ANNALS of Faculty Engineering Hunedoara
– International Journal of Engineering
Tome XIII [2015] – Fascicule 4 [November]
ISSN: 1584-2673 [CD-Rom; online]
a free-access multidisciplinary publication of the Faculty of Engineering Hunedoara

[1.] Anca-Elena IORDAN

INTERACTIVE SOFTWARE DESIGNED FOR THE STUDY OF THE PARABOLOID

[1.] University Politehnica Timisoara, Faculty of Engineering Hunedoara, ROMANIA

Abstract: This article illustrates the required phases for the achieving of the interactive software used for the study of the elliptic paraboloid and hyperbolic paraboloid. The modelling of the interactive software is accomplished by specific UML diagrams representing the phases of analysis, design and implementation. The analysis phase is characterized by two types of UML diagrams: use-case diagram and activity diagrams. The design phase is characterized by three types of diagrams: class diagram, state diagrams, and interaction diagrams. Implementation phase it corresponds the component diagram. By achieving these types of diagrams, the interactive software thus is described in an obvious and objective manner, without ambiguity. The implementation of the software was realised through the Java programming language. The main options of the software allow drawing a paraboloid, the normal line and tangent plane to a point, well as the application of isometry.
Keywords: Interactive Software, Java, UML, Elliptic Paraboloid, Hyperbolic Paraboloid

1. INTRODUCTION

In mathematics, a paraboloid is a quadric surface. There are two kinds of paraboloids: elliptic and hyperbolic.
The elliptic paraboloid is shaped like an oval cup and can have a maximum or minimum point. In a suitable coordinate system with three axes Ox, Oy, and Oz, it can be represented by the equation [1]:

$$\frac{(x-x_c)^2}{a^2}+\frac{(y-y_c)^2}{b^2}=2\cdot p\cdot(z-z_c) \tag{1}$$

where *a* and *b* are constants that dictate the level of curvature in the xOz and yOz planes respectively. This is an elliptic paraboloid which opens upward for p>0 and downward for p<0. The point of coordinates (x_c, y_c, z_c) represents the paraboloid vertex. For p = 1 it obtains the following parametric equations of elliptic paraboloid [2]:

$$\begin{cases} x(u,v)=x_c+a\cdot\sqrt{2\cdot u}\cdot\cos v \\ y(u,v)=y_c+b\cdot\sqrt{2\cdot u}\cdot\sin v;u\in R_+,v\in[0,360]. \\ z(u,v)=z_c+u \end{cases} \tag{2}$$

The hyperbolic paraboloid is a doubly ruled surface shaped like a saddle. In a suitable coordinate system, a hyperbolic paraboloid can be represented by the equation:

$$\frac{(x-x_c)^2}{a^2}-\frac{(y-y_c)^2}{b^2}=2\cdot p\cdot(z-z_c). \tag{3}$$

For p>0, this is a hyperbolic paraboloid that opens down along the x-axis and up along the y-axis. For p = 1/4 it obtains the following parametric equations of hyperbolic paraboloid [3]:

$$\begin{cases} x(u,v)=x_c+a\cdot\sqrt{u/2} \\ y(u,v)=y_c+b\cdot\sqrt{v/2} \\ z(u,v)=z_c+u-v \end{cases} \tag{4}$$

2. SOFTWARE ANALYSIS

From the perspective of unified modelling language, interactive software analysis includes the representation of two types of diagrams: the use-case diagram and activity diagrams.

The use-case diagram [4] offers simplified and graphical representation of what the interactive software must really do. Use-case diagram is based upon functionality and thus will focus on the "what" offers the interactive software and not the "how" will be realized. The use case diagram corresponding to this software, presented in figure 1, includes: an actor, seven use cases that describe the functionality of the software and relationships among them.

For every use-case of the previous diagram was achieved an activity diagram. Each activity diagram specifies processes and algorithms used to achieve the purpose specified by the use-case. In terms of UML, activity diagrams [5] are meant to model both computational processes and those of the organization. Activity diagrams show the general flow of control. In figure 2 is rendered the activity diagram corresponding to the use-case "Drawing of geometric elements specific to the paraboloid".

