Another Approach for the Teaching of the Foundations of Programming using UML and Java

RAFAEL RIVERA-LOPEZ, ESTELA RIVERA-LOPEZ and ABELARDO RODRIGUEZ-LEON
Departamento de Sistemas y Computación
Instituto Tecnológico de Veracruz
Calzada M. A. de Quevedro 2779, Col. Formando Hogar, Veracruz, Ver.
MÉXICO
{rrivera, erivera, arleon}@itver.edu.mx

Abstract: - This paper presents one proposal for teaching the foundations of object oriented programming using the Unified Modeling Language (UML) as one modeling language and Java as one object oriented programming language. Unlike other approaches where only two types of UML diagrams are used with the intention of transferring them to a codification (class diagram and sequence diagram, or class diagram and statechart diagram), this proposal uses three types of UML diagrams: class diagram, sequence diagram and activity diagram, for modeling basic programming problems. This is with the intention of facilitating to the student the transition between the diagrams used to model one solution and its final codification in one programming language.

Key-Words: - UML, object oriented teaching

1 Introduction
Teaching object oriented programming has become a common approach in the worldwide universities. In 2001 and 2005, the Joint Task on Computing Curricula, formed with IEEE and ACM, describes the characteristics for implementing programming courses in universities [9]. As a result of this effort, most universities have adopted an object oriented language for their first programming course. In 2004, the National System of Technological Superior Education of Mexico defines a new Educative Model named “for the XXI century” [18]; this model includes the object oriented paradigm from the beginning of the university studies in the computation area. In the first course, named Programming Foundations, students know the object oriented model, doing emphasis in UML as a tool of modeled; later they know elements of the logic of programming as the implementation of expressions and sentences. In a second course, students know the fundamental characteristics of the objects oriented programming: encapsulation, inheritance and polymorphism and, in a last course of programming, they handle the concepts of design of graphical interfaces, event based programming, multi-threads programming and components based programming.

This paper focuses in developing a proposal of application of three diagrams UML for the starting phase of the teaching of the Foundations of Programming, as well as its transferring to source code, using Java as one object oriented programming language.

2 UML and object oriented teaching
Teaching of object oriented paradigm has received especial attention by the educative community [8]. In the present day, several studies exist where the effect of to use the object oriented paradigm in the first computation course are both analyzed and evaluated ([1], [4], [17]). UML is accepted as a tool for modeling systems and it has been used for the teaching the object oriented paradigm ([5], [16]).

2.1 UML
UML is an OMG standard language to specify, visualize and document software models using the object oriented paradigm [2]. UML provides several diagram types that can be used to view and model the software system from different perspectives and different levels of abstraction [14]. By example, a diagram class shows a static view of the system, and the interactions diagrams show the dynamic view of the system.

Class Diagram: Class diagram shows a static view of the system. Classes are abstractions that especific the common structure and behavior of a set of objects [3]. Class diagrams do not describe behaviors or how instances of the classes interact. To describe behaviors and interactions between objects in a system, we can turn to interaction diagrams. Interactions diagrams are sequence diagram, state diagram and activity diagram.

Sequence diagram: Sequence diagrams show the classes along the top and messages sent between those classes, modeling a single flow through the objects in
the system. A sequence diagram implies a time ordering by following the sequence of messages from top left to bottom right [10].

**State diagram:** Whereas interaction diagrams show objects and the messages passed between them, a state diagram shows the changing state of a single object as that object passes through a system [10].

**Activity diagram:** An activity diagram is the UML version of a flowchart. There are used to analyze processes and allowing you to visualize the flow of events from one activity to another. They are most often used to model: Workflows of business processes, use cases and the actions of an operation (a computational process) [12].

### 2.2 Related Work

There are two approaches for teaching the object oriented paradigm. In both cases, the class diagram is the principal diagram used for modeling the static view a system. Several courses use state diagrams for modeling the dynamic behavior of a system, and another courses use sequence diagrams as base for modeling this behavior. In these courses appropriate UML modeling tools have to be used in conjunction with the theory developed.

Many object oriented modeling tools ([7], [19]) generate limited skeleton code from the static view of a system such models. In [6] and [14] automatically generate Java code from the class and state diagrams. In [11], [15] and [16] two types of diagrams are used (class and sequence diagram) for the conversion to code. In [13] is presented one approach where activity diagrams are used for modeling behavior and complete code generation.

### 3 An approach for teaching the Foundations of Programming using UML and Java

In this proposal, the teaching of the foundations of programming is developed in six stages. The first three conform the phase of modeling using UML and the others conform the phase of codification in Java. Fig. 1 shows the six stages in this proposal:

1.- To design the class diagram.
2.- To design the sequence diagram.
3.- To design the activity diagrams.
4.- To implement the Java class.
5.- To implement the Java Main class.
6.- To implement the methods of the Java class.

