A Novel Approach Based Ontology for Assessing Learner's skills in the Programming language Practical Works Activities

Karima Boussaha[1], Farid Mokhati [2],Amira Haneche[3]

[1,2,3] Department of Computer Science, Larbi ben M'hidi University, Oum El Bouaghi, Algeria.

[1,2]Research Laboratory on Computer Science's Complex Systems (ReLa(CS)2)

Abstract.

The work presented in this research paper is an improved version and a solution to the limitations of our previously published work (Boussaha et al., 2015). In the previous work, we have used the graphs as a modeling tool, to introduce a new learner's self-assessment environment as PBL (problem-based learning) that allows comparison of learners' programs with those elaborated by the teacher. The subjacent idea is to indirectly compare programs through their graphical representations described by graphs. Therefore, we could only detect syntactic errors in learner's program. In the present work, we have managed to detect syntactic and semantic errors in a learner's program by using ontologies as modeling tools. The PBL environment developed so-called LearnAsPWAS(**L**earners' **As**sessment **P**ractical **W**orks **A**ctivities **S**ystem) allowing comparing learners' productions with those elaborated by the teacher. The tool allows essentially: (1) generating two ontologies from the learner's program and the teacher's one. Besides, (2) the tool applies some matching algorithms for measuring degrees of similarity and dissimilarity between learner's program and teacher's one. It offers an observation to the learner (3) assessing the learners by giving them a list of semantic and syntactic errors detected in their programs.

Keywords: Learners' Assessment, Didactic of the Programming, CEHL, Ontology Matching Algorithm, Practical Works, Ontologies.

1. Introduction

Our context of research is integrated into that of " CSCL"(Computer-Supported Collaborative Learning) within the community " CEHL (Computing Environment for Human Learning) ". The teaching of practical works is essential in the presential as in distant scientific and technical trainings, and answers a true need (Aiouni et al.,2018). Often, learners are deprived of this essential teaching and this is due to the problem of availability of the assistants, the problem of obstruction of learners, the material is expensive and cannot be duplicated. To minimize these problems thus teaching must answer to these needs.

The work presented in this paper is interested in the statement of problems which we judge important to take into account, during the development of a platform of training of practical works in programming it is the learners' self- assessment . So, Andrade and Du(2007) provide a helpful definition of self-assessment that focuses on the formative learning that can promote: Self-assessment is a process of formative assessment during which students reflect on and evaluate the quality of their work and their learning. They can also judge the degree to which they reflect on explicitly stated goals or criteria, identify strengths and weaknesses in their work, and revise accordingly (Sambell,2016).However in the practical work in general and especially the practical work of programming languages in computer science in the introductory courses in the first university cycle in the university years is usually accomplished by a group of learners as a result of the lack of adequate devices. And therefore when the assessment is given, it is one mark for all the group members and this makes the assessment subjective and does not reflect the true level of each learner belonging to the group, because there are elements of the group that do not work and rely on the active elements.

To overcome these problems, several problem-based learning systems are developed (Tadjer et al., 2018) these systems did solve the hardware problem, but the assessment problem was still not resolved.

We think that the self-assessment, in its formative function, is in the middle of the training considering its regulating function, which is paramount. The construct of self-assessment refers to the degree to which students can regulate aspects of their thinking, motivation, and behavior during learning (Tadjer et al.,2018)(Tadjer, et al.2020).
The present work concerns more particularly, the learners' self-assessment in the Problem-Based Learning environments of remote practical works in programming. Our goal is to suggest a self-assessment PBL environment for thinking about measures of cognitive knowledge, a self-assessment PBL that will help generate feedbacks, guide future research, and develop learners' efforts.
Currently, some researchers have examined issues related to the learner' assessment we can cite among them (Tadjer,2018,2020) (Hadadi & Bouaarab-dahmani,2019)),(Pang et al.,2019)(Indira et al.,2019)(Seman et al.,2018)and others. But they have

omitted a more precise learners' self-assessment in the practical works activities. Nevertheless, environments dedicated to learning practical works have been developed in the few last years.