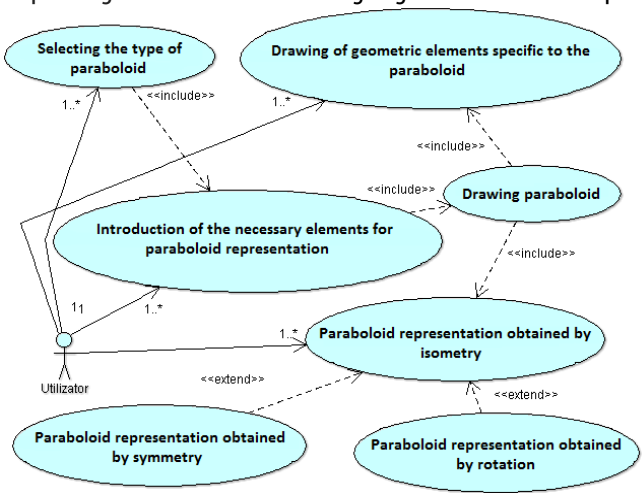


Figure 1. Use-case diagram

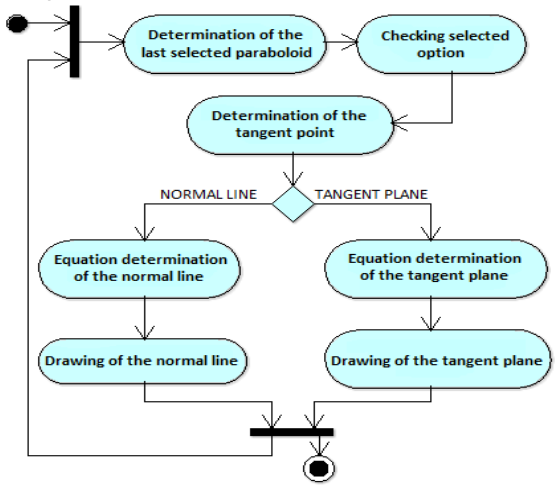


Figure 2. Activity diagram

3. SOFTWARE DESIGNING

Class diagram [6] represents the main block of object-oriented modeling. It is used both for static conceptual modeling software as well as detailed modeling aimed at translating in programming source code. For achieving the objectives of software were identified required classes and the relationships between them.

Figure 3 presents the inheritance, composition and aggregation relationships used. It may be noted that all attributes and methods of the *Element3D* class will apply to the all its derived classes: *Dreapta3D, Plan3D, Punct3D,* and *Paraboloid*. Similarly, that all attributes and methods of the *Izometrie3D* class will apply to the all its derived classes: *Rotatie3D* and *Simetrie3D.* This class diagram shows contain specific classes to the interactive software as well as existing classes and interfaces from Java.

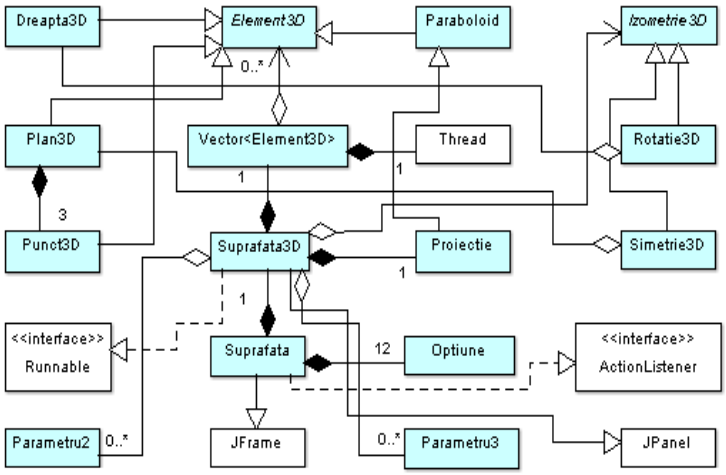


Figure 3. Class diagram

A sequence diagram [7] presents the interactions between objects arranged in a temporary sequence. He describes the objects involved in interaction and the sequence of messages exchanged between the needed objects to achieve the desired purpose. Sequence diagrams are usually associated with a use-case. The diagram illustrates in figure 4 shows the interactions between objects, which have as purpose the drawing the tangent plane of a paraboloid. One can notice that there are interactions between nine objects, out of which the objects of *Suprafata3D*, *Vector<Element3D>* and *Graphics2D* type are already created, and the other two objects: *Parametru2, Punct3D, Plan3D* and *Paraboloid* type will be instantiate during the interactions.

