ANNALS OF FACULTY ENGINEERING HUNEDOARA
– International Journal of Engineering
Tome XI (Year 2013) – FASCICULE 4 (ISSN 1584 – 2673)

[1.] Anca-Elena IORDAN

DEVELOPMENT OF INTERACTIVE DIDACTIC SOFTWARE USED IN THE STUDY OF TERNARY TREES

[1.] UNIVERSITY POLITECHNICA OF TIMISOARA, FACULTY OF ENGINEERING HUNEDOARA, ROMANIA

ABSTRACT: This work presents the needful phases required for object oriented development of interactive didactic software used in the study of ternary trees. The development of the software is achieved about specifically UML diagrams representing the phases of analysis, design and implementation, hereby the software being described in a terse and outright layout. The software is much helpful to both students and teacher's forasmuch computer programming area, for the most part trees theory area, is difficult to understand for most students.
KEYWORDS: Ternary Trees, Complete Ternary Trees, Ternary Search Trees, UML, Interactive Didactic Software, Java

INTRODUCTION IN TERNARY TREES THEORY

In computer science, a ternary tree [1,2] is a tree data structure in which each node has at the very outside three child nodes, accustomed distinguished as "left", "mid" and "right". Nodes with children are parent nodes, and child nodes may contain references to their parents. With the exception of the tree, there is frequently a reference to the "root" node (the ancestor of all nodes), if it exists. Any node in the data structure can be reached by starting at root node and repeatedly following references to either the left, mid or right child.

DEVELOPMENT PHASES OF INTERACTIVE DIDACTIC SOFTWARE

By virtue of more complicated problems, the programmers can not type immediate the code when they know the technical requirement and program designing became an industry for witch the programming is only one of sides. Therefore was born programming engineering that boards software complexity and autonomy, the abstractions introduced in new technologies, people, programs quality and evolution management, object orientated analysing and designing.

UML [3] offers assistance for realizing of sizeable grade object orientated analysing and designing, whichever represent significant elements to the effect that obtain good and hard software.

Analysis Phase

Using Unified Modelling Language, didactic software analysis consists in achivement of use case diagram and activity diagrams. To design diagrams was used the ArgoUML software [4]. The didactic software is described in a outright and terse layout by representing the use cases [5]. Each case describes the interactions of user and software.

Use case diagram is created in an iterative approach. At the start, there were identified the actors, starting from problem formulating by identifying the role played by different persons and external resources that are implicated in interactions. Identifying the uses cases and the relations between them was based on the analysing the responsibilities accomplished by every actor and also the global specification that are referring to the functional requisites. Use case diagram design is shown in figure 1.

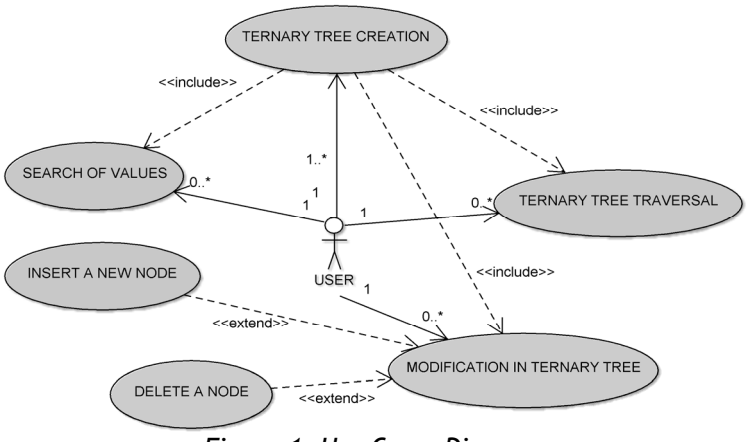

Figure 1. Use Cases Diagram

Diagram defines the area of software, allowing classic view of the proportion and sphere of the activity for the whole propagation process. This includes:

- *one actor - the user that have external entities with which the software interacts;*
- *six use cases that describe the functionality of the ineractive didactic software;*
- *relationships between user and use cases, relationships between use cases (dependency and generalization relationships), and relationships between actors.*

Beyond realizing the user needful specifications, there appears the query to look for fine points, using activity diagrams [6].

