Available online at www.sciencedirect.com

ScienceDirect

IFAC Papers Online CONFERENCE PAPER ARCHIVE

IFAC-PapersOnLine 48-29 (2015) 205–210

Virtual assistant for individualized practical training on controller design ⋆

Sanchez, Carlos * **Muñoz de la Peña, David** ** **Gomez-Estern, Fabio** ***

* *Wellness Telecom, Spain (e-mail: cscazorla@wtelecom.es).*
** *Departamento de Ingeniería de Sistemas y Automática, Universidad de Sevilla, Spain (e-mail: dmunoz@us.es)*
*** *Escuela Técnica Superior de Ingeniería, Universidad Loyola Andalucía, Spain (e-mail: fgestern@uloyola.es)*

Abstract: In this work we present a virtual assistant to help instructors offer individualized practical training on controller design for linear systems. The virtual assistant is designed to be used together with Goodle GMS, a web application which offers services for the collection, storage and automatic grading of students exercises. Using the proposed scheme, an instructor can generate in an easy way and without having knowledge of programming, individualized exercises that can be graded automatically, ensuring that all the exercises have a solution and the same degree of difficulty. The platform has been tested in an automatic control course at the University of Seville during the 2013-14 academic year.

Keywords: Internet-based Control Education Assessment, E-learning in Control Engineering, Problem-based Learning

1. INTRODUCTION

Goodle GMS is a project that began in 2007 in the Departamento de Ingeniería de Sistemas y Automática of the University of Seville (F. Gómez-Estern, 2010), (M. López-Martínez, 2010), (A. Muñoz de la Peña, 2012), (D. Muñoz de la Peña, 2012). The original objective of the platform was to execute in sequence a large number of Matlab programs developed by students in a programming course, and verify that the programs were correct using an input-output analysis approach without considering the structure of the code of the programs. The application developed offered services for the collection, storage and automatic grading of this class of exercises. However, it was soon clear that this application offered not only the possibility of grading Matlab programming exercises, but also general scientific or technical exercises.

This statement, which may look ambitious at first sight, is based on the idea that many scientific and technical exercises can be solved using a Matlab program. Consider a physic problem in which the solution is given by the inertia momentum and final speed of a given object. The web submission can be implemented with the following single text box:

```
J = 0; % Introduce the  inertia momentum Kg*m
V = 0; % Introduce the final speed m/s
```

The student replaces the zeros with the correct answer and submits the contents of the text box which is stored in the database. In this way, the single text box substitutes the standard sequence of text formularies that have to be defined for each exercise in most learning management systems (LMS) such as Moodle. Storing the solution of any exercise as a single character string provides flexibility at the cost of a more simple interface.

However, the main advantage of this approach is not avoiding to define the exercises formularies, but rather that the student solution, can be interpreted as Matlab code and hence it can be attached to a Matlab program designed by the instructor to analyze the values provided in the students answer. The next Matlab code verifies that the values provided by the students are equal to the correct answers, which in this case were 23.5 and 14.67:

```
grade=0;
Jc=23.5;
Vc=14.67;
if abs(J-Jc)<0.01
grade=grade+1;
end
if abs(Vinal-Vc)<0.01
grade=grade+1;
end
```

The execution of this code after the students code creates the variable `grade` in the workspace. The value of this variable depends on the number of correct answers of the students (0,1 or 2) and can be used to grade the student's work. This approach allows to grade automatically any exercise with a solution that can be codified numerically and that can be solved using Matlab.

This paradigm is of special interest in the context of automatic control courses because there are many design problems with numerical solutions such as the parameters of a proportional integral derivative (PID) controller or the

⋆ This work was supported by the MCYT-Spain under project DPI2013-48243-C2-2-R.