At first the execution control is taken by the object of *Suprafta3D* type. Following an event that interacts with the *Suprafta3D* object allows the creation of an instance of *Parametru2* class. After getting the parameters that characterize the paraboloid, control is returned to the *Suprafata3D* object. Following is created an instance of *Punct3D* class. The control will be given to the object of *Suprafata3D* type that will destroy the object of *Parametru2*. We can observe that lifeline of the *Parametru2* object is interrupt, by marking an X, the message appears bearing the stereotype *<<destroy>>*. Control is transmitted to the object of *Vector<Element3D>* type to obtain the last drawing paraboloid. The control will be given to the object of *Suprafta3D* type that will create the object of *Plan3D* type. Through interaction with *Graphics2D* object will redraw the tangent plane.

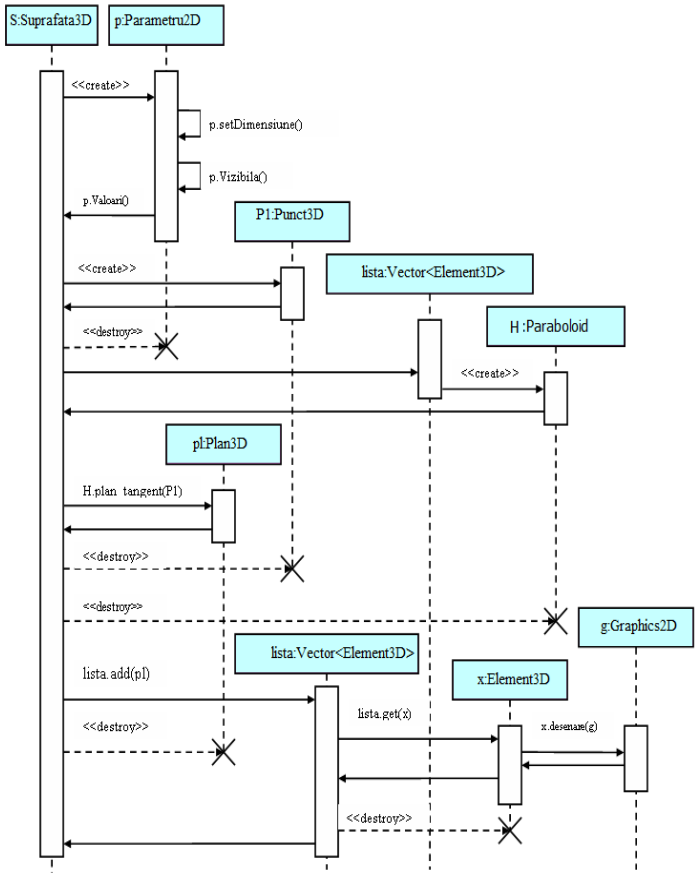


Figure 4. Sequence diagram

Collaboration diagrams, on the other side, concentrate on the relationships among the objects [8]. They are very useful for visualizing the way several objects collaborate to achieve the intended purpose and for comparing a dynamic model with a static model (figure 5). Collaboration and sequence diagrams render the same information, and can be transformed into one another without difficulty. The selection of the two types of diagrams depends upon what the designer wants to make visually apparent.