Design Phase

Object orientated procedures introduced the performing of the static structure of software using classes and relations among them. This performing is succeeded from entity-relation diagram.

Conceptual modelling allows identifying the handsomest objects for the interactive didactic software [7]. Inheritance was not used only as a generalization device, which is when derived classes are specializations of the base class. In figure 2 are presented as inheritance relationships, realization relationships, composition relationships and aggregation relationships.

We can observe that the TTreeProject classe inherit attributes and methods of the JFrame class, but implements the ActionListener interface.

TTreeParameter class inherits attributes and methods of the JDialog class, but implements the interface ActionListener. TTreeDrawing class inherit attributes and methods of the JPanel class, but implements Runnable interface, and TTreeDescendant class inherits attributes and methods of the JPanel class and implements the ActionListener interface.

In the composition relationship, unlike the aggregation relationship, the instance can not exist without the party objects. Analyzing figure 2 we can observe that an instance of TTreeDrawing type consists in one objects of TTree type, one of Graphics2D type and the other of Thread type.

Aggregation relationship is an association where it's specified who is integer and who is a part. For example, an object of TTreeVertex type or represent a part from an object of TTree type.

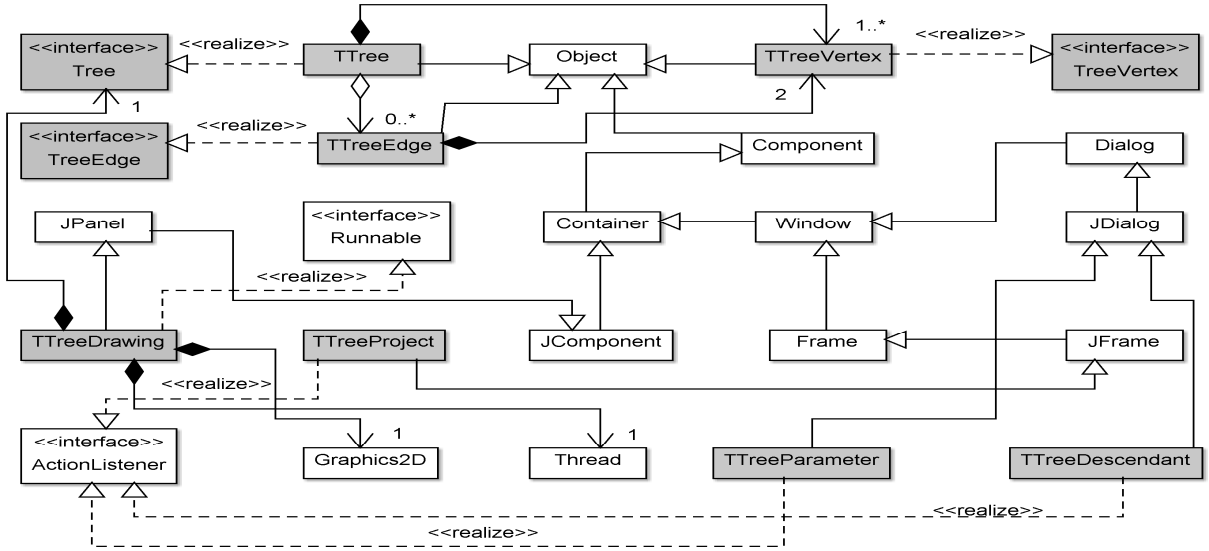

Figure 2. Classes Diagram

Sequence diagram emphasizes the temporal semblace [8], being suitable for real-time specifications and complex scenarios. These diagrams determine the objects and classes involved in a scenario and sequence of messages sent between objects necessary to execute script functionality.

Diagram presented in figure 3 shows the interactions between objects that are aimed to create a ternary tree. We can observe that there are interactions between 14 objects of which the objects of TTreeProject, Checkbox, TTreeDrawing, JButton, JTextFiled and Graphics2D type are already created, and objects of TTree, TTreeParameter, TTreeVertex and TTreeDescendant type will instantiate during interactions.