To cope with this problem, in this paper, we aim to prototype a new learner's self-assessment environment as a PBL system. The self-assessment environment developed so-called LearnAsPWAS allowing an individual assessment for each learner by comparing learners' productions with those elaborated by the teacher. The developed system consider like a result of the investment in answering the research questions presented in the second paragraph (see research questions paragraph).

This paper is organized as follows. In the second section, we give some research questions. The architecture of the LearnAsPWAS system is presented in section three. In section four we present the results of the LearnAsPWAS system with a concrete study case. We conclude with a conclusion and future works.

2. Research questions

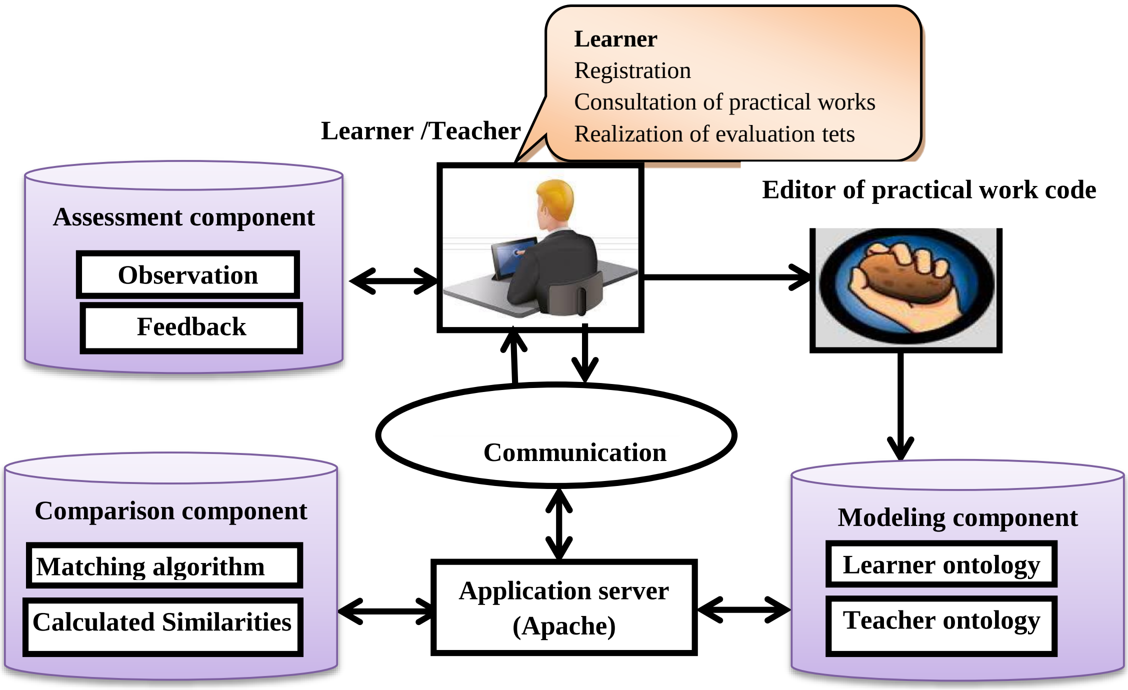
The problem of programming failures and the subjectivity of the assessment in practical works activities cited previously, let us reformulate our proposed contribution by investing in answering the following research question:

- Does the self-assessment in the PBL environment for learning practical works activities based on a matching algorithm and ontologies reduce the rate of failure or abandonment of the programming in the introductory courses in the first university cycle?

3. The methodology of the proposed system

The proposed system called LearnAsPWAS is organized in the form of three basic components: modelization component, comparison component, and assessment component. These three components interacted to adapt different aspects of the instructional process. Figure 1 illustrates the system architecture.