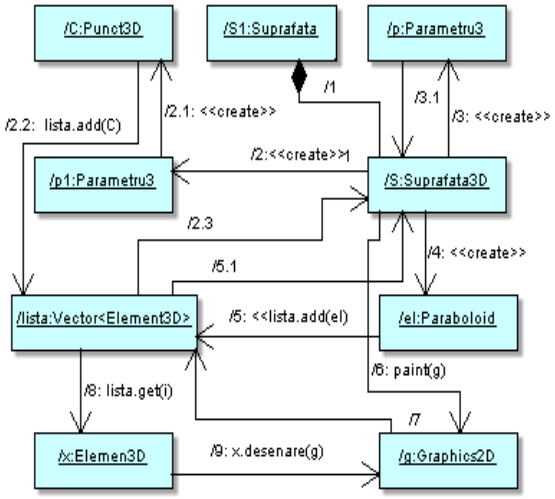


Figure 5. Collaboration diagram

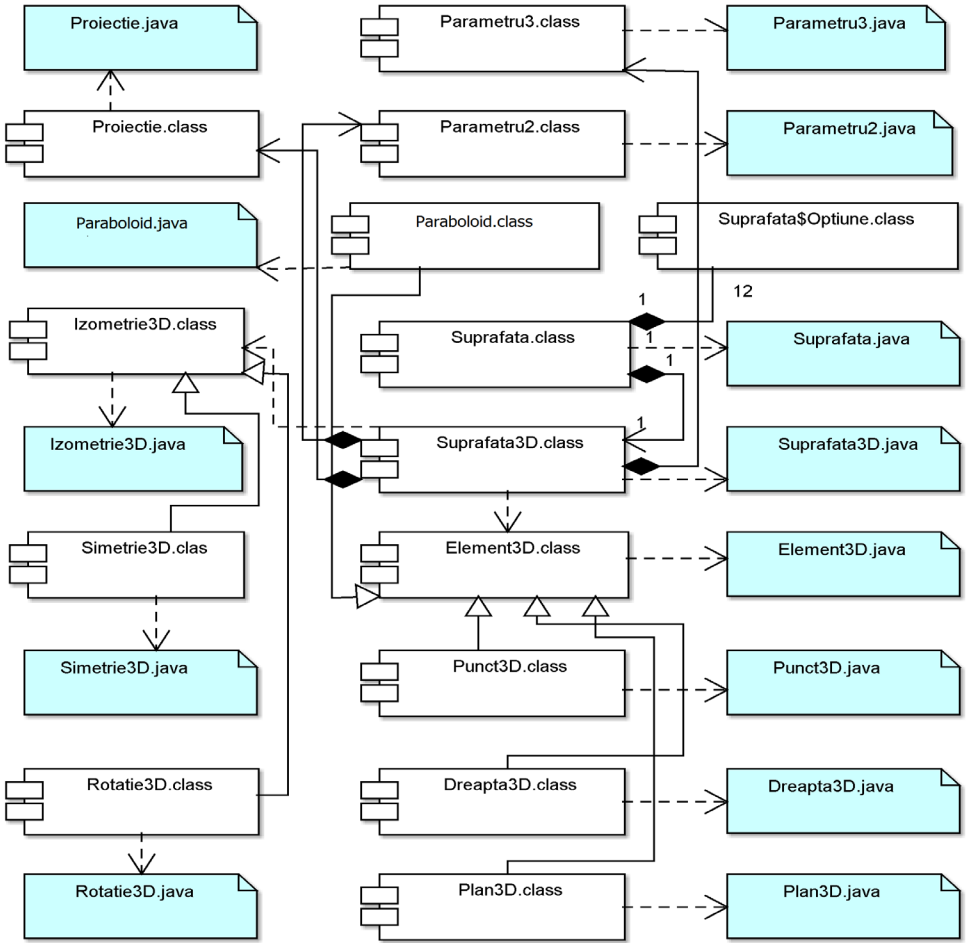


Figure 6. Component diagram

4. SOFTWARE IMPLEMENTATION

The component diagram [9] enables the visualization modules that compose the software and the dependencies between them. The diagram that is shown in figure 6 describes the collection of components that all together ensure the functionality of the interactive software.

Central component of the diagram is *Suprafata.class,* a component obtained by transforming by the Java compiler into executable code of the *Suprafata.java* component. As can be seen that component interacts directly with components *Suprafata3D.class*. This component interacts with the next three components: *Element3D.class*, *Izometrie3D. class* and *Parametru2.class*.

5. GRAPHICAL USER INTERFACE

The interactive software is accomplished using the Java object-oriented programming language [10, 11]. From the application window it can selected by a main menu the following options:

- the graphic representation of elliptic paraboloid;
- the graphic representation of hyperbolic paraboloid;
- the plan tangent and the normal line to the paraboloid into a point;
- the paraboloid obtained by applying an isometry: rotation or symmetry.