The approach presented in this article proposes to avoid the inherent complexity of a object oriented design using all the potentiality of UML to the inexperienced students in the computation area. In order to reach the objectives of an initial course of computation, this approach proposes six stages described in the following sections, so that the students adopt a basic methodology to construct solutions using this paradigm.

### 3.1 To design the class diagram

From the initial formulation of the problem, the nouns and the verbs are identified. Later, using abstraction, the main object of the problem is identified, as well as its characteristics (using some nouns) and its possible behavior (using the verbs). In any computer system, the values of the attributes are stored in memory spaces. These spaces must have a name and a data type (integers, with decimals, logical, characters, etc.). Then, class characteristics are represented as variables with names and types. In other hands, the behavior of an object is represented by functions or methods. In general terms, the concept of function is the same that in mathematics. A function has arguments and always returns a value. The class diagram describes static aspects of the system, still in its simpler phase, due to in a basic programming course. In this type of courses, students start solving basic problems where many interactions between several objects within the system are not presented.

Like in another courses of physics, algebra or chemistry, both the procedures of resolution and the fundamental principles of the matter are acquired gradually using basic problems. In the case of programming course, before working with problems that involve inheritance and polymorphism, the
student works with basic problems where the concepts of class, attribute, method and the basic logic of each method are used.

3.2 To design the sequence diagram
In the static view cannot be represented a logical flow of operations on objects. A sequence diagram is used to indicate the order in which the messages are sent to the objects that exist in the system. Sequence diagram are used for modeling the main sequence of operations within a system. The messages can be both asynchronous and synchronous.

When the sequence diagram is translated into one Java program, this program represents one class identified as Main Class. This class is not used as base for creating objects, and it only serves to implement the main() method of a program codified in Java. This has similarity with the resolution procedure of problems in physics or chemistry, where, from a set of formulas, one sequence of operations is applied in order to found the result.

3.3 To design the activity diagram
Using an approach similar to the proposed in [13], activity diagrams are used to design the logic of each method of a class. In this approach, a activity diagram for each method declared in the class diagram will be designed. As is indicated in [12], one activity diagram is used for modeling a computational process. In this case, an activity diagram represents the logic of each method. One activity diagram can be used for designing sequential algorithms, with both selection and iterative statements.

The idea to associate a activity diagram for each method that models the behavior of a class is for simplifying the development of algorithms to the students, avoiding the construction of complex solutions in a single activity diagram. Each activity diagram has a specific function that represents a type of behavior clearly defined and delimited for a class.

3.4 To design the Java class
Using the class diagram designed in section 3.1, one program in the Java language will be codified. This program will be used to create the objects that interact in the system (fig. 2). Students must identify the relation that exists between the class diagram and the Java code. Students must understand that in both cases a language is used: for the class diagram a graphical language is used and for the program in Java a written language is used. Students must also understand that this step is simply a translation between languages.

3.5 To design the Java Main class
Using the sequence diagram designed in section 3.2, students will translate this diagram in one Java class named the Main Class, where the main() method will be located and the messages included in the sequence diagram will be implemented (fig. 3).

The synchronous messages are associated to methods
that return void and the asynchronous messages are those that return a value. In this stage is desirable not to include loops or nested messages, so that the student identifies initially the relation between a sequence of messages and the sequence of operations within the main() method.

3.6 To construct the logic of each method
In this last stage, each one of the activity diagrams designed in section 3.3 of this proposal are transformed. Previously to the translation in one Java program, students must learn the rules for the writing of mathematical and logics expressions. Also, students must learn the rules of syntax of the main types of sentences: selections and iterations (fig 4).

![Fig. 4. The translation from activity diagrams to methods in one Java class.](image)

4 Remarks
It is important to indicate that the objective of the course of Foundations of Programming proposed by the Educativo System in Mexico is that the student learns to analyze problems and to design solutions to these problems using the object oriented approach. The introduction of the concepts of encapsulation, inheritance and polymorphism in the second course of Programming allows the student to reinforce the concepts learned in the fundamental course.

The approach in this work focuses in only using a subgroup of UML diagrams and to apply a basic procedure to design solutions using an object oriented approach. This procedure will help the student to transform, of practical way, the graphical schemes of design to a set of concrete programs in a object oriented programming language.

5 Conclusion
This article presents an object oriented approach using UML and Java for the teaching of the foundations of programming. This approach proposes the application of a procedure of six steps to analyze, to design and to implement a solution to a basic problem of computation with a program codified in a object oriented programming language. This is an initial proposal, that will be refined in the future for including of incremental way the schemes of exceptions handling, arrays, overloading, inheritance and polymorphism.

References:
[1] C. Alphonce, and P. Ventura, Object orientation in CS1-CS2 by design, in Proc. of the 7th Annual Conf. on Innovation and Technology in Computer Science Education (ITiCSE '02), ACM, 2002, pp. 70-74.