Figure 1: Architecture of the proposed system LearnAsPWA



3.1 Modeling component

This component is responsible for the modelization of the practical work code written either by the teacher or by the learner with two ontologies. In the flowing paragraph, we detail the modelization of the written code with the ontology.

3.1.1 The modeling of the practical work with the ontology

According to (Boaarab-Dahmani et al.,2017) an ontology has been defined as a formal representation of knowledge. Ontology is made up of four main elements: concept, instance, relation, and axiom.

3.1.1.1 The benefits of using ontologies PBL environments for learning practical work activities

- ✓ The formal representation of knowledge: ontology provides the basis for a formal encoding of entities, attributes, their relationships.
- ✓ Reuse and sharing of educational objects and this is relevant in the case of systems using educational resources that are already built because building them again can waste time. This through reusable ontology libraries.
- ✓ Identification of educational objects: an ontology can be used as a meta-descriptor to describe the semantic content of educational objects;
- ✓ Knowledge acquisition: The use of ontologies increases the speed and reliability of the knowledge acquisition process when building a practical work.

- ✓ it provides annotation markers that might facilitate the interoperability and exchange of learning resources,

3.1.1.2 The general structure of the practical work ontology

In this work, we need ontology to fully understand the structure of practical work, as well as to ensure the correct assessment of learners. For the construction of our ontology, we relied on the use of the Stanford methodology: it is suitable with the work on E-Learning. We used this method because Stanford University itself, which is developing the latter, is developing an editor called "protégé 2000" to properly show the practical side of ontology. This method goes through the following steps to build an ontology (Konys,2018).

- Step 1: Determine the domain and scope of the ontology

The field of use designed in our ontology is E-Learning. The purpose of using our ontology is to properly structure the practical work offered by the teacher, as well as implicitly ensuring an automated task which is the assessment task which is the key point in distance learning. Our ontology will be used by two actors: the learner and the teacher.

- Step 2: Reuse of existing ontologies

We don't need to reuse an existing ontology we have to build our ontology because the domain is restricted.

- Step 3: List the important terms of the ontology

The important terms in our work are: practical work, assessment, resource, editor, the standard answer, learner answer...

- Step 4, 5, and 6: Description of the ontology classes, their properties, and the class hierarchy

We have summarized these steps as follows: table 1 details each class with its attributes and its designation.

Table 1: The set of some classes and its attributes of the Practical work ontology

Class	Data Properties	Designation
PW(practical work)		This is the main class of our ontology.
date	Start-date End-date	Every PW has a start date and an end date.
Language	None-language	This class shows the language used by the learner to write his code.
Notion	Content-notion	Each PW has a notion that shows the main goal to be achieved and the work to be done in that practical work.
Example	Content –exple	Each PW presents examples to

		help learners to understand.
Observation	Observation	This class represents well the observation assigned to a learner after he has passed his assessment (the result of the test).
Question	Num-qst Ennonce-qst Type-qst Num-open-qst- Content-open -qst Num-closed-qst Content-closed-qst	As the precise class for assessing learners, it is presented as an open or closed question. Open question: represents questions for which the answer is open, a learner constructs his practical work freely. Closed question: represents questions whose answer is true or false or by a choice between propositions well defined in advance.
the standard answer	Ennonc-qst Content-Stdans	It represents the answer key to an open question, but it is not a final version as the learner will respond in their own style.
Answer	Content -Answ	It shows the answer proposed by the learner concerning the practical work.
Error	Content-err Num-er Type-er	This is the class that defines the set of errors that the learner makes when responding to an open-ended question. And it contains sub-classes: Syntax and lexicon errors. Semantic error which represents all the errors of meaning.
Title	Title	The title of each PW suggested by the teacher.