10.1016/j.ifacol.2015.11.238

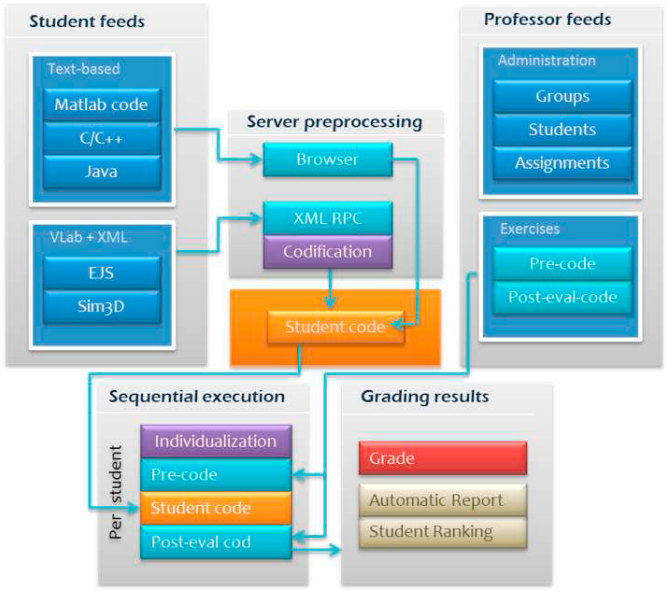

Fig. 1. Information flow in Goodle GMS

time domains specifications of the corresponding closed loop system (settling time, overshoot...). In this case, in order to determine if a solution is correct, a complex simulation is needed, which can be carried out in the instructor evaluation code using standard Matlab tools such as the Control Toolbox.

Finally, this scheme provides another important functionality from the education point of view, related to the possibility of providing individualized exercises to the students. The main idea is that the students of a given course have to solve exercises which are similar, but not the same. This is possible using the proposed paradigm in which the student's answer is interpreted as code if the exercises depend on a set of numerical input parameters, which is the case of scientific and technical exercises. For example, the final speed of an object after a known time with uniform acceleration depends on the initial speed and the acceleration. If these two parameters are different for each student, their solutions will also be different. To this end, an individual identification number such as the enrollment number can be used. Adding the following code to the grading code presented before, the correct answer is individualized:

```
V0 = 5 + ID(1);
a = 0.1*(1 + ID(2));
Vc = V0 + a*2;
```

where `ID(1)` and `ID(2)` are the first and second digits of the students ID which is retrieved from the database before executing the code. Providing the student with the same information in the exercise wording, individualized exercises can be implemented.

Goodle GMS implements this logic in a web application providing professional service to a large number of students and instructors. Goodle GMS has expanded over the last years, first because of the new universities that have collaborated in the project, and second because of the updates carried out on the web application that have provided new functionalities. This architecture has been applied in several different fields such as automatic control, optimal control, automation, mechanical engineering, chemical engineering and analytical chemistry. In addition, Goodle GMS has been extended to include not only Matlab based exercises, but also C, C++ and Java based exercises.

One of the most interesting functionalities included in the newest version of Goodle GMS is the possibility to connect directly with virtual and remote laboratories developed with Easy Java Simulations (EJS), which provides an alternative graphical interface for the students to submit their exercises and also provides the instructors access to this powerful modeling and authoring tool and the existing material (using the existing repositories). The connection is based on the Automatic Evaluation Element, described in (G. Farias, 2012) y (C. Sánchez, 2012).

The possibility of creating individualized exercises using simple explicit functions of the ID to define the exercise parameters has been of great use in different courses such as mechanical engineering, chemistry and physics because there are many different exercises in which, the solution is an explicit solution of the parameters and in general, modifying the value of the parameters does not affect the degree of difficulty of solving the exercise. Note that this does not implies that the exercises are simple, the calculations needed to obtain the solutions can be of any complexity.

However, this paradigm cannot be used in all classes of exercises. In particular, in control design problems, these conditions do not hold. The parameters that define the transfer function of a linear systems are not explicitly related to the values of the controller parameters that solves a given problem. In addition, controller design problems in general, do not have a single solution, but an infinite number of solutions because they are defined with a set of specifications that must be satisfied, for example, maximum rise time.