The equation of the tangent plane to a quadratic surfaces in a point $M(x_0,y_0,z_0)$ is:

$$\begin{vmatrix} x-x_0 & y-y_0 & z-z_0 \\ x'_u & y'_u & z'_u \\ x'_v & y'_v & z'_v \end{vmatrix} = 0 \tag{5}$$

For tangent plane determination in the point $M(x_0,y_0,z_0)$ at a paraboloid it starts from the parametric equations of the paraboloid. The equation of the tangent plane in a point $M(x_1(u,v),y_1(u,v),z_1(u,v))$ at an elliptic paraboloid is:

$$p\cdot x+q\cdot y+r\cdot z-(p\cdot x_1+q\cdot y_1+r\cdot z_1)=0,$$

where

$$\begin{cases} p=b\cdot\sqrt{2pu}\cdot\cos v \\ q=-a\cdot\sqrt{2pu}\cdot\sin v. \\ r=abp \end{cases} \tag{6}$$

The equation of the tangent plane in a point $M(x_1(u,v),y_1(u,v),z_1(u,v))$ at an hyperbolic paraboloid is:

$$p\cdot x+q\cdot y+r\cdot z-(p\cdot x_0+q\cdot y_0+r\cdot z_0)=0,$$

where

$$\begin{cases} p=\dfrac{-b}{2\cdot\sqrt{2\cdot v}} \\ q=\dfrac{a}{2\cdot\sqrt{2\cdot u}}. \\ r=\dfrac{a\cdot b}{8\cdot\sqrt{u\cdot v}} \end{cases} \tag{7}$$

In figure 7 it shows the tangent plane characteristic to an elliptic paraboloid, when the tangent point was determined using an instance of *Parametru2* class, applying the default projection (α=-135, β=-30, γ=90). If it is desired the viewing using the projection angles (α = -50, β = 45, γ = 160), the elliptic paraboloid with the tangent plane will have representation shown in figure 8.

The equation of the normal line to a quadratic surfaces in a point $M(x_0,y_0,z_0)$ is:

$$\frac{x-x_0}{\dfrac{D(y,z)}{D(u,v)}}=\frac{y-y_0}{\dfrac{D(z,x)}{D(u,v)}}=\frac{z-z_0}{\dfrac{D(x,y)}{D(u,v)}} \tag{8}$$

For normal line determination in the point $M(x_0,y_0,z_0)$ at a paraboloid it starts from the parametric equations of the paraboloid. The equation of the normal line in a point $M(x_1(u,v),y_1(u,v),z_1(u,v))$ at an elliptic paraboloid is:

$$\frac{x-x_0}{-b\cdot\sqrt{2\cdot u}\cdot\cos v}=\frac{y-y_0}{-a\cdot\sqrt{2\cdot u}\cdot\sin v}=\frac{z-z_0}{a\cdot b} \tag{9}$$

The equation of the normal line in a point $M(x_1(u,v),y_1(u,v),z_1(u,v))$ at an hyperbolic paraboloid is:

$$\frac{x-x_0}{\frac{-b}{2\cdot\sqrt{2\cdot v}}}=\frac{y-y_0}{\frac{a}{2\cdot\sqrt{2\cdot u}}}=\frac{z-z_0}{\frac{a\cdot b}{8\cdot\sqrt{u\cdot v}}} \quad (9)$$

In figure 9 it shows the normal line characteristic to an elliptic paraboloid, when the tangent point was determined using an instance of *Parametru2* class, applying the default projection (α=-135, β=-30, γ=90). If it is desired the viewing using the projection angles (α = -50, β = 45, γ = 160), the elliptic paraboloid with the normal line will have representation shown in figure 10.