To create our ontology we used the «protégé 2000» editor. «Protégé 2000» is an authoring system for creating ontologies. It was created at Stanford University and very popular in the field of the Semantic Web and computer science research. «Protégé2000» is developed in Java. Protégé can read and save ontologies in most ontology formats: RDF, RDFS, OWL, …..etc.

In the following figure 2, we present an OntoGraf generated from Protégé editor to clearly understand the hierarchy of our practical work ontology. Consequently figure 3 shows the creation of the practical work ontology with protégé2000 editor.

Figure 2- the OntoGraf of the practical work ontology

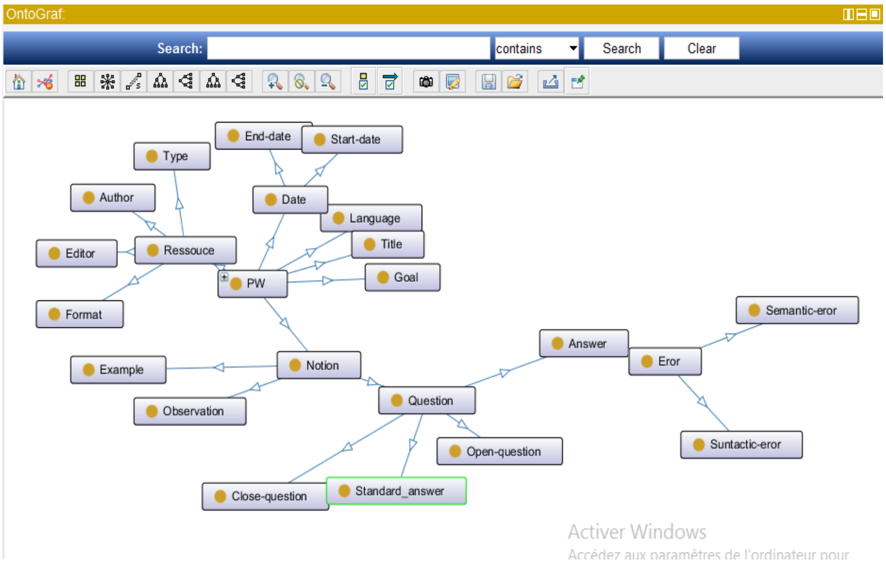
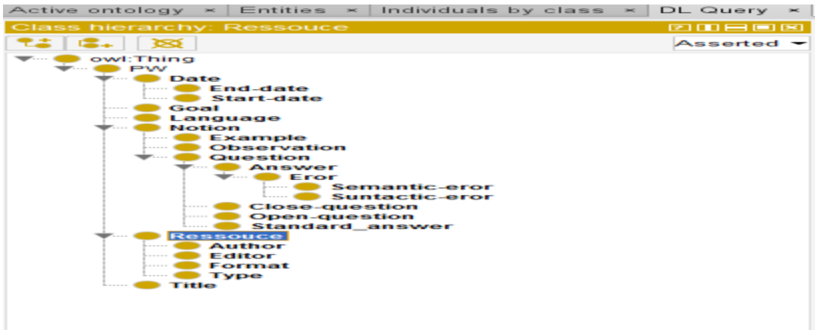


Figure 3: The hierarchy of concepts in the practical work ontology



.

In the following table 2, we represent the different links between some classes of practical work ontology in our ontology: there are different types of links which are:

- Generalization/specialization type links: this is «is a" type links: this is known in inheritance links, they are defined in the strict hierarchy of the model.

• The links "is connected of": this type of link defines the semantics between two classes. In our ontology, we used generalization/specification type links as well as a set of semantic links between classes.

Table 2: The description of some links between the classes of the practical work ontology.

Object Properties	Domains	Ranges	Inverse Of	Comment
attribute	Answer	Observation	is attributed to	Each completed PW has an observation.
Compare with	Answer	Standard _answer	Is compared to	The standard_answer is compared by the learner's answer to get an assessment.
Contain	Notion	Question	Is Expressed A	Each notion of PW contains questions.
Is a	Error	Syntactic_eror Semantique_eror	/	Each PW has errors and these can be syntactic or semantic.