There is one additional problem when trying to use the aforementioned procedure to individualize controller design problems, which is the degree of difficulty of the different problems generated for different students. If the coefficients of the system transfer function depend on the different digits of the ID number for example, the dynamics may be different and the closed-loop specification that can be obtained can also be different, even leading to different design procedures. Parameterizing a control problem using an explicit formula of the ID number so that it can be explained in the student wording leads in general to different degrees of difficulty (the problems may not even have a feasible solution). This makes the individualization of control problems a very hard task, which in general can only be solved by trial and error in order to find a parametrization which provides a sufficiently individualized problems.

In this paper, this problem is tackled, providing a virtual assistant to help instructors offer individualized practical training on controller design for linear systems. Using the proposed scheme, an instructor can generate in an easy way and without having knowledge of programming, individualized exercises that can be graded automatically, ensuring that all the exercises have a solution and the same degree of difficulty. The virtual assistant is a web based application in which the instructor defines a control

problem, and it generates the evaluation code and the parameter generation code. In addition, it allows for a competitive evaluation paradigm in which the students compete with each other to design the best controller taking into account a particular performance index.

2. GOODLE GMS

The evaluation procedure described in the previous section is shown in Figure 1. Goodle GMS is a web application that runs on a dedicated server with a database and several applications such as Matlab and a C compiler. There are three different types of users, administrators, instructors and students. Administrators are in charge of configuration and of creating instructors profiles. Instructors on the other hand define all the necessary elements to carry out the automatic evaluation in a course. In particular they create students groups (with the corresponding profiles), exercises and assignments of exercises to student groups.

For each Matlab exercise, instructors have to provide the following two sets of Matlab code. First, a pre-evaluation code which is in charge of generating the parameters necessary to grade the exercise. In particular, they generate the individualized set of parameters of a problem depending on the ID of the student and the constants needed, for example the gravity value. Second, the evaluation code that is executed after the student solution which is in charge of providing a grade depending on the variables created by the students code. In addition to this code, there are other parameters that define some interface details of the web platform, such as whether the grade is public or not.

When a student logs in, all the exercises assigned to any of his groups are shown. If he selects one of them, the wording of the exercise, the text box with the answer template, and a link to a extended wording information (usually a PDF file) are shown. He submits the solution filling out the text template, and if the solution passes a set of Matlab syntax filters, the text is stored in the database to be graded later. The student answer is written in Matlab code (orange block in Figure 1) and is ready for execution. The evaluation consists in executing the pre-evaluation code, the student code and the evaluation code in order. Goodle GMS retrieves the value of a variable from the workspace and stores it in the database as the grade of the student submission.

3. LINEAR CONTROL DESIGN PROBLEMS

The contents of a basic control course are very large (Dorf and Bishop, 2005), (Guzmán, 2012), (Ogata, 2011), however one class of exercises which is of particular interest is the design of controllers for linear single-input single-output systems. The virtual assistant presented in this work focuses on this class of design problems, in particular in the design of a linear controller for the block diagram shown in Figure 2. The objective is to design a controller defined with transfer function C(s) in a way such that the closed-loop system satisfies a series of specifications, both in the time and the frequency domains.

In the time domain, we have considered overshoot, rising time and settling time of the step response. The steady state error for step, ramp and parabolic inputs are also

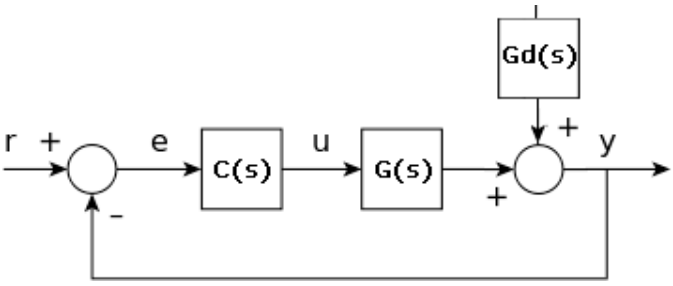

Fig. 2. Control exercise block diagram

considered. In the frequency domain we have considered the crossover frequency, the phase margin and the gain margin of the compensated system. All these specifications can be given with a maximum and/or a minimum value. This list is not closed, and it can be extended to include any other specification that can be measured from the block diagram and the corresponding transfer functions, both in an explicit way or by means of a simulation. With respect to the controller, any linear controller described by a transfer function can be evaluated, although we have consider standard PID and lead-lag controllers in standard form.