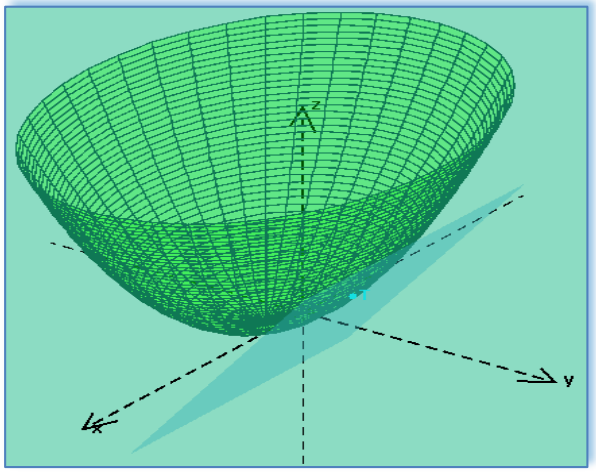


Figure 7. Tangent plane of an elliptic paraboloid

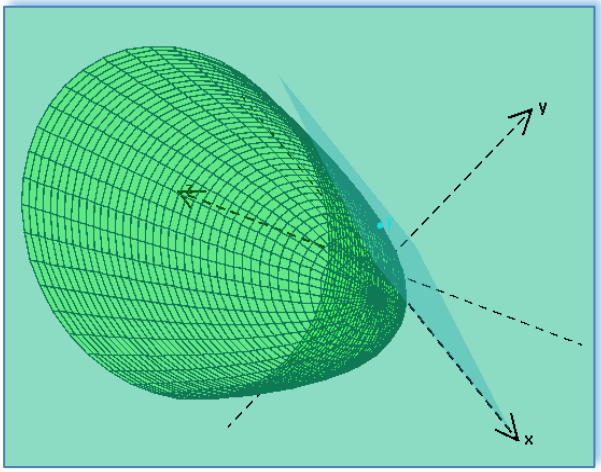


Figure 8. Tangent plane of an elliptic paraboloid

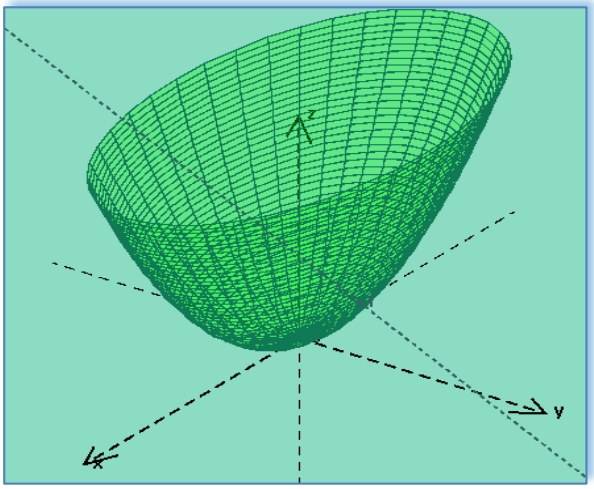


Figure 9. Normal line of an elliptic paraboloid

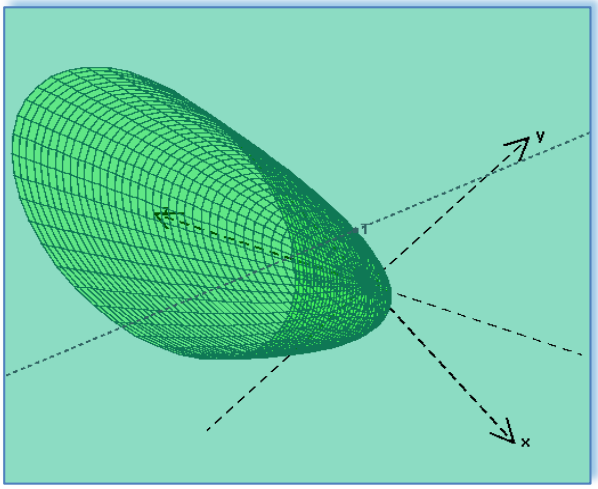


Figure 10. Normal line of an elliptic paraboloid

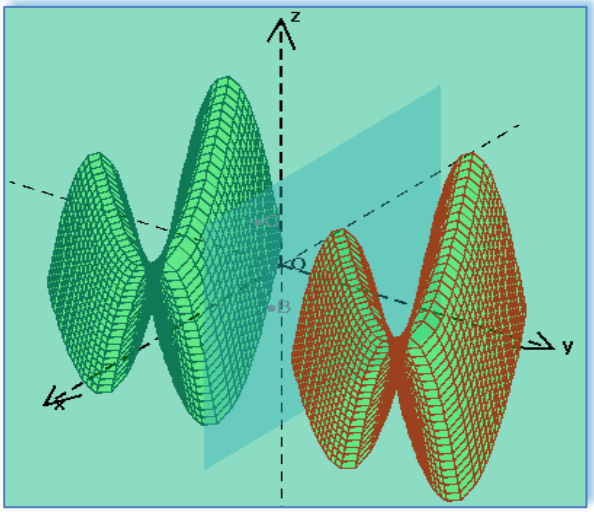


Figure 11. Hyperbolic paraboloid obtained by symmetry

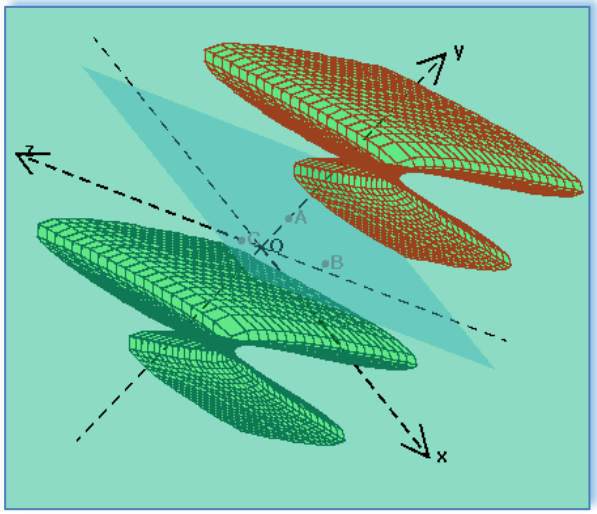


Figure 12. Hyperbolic paraboloid obtained by symmetry

In figure 11 is represented a hyperbolic paraboloid obtained by applying a symmetry about a plane, using the default projection (α=-135, β=-30, γ=90). If it is desired the viewing applying the projection angles (α = -50, β = 45, γ = 160), the hyperbolic paraboloid will have representation shown in figure 12.

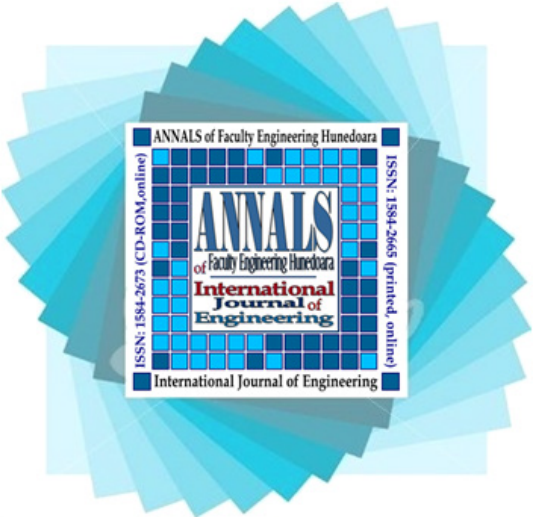
6. CONCLUSIONS

Because the representation of the diagrams corresponding to all three phases: analysis, design and implementation, the interactive software has been described in an obvious and objective manner, without ambiguity. The use of the unified modelling language for the achievement of the diagrams is characterized by rigorous syntactic, rich semantic and visual modelling support.