3.2 Comparison component

This component allows comparison of learners' programs with those elaborated by the teacher. The subjacent idea is to indirectly compare program codes through their graphical representations described by ontologies. This component is divided into two modules:

3.2.1 Calculated similarities module:

This consists of recognizing the reference solution in the solution's base: to compare the proposed solution with the solutions' base, we must measure the degree of similarity and retain the solution closest to the solution proposed by the learner.

3.2.2. Algorithm matching module:

To calculate the similarity between the two ontologies we re-use the matching algorithm of Wu-Palmer and Dice(1994), This algorithm is based on ontologies and the similarity calculation between ontology concepts.

3.3 Assessment component

Is the most important component. It is implemented using the developed matching algorithm that evaluates the learner's solution in terms of similarity values to compare between learners' programs and the teacher's one. It divided into two modules:

3.3.1 Observation module

This module gives an observation to the learner according to the value of similarity calculated like flowing:

- [0.8; 1] ⇒ the observation will be ″Excellent″.
- [0.6; 0.8] ⇒the observation will be ″Very good″.
- [0.5; 0.6] ⇒the observation will be ″Average″.
- [0.3; 0.5] ⇒the observation will be ″Below Average ″.

- [0; 0.3] ⇒the observation will be ″Failing″.

3.3.2 Feedback module

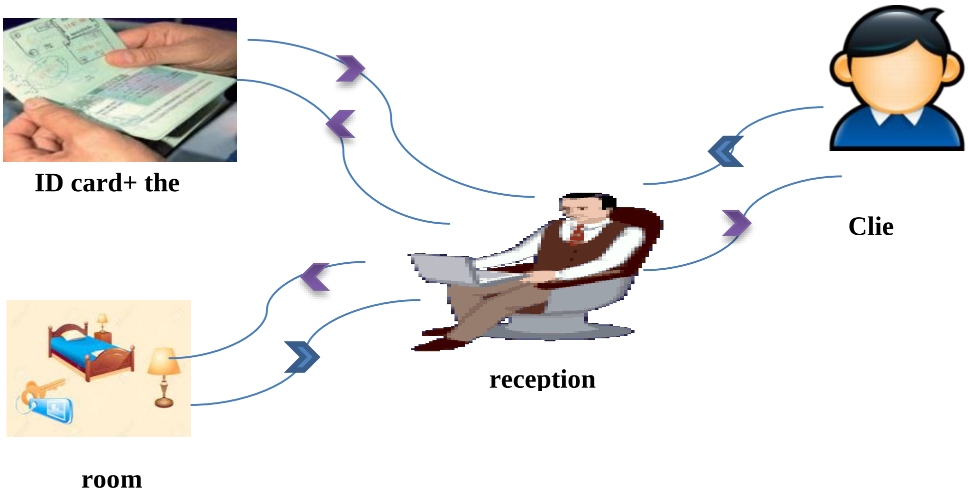
This module offers to the learners a set of syntactic and semantic errors detected in their programs.

4. Results

4.1 The Practical Work Statements

We present in what follows our environment using a concrete case study. It is about a simple hotel booking process. When a client arrives at the hotel, he presents his ID card. The hotel receptionist checks the availability of rooms. He proposes to the client the types and the price list of rooms. The client chooses a room and informs the date of departure to the hotel receptionist. The latter gives him the number of the reserved room.like it shown in figure 4.

Figure 4: The Practical Work Statements



4.1.1 Teacher's Java Object-oriented Program Practical Work (Reference Solution)

The teacher write his java object-oriented program code .It consider like a reference solution.

4.1.2 Program Fragment of Learner's Solution with Some Errors

The learner write his java object-oriented program code with some syntactic and semantic errors. In what follows the learner's fragment program with some errors .