In addition to defining the specifications that the controller has to satisfy, in control is interesting the possibility of comparing different feasible solutions on behalf of a given performance index. The virtual assistant includes the possibility of implementing a competitive grading that depends on the following three indexes:

(1) Minimizing the integral square error (ISE). ISE integrates the square of the error over time. ISE will penalize large errors more than smaller ones (since the square of a large error will be much bigger). Control systems specified to minimize ISE will tend to eliminate large errors quickly, but will tolerate small errors persisting for a long period of time. Often this leads to fast responses, but with considerable, low amplitude, oscillation.

$$ISE = \int_0^{t_f} e^2(t)dt$$

(2) Minimizing the integral absolute error (IAE). IAE integrates the absolute error over time. It doesn't add weight to any of the errors in a systems response. It tends to produce slower response than ISE optimal systems, but usually with less sustained oscillation.

$$IAE = \int_0^{t_f} |e(t)|dt$$

(3) Minimizing the rising time.

4. INDIVIDUALIZED EXERCISE GENERATION

As it has been discussed in the previous section, we consider design problems defined by the block diagram of Figure 2. This class of exercise is defined by the transfer function of the system, the set of specifications, the class of controller considered and optionally, the performance index to be minimized. In this section, we present a

method to generate automatically individualized exercises ensuring that all the exercises have a solution and the same degree of difficulty.

In general, a random generation of the system parameters and the specifications cannot guarantee anything with respect to the feasibility, the difficulty or the procedures needed to solve the resulting problem. The proposed method consists on generating problems from a base problem provided by the instructor, in a way such that they are equivalent to the base one. This procedure allows a instructor to use exercises available in standard text books while guaranteeing that each student has to solve a different problem.

Small bounded variations of the exercise parameters can provide equivalent exercises, but there is no guarantee on the results obtained which implies that a trial and error procedure has to be implemented to test a posteriori whether the resulting problems are good enough. In addition, the problems may be too similar. In the virtual assistant we have followed a different path based on applying two transformations that do not change the exercise nature, but they change the parameters. In particular, we apply a time and gain scaling of the open-loop system. To generate an exercise, two scaling parameters are generated for each student, A and τ. The scaled system for the student is obtained as follows

$$G(s) = A \cdot Gb(\tau s)$$

where $G(s)$ is the transfer function of the student generated from the transfer function of the base problem $Gb(s)$ provided by the instructor to the virtual assistant. Taking into account the properties of the Laplace transform, this implies a time scale change, that is

$$t = tb/\tau$$

where t is the time of the generated exercise and tb the time of the base problem. A time scaling does not modifies the exercise nature, and the gain scale can be compensated by the controller. In particular, the control law

$$C(s) = Cb(\tau s)/A$$

where $Cb(s)$ is a controller that satisfies all the specification of the base problem, guarantees that the corresponding closed-loop system provides the same response as the base problem but with a time scaling. This implies that if the specifications are scaled appropriately, the controller $C(s)$ provides a feasible solution to the new exercise. Table 1 shows how to scale the different specifications considered.

One additional property of the proposed approach, is that it is still possible to compare the performance indexes of two different scaled systems, if the performance indexes are normalized with respect to the base time variable tb. This allows to grade in a competitive framework students which each one has different exercise parameters. Table 2 shows the normalized performance indexes.

