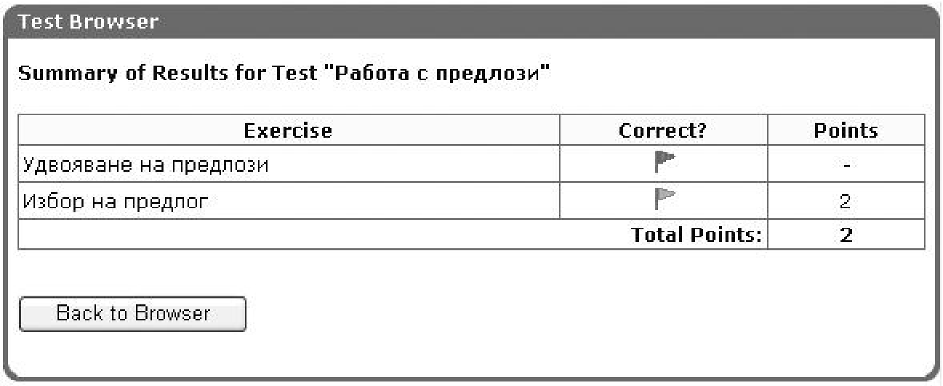
Поставете правилната форма на съответния предлог

Example: Иван играе с Мария.
Иван играе със Светла.

Иван играе [в] Мария.
Иван играе [ ] Светла.
Иван пристигна [със] Пловдив.
Иван пристигна [във] Видин.

Figure 6 Tests as seen by a Student user

Navigation buttons take the student through the exercises of the tests in the stipulated order. At the end of the test the student's score is calculated and shown in a detailed summary (Figure 7).



Summary of Results for Test "Работа с предлози"

Exercise	Correct?	Points
Удвояване на предлози	⚑	-
Избор на предлог	⚐	2
	Total Points:	**2**

Figure 7 Summary of the results

3 DaskaL application

There are two logical databases: The DaskaL application works entirely through a dynamic web interface, for students as well as for educators and administrators. From a technological point of view it is organized as a portal. A portal in this context is a web-based application that provides personalisation, single sign-on, and content aggregation from different sources and hosts the presentation layer of information systems [4]. It is organized through the portal framework Jetspeed [1]. The user interface consists of portlets giving various degrees of access according to the status of the user (guest; registered student; content creator: teacher, administrator). A portlet is defined as a Web component, usually managed by a container that processes requests and generates dynamic content. Portals use portlets as pluggable user interface components to provide a presentation layer to information systems [4]. The architecture is shown in Figure 8. Users may customize parts of the user interface. The application uses two separate databases, currently served by MySQL:

- for administrating the portal: users, status, permissions, customization, etc.
- for administrating the learning framework: exercises, dictionaries, tests, etc.

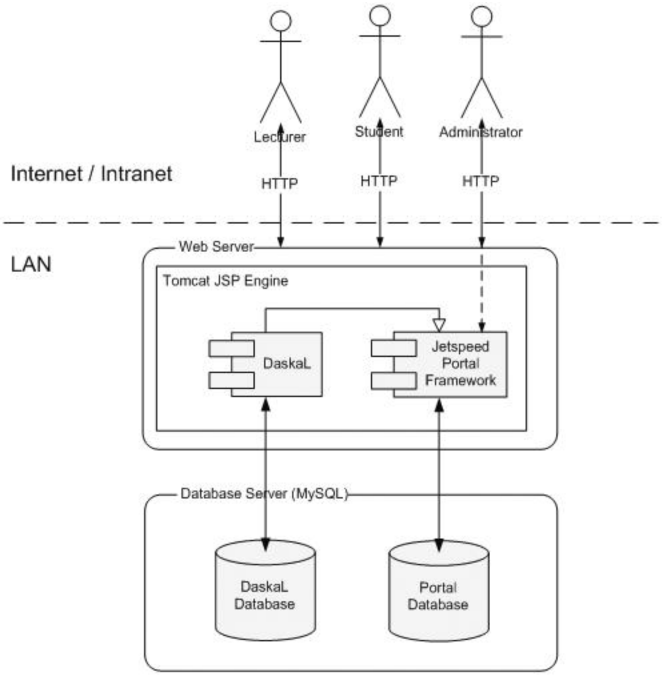


Figure 8 The DaskaL architecture

The architecture is multilayered, with the application layer divided into several sublayers implementing different functions. The system allows three kinds of users: – Educator, Student (which for this purpose includes Guest users), and Administrator. The administrator functions are implemented exclusively by Jetspeed, with new functions created within its framework for serving the students and lecturers. DaskaL is a pure Java application where the JSP technology is used for portlets implementation. The runtime environment is supplied by the JSP Engine – Apache Tomcat 5 [2]. For database server is used MySQL. There are two logical databases:

- the Portal Database – with the structures and data necessary for the operation of the portal – user accounts, ownership and permissions, user customizations, etc.

- the DaskaL Database – with the structures needed for DaskaL's functionality – exercises, tests, answers, etc.

Figure 9 shows the model of the DaskaL Database. The DaskaL Database is logically independent of the Portal database. Connection between the two is served by the table of user accounts. The DaskaL Database contains two groups of tables:

- For exercises and tests, with the description of the exercises, possible correct and false answers, description of the tests and students' test results.

- Dictionary tables supporting Unicode and thus allowing for dictionaries in almost any language.