- Define the same variables twice in different classes, for example:

```
public class client_a extends pers{
the_hotel_receptionist_a p;
int id_card;
String id_card;
void want_reserve(){
p.verif();
p.set();}}
```

- The combination of two modifiers in the practical work, for example:

```
public private class person_a{
final String first_name;
String last_name;
String adress; }
```

- named an attribute or a class with a name reserved for a keyword in the language used for programming, for example:

```
public class person_a{
}
public class{
        }
public class hotel_booking_a{
protected room_booking_a rb;
int nb_room;
int if;
boolean existe){}
int get_nbfreeroom(){}
void reserve(public int nb,int nb_ch){}
}
```

4.1.3 Generated ontology from Java Object-oriented Program Practical Work

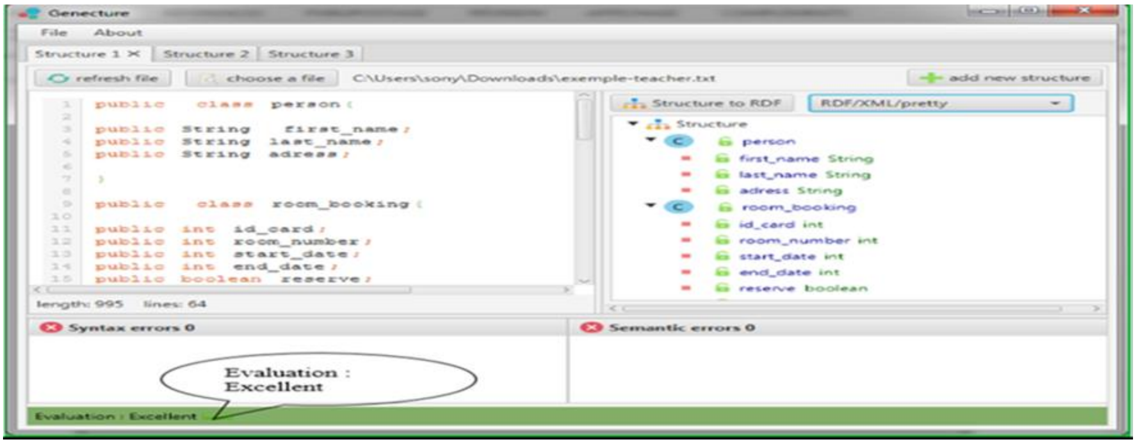
In our realized system we can generate from each practical work its ontologies in the form of an RDF / XML file. Figure 5 shows part of the RDF code generated from the teacher's Java object-oriented program practical work.

Figure 5: ontology generation in the form of an RDF code.

```
<rdf:Description rdf:about="C:\Users\Desktop\Mémoire\Amira\exemple 1.txt\class.0#">
  <sub_declaration.5:sub_declaration>
    <rdf:Description rdf:about="C:\Users\Desktop\Mémoire\Amira\exemple 1.txt\class.0\sub_declaration.5#">
      <sub_declaration.5:identifier>smoking</sub_declaration.5:identifier>
      <sub_declaration.5:type>boolean</sub_declaration.5:type>
      <sub_declaration.5:modifier>public</sub_declaration.5:modifier>
    </rdf:Description>
  </sub_declaration.5:sub_declaration>
  <sub_declaration.4:sub_declaration>
    <rdf:Description rdf:about="C:\Users\Desktop\Mémoire\Amira\exemple 1.txt\class.0\sub_declaration.4#">
      <sub_declaration.4:identifier>reserve</sub_declaration.4:identifier>
      <sub_declaration.4:type>boolean</sub_declaration.4:type>
      <sub_declaration.4:modifier>public</sub_declaration.4:modifier>
    </rdf:Description>
  </sub_declaration.4:sub_declaration>
  <sub_declaration.3:sub_declaration>
    <rdf:Description rdf:about="C:\Users\Desktop\Mémoire\Amira\exemple 1.txt\class.0\sub_declaration.3#">
      <sub_declaration.3:identifier>end_date</sub_declaration.3:identifier>
      <sub_declaration.3:type>int</sub_declaration.3:type>
      <sub_declaration.3:modifier>public</sub_declaration.3:modifier>
    </rdf:Description>
  </sub_declaration.3:sub_declaration>
  <sub_declaration.2:sub_declaration>
    <rdf:Description rdf:about="C:\Users\Desktop\Mémoire\Amira\exemple 1.txt\class.0\sub_declaration.2#">
      <sub_declaration.2:identifier>start_date</sub_declaration.2:identifier>
      <sub_declaration.2:type>int</sub_declaration.2:type>
      <sub_declaration.2:modifier>public</sub_declaration.2:modifier>
    </rdf:Description>
  </sub_declaration.2:sub_declaration>
  <sub_declaration.1:sub_declaration>
    <rdf:Description rdf:about="C:\Users\Desktop\Mémoire\Amira\exemple 1.txt\class.0\sub_declaration.1#">
      <sub_declaration.1:identifier>room_number</sub_declaration.1:identifier>
      <sub_declaration.1:type>int</sub_declaration.1:type>
```