Finally, in order to provide the students with their corresponding individualized parameters, a new procedure is proposed. In general, the individualized parameters were provided as explicit simple functions of the ID, so that each student would calculate their corresponding parameter. In this case, in order to hide from the students the transformation used and the base problem, a Matlab .p program is provided. This Matlab program asks for the ID of the student, and generates a text file with the corresponding individualized parameters and wording. The Matlab program is based on generating randomly but in a deterministic manner the two scaling parameters from the ID of the student. In particular, the parameters are generated using the function `rand` with the seed equal to the individual ID number, which generates a fixed random sequence of numbers that can be generated again in the pre-evaluation code during the evaluation procedure. The code that generates the scaling parameters between minimum and maximum values is the following:

Table 1. Scaled specifications

Specification	Base	Scaled
Overshoot	So	So
Rise time	Ts	Ts*tau
Settling time	Te	Te*tau
Crossover frequency	Wc	Wc/tau
Phase margin	Mf	Mf
Gain margin	Mg	Mg
Step steady state error	Erp	Erp
Ramp steady state error	Erv	Erv*tau
Parabolic steady state error	Era	Era*tau^2

Table 2. Normalized performance indexes

Rise time	Ts	Ts*tau
Integral square error	ISE	ISE/tau
Integral absolute error	IAE	IAE/tau
Integration time (ISE and IAE)	Tf	Tf*tau

```
rand('seed',ID);
A = Amin + (Amax-Amin)*rand(1);
Tau = Taumin + (Taumax-Taumin)*rand(1);
```

From these parameters, a code that generates and prints in a text file the corresponding transfer function and scaled specifications can be obtained. Figure 3 shows the modified work flow, in which it can be seen that in order to carry out the exercise, the student must execute in an external Matlab the .p Matlab program designed by the instructor (or in this case, by the virtual assistant). In future Goodle GMS versions, the generation of the parameters executing a Matlab code will be included in the functionalities.

5. VIRTUAL ASSISTANT

We present next the virtual assistant developed, which is accessible at `http://cscazorla.es/tesis/`. The virtual assistant is composed of two different elements, the graphical interface that the instructor uses to define the base problem and the underlaying application that recovers the information and generates automatically the submission template, the parameter generation (pre-evaluation) code and the evaluation code from this information.

The graphical interface is divided into two areas. The upper area consists of a tabbed document interface that allows the instructor introduce all the information needed to generate the exercises: the base transfer function, the specifications and their value on the grade, the controller

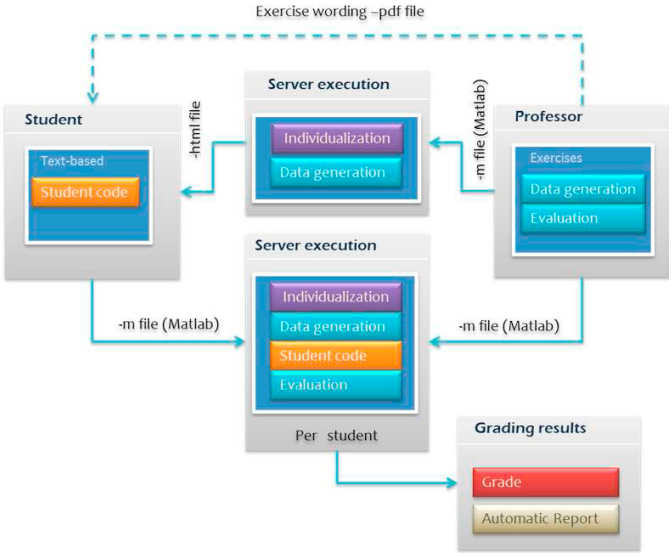

Fig. 3. Information flow in Goodle GMS with external Matlab execution

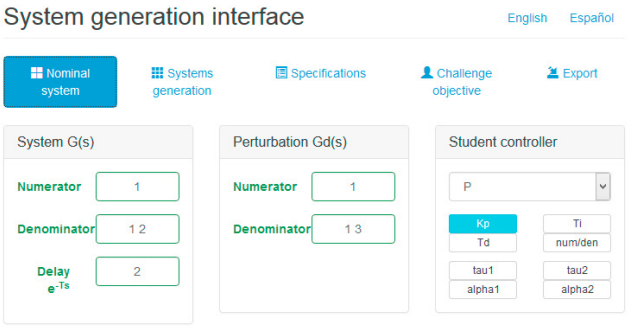

Fig. 4. Virtual assistant interface

structure and the performance index for competitive evaluation. Figure 4 shows this graphical interface.