Diagram presented in figure 4 shows the interactions between objects that are aimed to ternary tree traversal in inorder. We can observe that there are interactions between 7 objects of which the objects of TTreeProject, JButton, JTextArea, TTreeDrawing and TTree type are already created, and objects of Thread and Vector<TTreeVertex> type will instantiate during interactions.

Implementation Phase

Component diagram [9] is suchlike to packages diagram, allowing visualization of how the software is divided and the dependencies amongst modules. Component diagram put emphasis on software physical elements and not on the logical elements like in case of packages.

The diagram in figure 5 describes the collection of components that together provide system functionality.

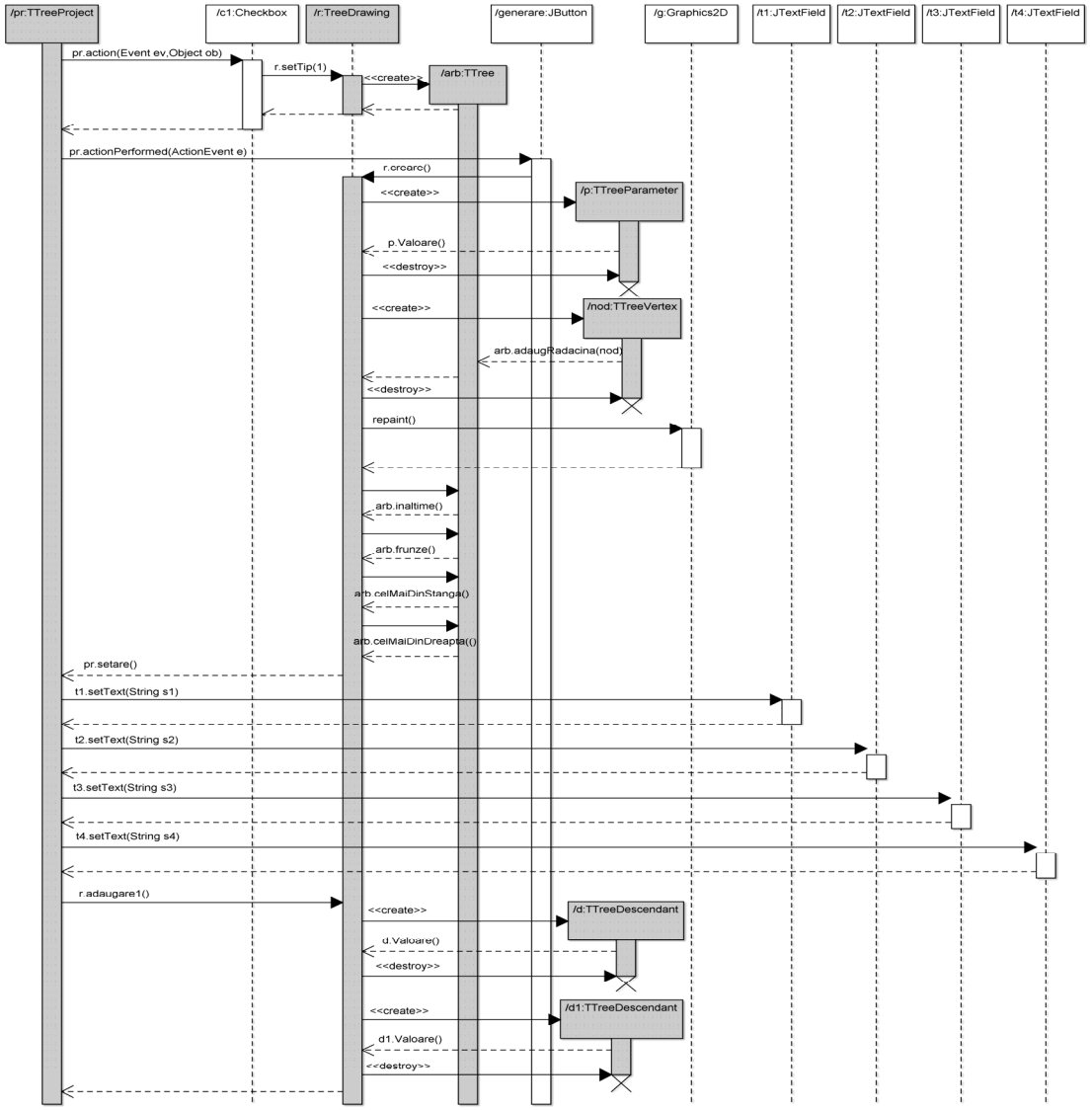

Figure 3. Sequence diagram for create of a ternary tree

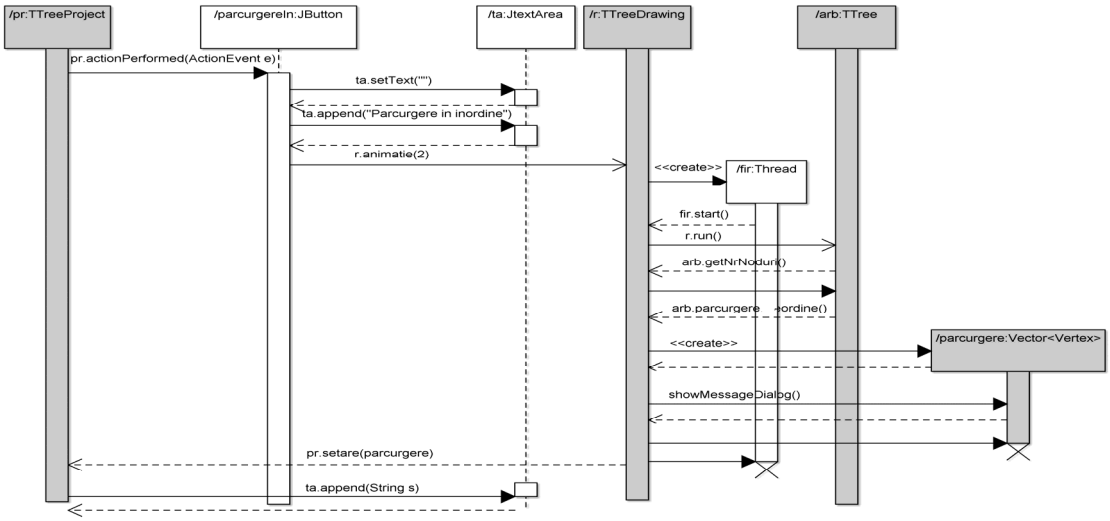

Figure 4. Sequence diagram for ternary tree traversal in inorder

Central component of the diagram is TTreeProject.class, a component obtained by transforming by the Java compiler into executable code of the TTreeProject.java component. As can be seen that component interacts directly with component TTreeDrawing.class. Component TTreeDrawing.class that is obtained by Java compiler from component TTreeDrawing.java in executable code interactions directly with components TTreeDescendant.class, TTree.class and TTreeParameter.class. Component TTree.class interactions directly with components Tree.class, TTreeVertex.class and TTreeEdges.class.