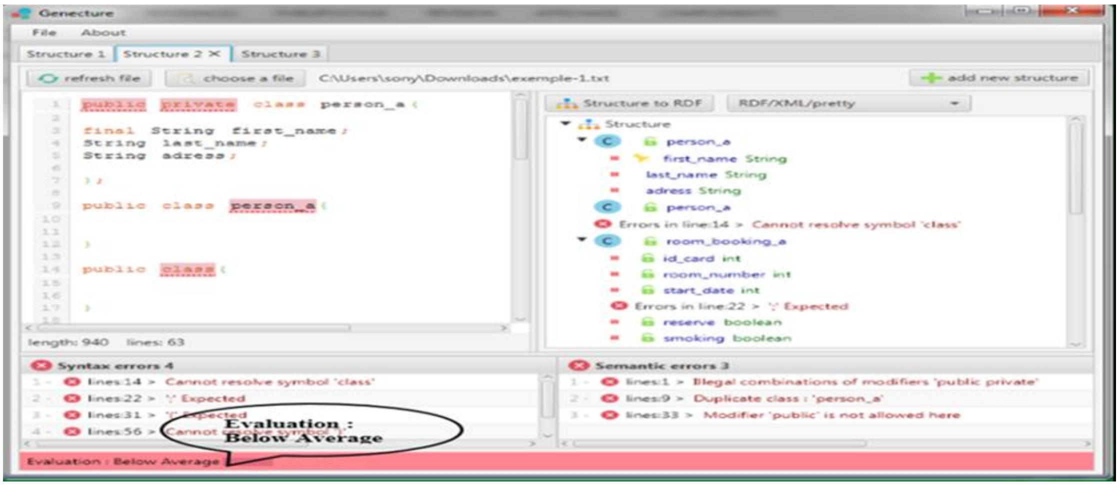
The main objective of the developed PBL environment is to allow learners to auto-evaluate their skills in Java object-oriented programming. The rest of the steps are only done by learners (i.e., the teacher is not involved in the rest of the self-assessment process).. During the comparison process, we proposed, we use the teacher's ontologies as references (i.e., they are supposed correct because they are generated from the teacher's correct programs). So, the list of detected errors is essentially different concepts that are not found in the learners' programs. Also, several kinds of errors may be detected in learners' programs (see Figures 5, 6).

Figure 5: list of detected errors in the case of learner's correct programming practical work



Our LearnAsPWAS displays to learners the list of the detected errors in their programs. Figure 6 illustrates the errors detected in learner's programs.