In the lower area the results generated by the virtual assistant are shown: a graphic representation of the base transfer function and the controller, the submission template, the parameter generation code and the evaluation code from this information. This information is refreshed automatically when any input parameter is modified by the instructor. This is achieved using asynchronous JavaScript and XML petitions (AJAX) and a PHP script in charge of processing all the input information and generate the output. This implies that the instructor can modify the parameters without having to reload the web-page, providing a better user experience than standard web formularies.

The upper area has five menus: *Nominal system*, *System generation*, *Specifications*, *Challenge objective* and *Export*. In the menu *Nominal system* the instructor defines the transfer function of the system base problem providing the coefficients of the numerator and denominator polynomials using Matlab vector syntax, and chooses the controller that the student has to design among PID, lead-lag compensation networks and free-form. In the menu *System generation* the instructor provides the minimum and maximum values of the scaling factors A and τ.

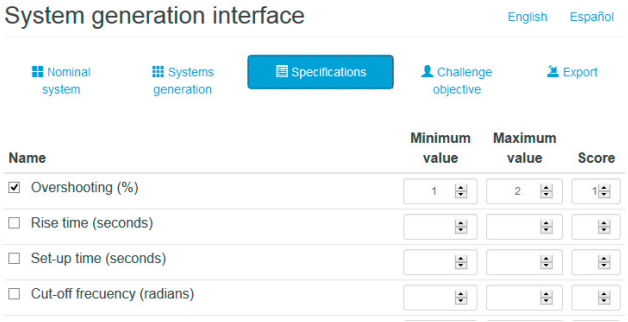

Fig. 5. Specifications menu

In the menu *Specifications* the instructor selects which specifications has to satisfy the controller defining a minimum/maximum value for each of them as well as the score given to this particular specification in the grade, which is computed as the sum of the score of all the specifications satisfied by the student controller.

In the menu *Challenge objective* the instructor can chose to do a competitive grading based on one of the three performance indexes considered, namely ISE, IAE and rising time. If the ISE or ASE are chosen, the instructor has to define the integral time. The grade of the competitive evaluation is obtained as follows. Each student that does not satisfies all the specifications is given a grade 0. The rest of the students are given a grade that depends linearly on their performance index, scaled in a way such that the worst student obtains a five, and the best student obtains a ten.

Finally, the menu *Export* allows the instructor to download a .prac file that can be used to generate an exercise in Goodle GMS directly with the resulting template and evaluation codes. Note that in any case, the instructor has to manually make the .p Matlab program with the generation code and provide it to the students using an external method (for example in the Moodle course page).

The virtual assistant currently supports English and Spanish, however it can be extended to new languages providing a single configuration text file with a set of text strings to translate the assistant. Each time the virtual assistant is opened it reads the language configuration files.

6. CASE STUDY

The virtual assistant developed has been used with success in the second year Control Theory course of degree “Grado en ingeniería de tecnologías industriales (industrial engineering) of the Universidad de Sevilla during the courses 2013-2014 to assign and grade five different PID control design exercises to the 453 students enrolled in the course. Goodle GMS was used to collect and grade the over 1500 submissions received.