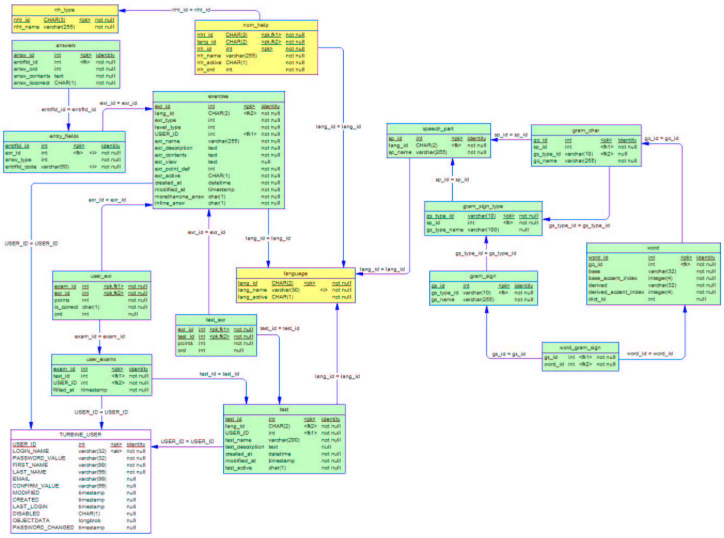

nh_type
nht_id CHAR(3) <pk> not null
nht_name varchar(255) not null
nht_id = nht_id
nom_help
nht_id CHAR(3) <pk,fk1> not null
lang_id CHAR(2) <pk,fk2> not null
nh_id int <pk> not null
nh_name varchar(255) not null
nh_active CHAR(1) not null
nh_ord int not null
answers
answ_id int <pk> identity
entrfld_id int <fk> not null
answ_ord int not null
answ_contents text not null
answ_iscorrect CHAR(1) not null
entrfld_id = entrfld_id
exr_id = exr_id
entry_fields
entrfld_id int <pk> identity
exr_id int <fk> <i> not null
answ_type int not null
entrfld_code varchar(50) <i> not null
exercise
exr_id int <pk> identity
lang_id CHAR(2) <fk2> not null
exr_type int not null
level_type int not null
USER_ID int <fk1> not null
exr_name varchar(255) not null
exr_description text not null
exr_contents text not null
exr_view text null
exr_point_def int not null
exr_active CHAR(1) not null
created_at datetime not null
modified_at timestamp not null
morethanone_answ char(1) not null
inline_answ char(1) not null
lang_id = lang_id
exr_id = exr_id
USER_ID = USER_ID
exr_id = exr_id
lang_id = lang_id
user_exr
exam_id int <pk,fk1> not null
exr_id int <pk,fk2> not null
points int not null
is_correct char(1) not null
ord int null
language
lang_id CHAR(2) <pk> not null
lang_name varchar(30) <i> not null
lang_active CHAR(1) not null
lang_id = lang_id
test_exr
exr_id int <pk,fk1> not null
test_id int <pk,fk2> not null
points int not null
ord int null
test_id = test_id
exam_id = exam_id
user_exams
exam_id int <pk> identity
test_id int <fk1> not null
USER_ID int <fk2> not null
filled_at timestamp not null
test_id = test_id
lang_id = lang_id
USER_ID = USER_ID
TURBINE_USER
USER_ID int <pk> identity
LOGIN_NAME varchar(32) <ak> not null
PASSWORD_VALUE varchar(32) not null
FIRST_NAME varchar(99) not null
LAST_NAME varchar(99) not null
EMAIL varchar(99) null
CONFIRM_VALUE varchar(99) null
MODIFIED timestamp null
CREATED timestamp null
LAST_LOGIN timestamp null
DISABLED CHAR(1) null
OBJECTDATA longblob null
PASSWORD_CHANGED timestamp null
USER_ID = USER_ID
test
test_id int <pk> identity
lang_id CHAR(2) <fk2> not null
USER_ID int <fk1> not null
test_name varchar(200) not null
test_description text null
created_at datetime not null
modified_at timestamp not null
test_active char(1) not null
speech_part
sp_id int <pk> identity
lang_id CHAR(2) <fk> not null
sp_name varchar(255) not null
sp_id = sp_id
gram_char
gc_id int <pk> identity
sp_id int <fk1> not null
gs_type_id varchar(10) <fk2> null
gc_name varchar(255) not null
gc_id = gc_id
sp_id = sp_id
gram_sign_type
gs_type_id varchar(10) <pk> not null
sp_id int <fk> not null
gs_type_name varchar(100) null
gs_type_id = gs_type_id
gs_type_id = gs_type_id
word
word_id int <pk> identity
gc_id int <fk> not null
base varchar(32) not null
base_accent_index integer(4) not null
derived varchar(32) not null
derived_accent_index integer(4) not null
dict_id int null
gram_sign
gs_id int <pk> identity
gs_type_id varchar(10) <fk> not null
gs_name varchar(255) not null
gs_id = gs_id
word_gram_sign
gs_id int <fk1> not null
word_id int <fk2> not null
word_id = word_id

Figure 9 The DaskaL Database

4 Conclusions and future directions

DaskaL provides a convenient tool for creating, editing, storing, retrieving and presenting foreign-language exercises on the World Wide Web, in almost any language supported by Unicode. The use of morphological dictionaries further speeds up development of exercises. Future development of the application includes XML import/export options for exercises, tests, and dictionaries. This will be important for exchange of exercises and tests between different organizations using DaskaL, for import into other computer-aided learning systems, for external (i.e. outside the DaskaL framework) archiving of exercises and tests in a device-independent format, and for hand-tooling exercises.

References

[1] Apache Portals – Jetspeed, http://portals.apache.org/jetspeed-1/

[2] Apache Tomcat, http://jakarta.apache.org/tomcat

[3] Koeva S. Bulgarian Grammatical dictionary. Organization of the language data, Bulgarski ezik, 1998, vol. 6: 49-58.

[4] Question mark Glossary, http://www.questionmark.com/uk/glossary.htm