Figure 6: list of detected errors in the case of learner's incorrect programming practical work



5. Conclusion and future work

as it's known, programming cannot be learned solely from books as in other subjects, learning and teaching the basics of programming is a complicated task, and the best way for learning programing is practicing programming techniques and concepts by trying them out yourself. So, Computer-based assessment(CBA) is useful for handling very large numbers of students. Apart from giving a score, these tools may also offer an environment to practice programming by developing algorithms themselves to deepen their understanding.

In this research paper, we have proposed a new learner's self-assessment environment for allowing learners to evaluate and test their skills in Object-Oriented Programming using Java language. It implements our approach that considers both structural and dynamic aspects of object-oriented programs. The developed environment allows comparing indirectly the learners' programs with the teacher's one through the comparison (syntactic and semantic comparison) of their ontologies descriptions using two kinds of matching (static and dynamic matching).

As future work, we plan to extend the proposed approach and the developed environment to:

- Introducing conformity testing techniques for testing all the functionalities of the learners' programs and to ensure either they comply with the teacher's program or not.
- Assessing learners' practical works activities in different domains not just in computer science.
- Designing other ontologies and combine them with ours to enrich the semantics especially in the assessment task.
- we still need to assess how much this system can improve students' achievement in programming but in the Moocs context.

Reference

Aiouni, R., Bey, A. & Bensebaa, T. (2018) 'eALGO: an automated assessment tool of flowchart programs for novices', *Int. J. Innovation and Learning*, 23(2).

Andrade, H., & Du, Y. (2007).Student responses to criteria-referenced self-Assessment. Assessment and Evaluation. Int.J. of Higher Education, 32(2), 159-181.

Bouarab-Dahmani, F,., Comparot, C,. , Si-Mohammed, M,.&Charrel, P, J,. (2017). Ontology-Based Teaching Domain Knowledge Management for E-Learning by Doing Systems. *Electronic Journal of Knowledge Management*, 13 (2). pp. 156-171.

Boussaha, K., Mokhati, F.,& Chaoua, Z.(2015).Architecture of a specific platform for training practical works: integration of learners assessment component. , *Int.J. of Technology Enhanced Learning(IJTEL)*, 7(3).

Indira, B., Valarmathi, K., & Devaraj, D. (2019). An approach to enhance packet classification performance of software-defined networks using deep learning. *Soft Computing, 23(18)*, 8609–8619. https://doi.org/10.1007/s00500-019-03975-8.

Konys, A.(2018). An Ontology-Based Knowledge Modelling for a Sustainability Assessment Domain. *Sustainability*10(300)

Tadjer, H., Lafifi, Y., & Seridi-Bouchelaghem, H. (2018). A new approach for assessing Learners in an Online problem-based learning environment. *International Journal of Information and Communication Technology Education (IJICTE)*, 14(4), 18–33.

Tadjer, H., Lafifi, Y., Seridi-Bouchelaghem, H., & Gülseçen, S.,(2020) Improving soft skills based on students' traces in problem-based learning environments *Interactive Learning Environments*, p. 1-18, https://doi.org/10.1080/10494820.2020.1753215.

Palmer,M,S., & Wu,Z.(1994). Verb Semantics And Lexical Selection. In the 32nd. Annual Meeting of the Association for Computational Linguistics, pp. 133--138.

Pang, E., Wong, M., Leung, C. H., & Coombes, J. (2019). Competencies for fresh graduates' success at work: Perspectives of employers. *Industry and Higher Education*, 33(1), 55–65. https://doi.org/10.1177/0950422218792333.

Sambell, K.(2016).Assessment and feedback in higher education: considerable room for improvement?. , in Student Engagement Int.J.of Higher Education, 1(1),2016.

Seman, L. O., Hausmann, R., & Bezerra, E. A. (2018). On the students' perceptions of the knowledge formation when submitted to a project-based learning environment using web applications. *Computers & Education*, 117, 16–30. HTTP:// doi.org/10.1016/j.compedu.2017.10.001.