The individualized exercise wording for each student defined the transfer functions polynomials using Matlab syntax, the controller structure and the specifications for each exercise. The students downloaded from the course webpage a Matlab program that given their ID number it generated a text file with this information. For the ID number 80071067 the following text was generated:

```
Exercise 1
For the system defined by the following
polynomials:
Numerator = [ 0 0 0 332.8 ];
Denominator = [ 0.1461 2.774 8.427 0 ];
design a PD controller that satisfies the
following specifications:
    SO < 20
    Ts (10-90%) < 0.1622

Exercise 2
For the system defined by the following
polynomials:
Numerator = [ 0 0 0 332.8 ];
Denominator = [ 0.1461 2.774 8.427 0 ];
design a PID controller that satisfies the
following specifications:
    SO < 20
    Ts (10-90%) < 0.1622

Exercise 3
For the system defined by the following
polynomials:
Numerator = [ 0 0 0 -3.328 ];
Denominator = [ 0.1461 3.051 13.69 16 ];
design a PD controller that satisfies the
following specifications:
    Mf > 30
    Wc > 11.3925

Exercise 4
For the system defined by the following
polynomials:
Numerator = [ 0 0 0 -3.328 ];
Denominator = [ 0.1461 3.051 13.69 16 ];
design a PID controller that satisfies the
following specifications:
    Mf > 40
    Wc > 9.4937

Exercise 5
For the system defined by the following
polynomials:
Numerator = [ 0 0 20.8 ];
Denominator = [ 0.2774 5.793 10 ];
design aPI controller that satisfies the
following specifications:
    Mf > 20
    Evrp < 0.0053
```

Each of these problems can be solved using a particular procedure, just as the base problems used to define them. The students submitted their answers without problems and the grading was carried out using Goodle GMS. For each student the evaluation code provided not only a grade, but also text comments indicating the results obtained in simulation of the students controller. For example, for one of the problems the evaluation code generated the following comments:

```
For the system defined by the following
polynomials:
Numerator = [ 0 0 0 462.3 ];
Denominator = [ 12.45 53.73 37.09 0 ];
the closed loop system had:
SO = 17.2497.
It satisfies the specification SO<21.00
Ts(10-90%) = 0.4355.
It satisfies the specification Ts(10-90%)<0.75
Grade: 2.0
```

In general, all the instructors involved in the course consider that the virtual assistant was of great help to implement the individualized exercises in the course. In particular, they valued very positively the possibility of using exercises designed in previous years as base problems, which allows to use all the material developed along the course history.

7. CONCLUSIONS

In this work we have studied the problem of generating individualized linear controller design exercises. We have presented a virtual assistant that given a base problem, provides the parameter generation code and evaluation code needed to use Goodle GMS in an individualized framework. The proposed approach guarantees that all the exercises have a solution and the same degree of difficulty. The virtual assistant allows instructors to implement individualized practical training techniques in control courses without any programming knowledge and using available exercises repositories.

REFERENCES

D. Muñoz de la Peña F. Gómez-Estern M. Sánchez A. Muñoz de la Peña, D. González-Gómez. Automatic web-based grading system: Application in an advanced instrumental analysis chemistry laboratory. *Journal of Chemical Education*, 90:308–314, 2012.

D. Muñoz de la Peña C. Sánchez, F. Gómez-Estern. A virtual lab with automatic assessment for nonlinear controller design exercises. *9th IFAC Symposium Advances in Control Education*, 2012.

S. Dormido D. Muñoz de la Peña, F. Gómez-Estern. A new internet tool for automatic evaluation in control, systems and programming. *IEEE Computers & Education*, 59:535–550, 2012.

R. C. Dorf and R. H. Bishop. *Modern Control Systems (12th Edition)*. Pearson, Inglaterra, 2005.

D. Muñoz de la Peña F. Gómez-Estern, M. López-Martínez. Sistemas de evaluación automática vía web en asignaturas prácticas de ingeniería. *Revista Iberoamericana de Automtica e Informtica Industrial*, 7(3):111–119, 2010.

L. de la Torre D. Muñoz de la Peña C. Sánchez S. Dormido G. Farias, F. Gómez-Estern. Enhancing virtual and remote labs to perform automatic evaluation. *9th IFAC Symposium Advances in Control Education*, 2012.

Costa-R. Berenguel M. Dormido S. Guzmán, J. L. *Control automático con herramientas interactivas*. Pearson, Inglaterra, 2012.

D. Muñoz de la Peña M. López-Martínez, F. Gómez-Estern. Automatic web-based evaluation of c-programming exercises in engineering education. *International Journal for Knowledge, Science and Technology*, 1(2):1–6, 2010.

K. Ogata. *Modern Control Engineering (5th Edition)*. Pearson, Inglaterra, 2011.