The diagrams have been achieved using a new approach, multidisciplinary of the interactive software, grouping both modern pedagogy methods and discipline-specific components. The connection between teaching activities and scientific goals and objectives was established through the development of the new methods and the assimilation of new means, capable of enhancing school performance, enabling students to acquire the knowledge and techniques required and apply them in optimum conditions.

REFERENCES

[1]. J. Gray, M. Esplen, D. Brannan, *Geometry,* Cambridge University Press, 2011
[2]. W. McCrea, *Analytical Geometry of Three Dimensions*, Dover Publications, 2006
[3]. A. Church, *Elements of Analytical Geometry*, 2014
[4]. M. Fowler, K. Scott, *UML Distilled: A Brief Guide to the Standard Object Modelling Language*, Pearson Education, 2003
[5] J. Hunt, *Guide to the Unified Process featuring UML, Java and Design Patterns*, Springer London Ltd, 2014
[6] K. Lunn, J. Skelton, S. Bennett, *Schaum's Outline of UML*, McGraw-Hill Education, 2004
[7] B. Bruegge, A. Dutoit, *Object Oriented Software Engineering Using UML, Patterns, and Java*, Pearson Education, 2013
[8] A. Dennis, B. H. Wixom, D. Tegarden, *Systems Analysis and Design with UML*, John Wiley & Sons Ltd, 2012
[9] P. Jalote, *A Concise Introduction to Software Engineering*, Springer-Verlag, 2008
[10] J. Bryant, *Java 7 for Absolute Beginners*, Springer, 2012
[11] J. Gosling, B. Joy, G. Steele, G. Bracha, A. Buckley, *The Java Language Specification*, Oracle, 2013



ANNALS of Faculty Engineering Hunedoara – International Journal of Engineering