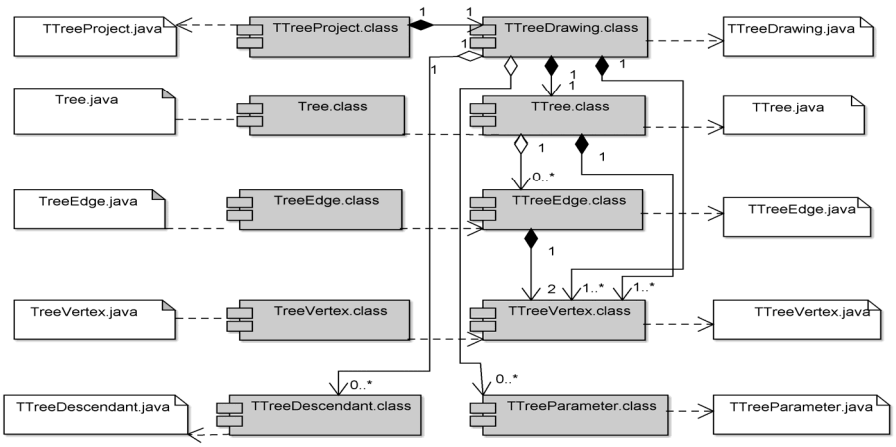

Figure 5. Component diagram

GRAPHICAL INTERFACE

The interactive software was implemented in Java [10] as independent application. By using visual simulations in computer assisted learning the efficiency of learning is increased. Starting from specified requisites in uses cases diagram (figure 1) it was designed graphical user interface of the didactic software which is presented in figures 6.

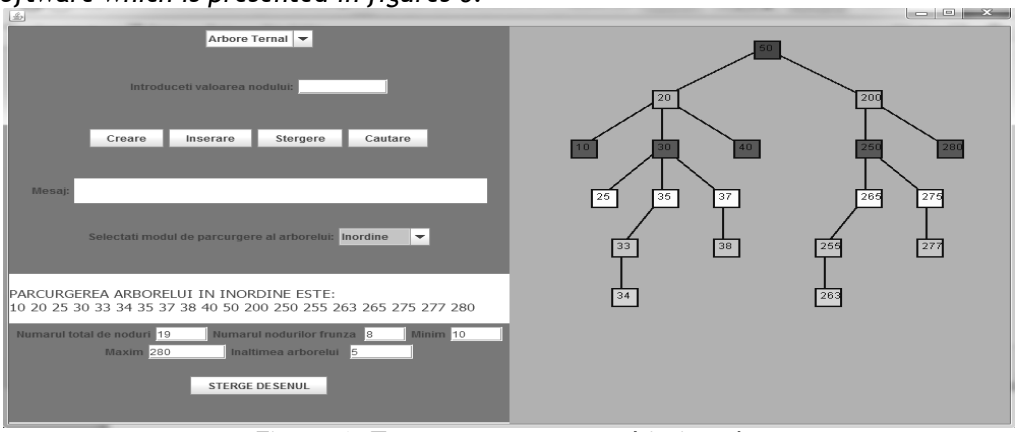

Figure 6. Ternary tree traversal in inorder

The main page of the application contains buttons for selecting the following options: ternary tree creation, ternary tree traversal in preorder, ternary tree traversal in postorder, ternary tree traversal in inorder and search a value in ternary search tree.

CONCLUSIONS

About performing of diagrams for all three phases: analysis, design and implementation, the interactive didactic software has been presented in a outright and terse layout. The use of the Unified Modeling Language for the achievement of the diagrams is characterized by stringent syntactic, rich semantic and visual modeling sustentation.

The diagrams were made using a further approach, multidisciplinary of the informatics application, encompassing both modern pedagogy methodologies and discipline specific components. The nexus of teaching activities and scientific aims and objectives was established about the design of the new methods and the assimilation of further paths, capable of enhancing school showing, enabling students to acquire the knowledge and techniques required and apply them in optimal conditions.

REFERENCES

[1] V. Voloshin, introduction to Graph Theory, Nova Science Publishers Inc., 2009
[2] R. Merris, Graph theory, John Wiley & Sons Ltd, 2000
[3] K. Lunn, Software Development with UML, Palgrave Macmillan, 2002
[4] http://argouml.tigris.org
[5] P. Stevens, Using UML: Software Engineering with Objects and Components, Addison-Wesley Educational Publishers Inc., 2005
[6] S. Bennett, J. Skelton, K. Lunn, Schaum's Outline of UML, McGraw Hill, 2004
[7] L. Craig, Applying UML and Patterns, Pearson Education, 2004
[8] G. Booch, J. Rumbaugh, I. Jacobson, The Unified Modeling Language User Guide, Addison Wesley, 1999
[9] M. Fowler, K. Scott, UML Distilled: A Brief Guide to the Standard Object Modeling Language, Addison Wesley, Readings MA, USA, 2000
[10] K. Arnold, J. Gosling, The Java Programming Language, Addison-Wesley, 2005