UML and JAVA as Effective Tools for Implementing Algorithms in Computer Graphics

Camil Sukic[1], Muzafer Saracevic[2]

[1]*Department for Computer Sciences, University of Novi Pazar, Serbia*
[2]*Faculty of Science and Mathematics, University of Niš, Serbia*

Abstract: **In this paper we presents process of UML modeling for algorithm which is used in computational geometry and computer graphics. Implementation of algorithm for triangulation polygon is given in programming language JAVA. The main motivation is to present a way of solving problems through so-called visual planning and programming using object-oriented concepts. Dynamic and static model processing show the problem and give the plan for solving and implementation in some of objective-oriented program language. In one section of this paper we will list some research of authors and their results in the application of programming languages in the field of computer graphics. We specify reason for programming in Java in computer graphics and compared with other programming languages.**

Keywords: **Computer Graphics, Triangulation Polygon, Java, UML, Computational Geometry**

1. Introduction

The important thing in computer graphics is a triangulation of the polygon. A triangulation of a simple polygon assumes the decomposition of the polygon interior into triangles, where mutually internal diagonals do not intersect (Figure 1, Left). Triangulation allows that from the set of points gets three-dimensional view of objects. Triangulation also provides a mechanism for "glazing" 3D figures (Figure 1, Right) which is very important for speed, quality and resolution of the objects in computer graphics.

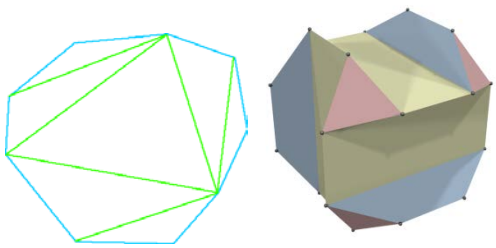


Figure 1. Proces of triangulation and 3D view

The main goal of this paper is to simplify the process of solving the algorithm. The analysis and problem solving in this way has many advantages just because it provides a clear definition of requirements and specific plan that we will later use to create applications. Object - oriented analysis and design (*OOAD*) of algorithms is a software engineering approach that models a system as a group of interacting objects. The advantage of object-oriented modeling is the fact that objects encapsulate state expressed by attributes and behavior specified methods or operation and they are able to communicate by sending and retrieving messages. Using the advantages of object-oriented programming and modeling techniques it is possible to build a data model of custom features.

2. Reason for java programming and comparison with other languages

Java as a programming language can be characterized as: simple, high-performance, object-oriented, multithreading, dynamic, distributed, secure, portable and etc. The advantage of Java is that the most programming languages or interprets or compiles to run on the computer, while Java and compiles and interprets. We specify some other programming languages that can find applications in computer graphics that can be used to solve similar algorithms (*Python*, *C++*, *PHP*).

Python is also object-oriented interpreted programming language. Python was created in response to, and improving the compiled languages such as C++ and Java. The main advantage of Python is his simple syntax and simplicity, which allows developers to focus on the problems rather than on programming. The disadvantage is speed, in case you are doing some things that require lots of processing power. The first connotation in comparison Python with Java is usually the ease of work. Compared to Java, Python applications that implement the same algorithm are much shorter. Productivity is at a very high level, primarily because of the syntax and especially because of the large number of ready-made libraries and modules that come with the standard distribution. Runtime implementation of

Python language is available on many platforms, which means that written code will work without modification on these platforms. Unlike Java, Python's syntax does not require encapsulation of the class and do not support real interfaces but it supports multiple inheritance and functional programming.

JAVA computational geometry library contains an implementation of major *Computational Geometry algorithms in Java* [10], pretty much as CGAL does for the C++ language. In recent years, Java has significantly improved, particularly since the appearance of Java 2 standard. Now under this programming language, there are many opportunities to create applications with interactive graphical user interface (GUI), a detailed image processing and programming of graphic elements.

Comparison of Python and PHP language is probably the most difficult. Python have much wider perspective than PHP. Only after development of PHP-GTK (*Graphic – Tool – Kit*) module, PHP supports programming interface. Python in purpose of use is much closer to Java than PHP. Java is primarily used to create desktop applications/software while PHP is used to make web applications. Of course both support web and desktop applications but you would find very few who interchange those two. Java is complex while PHP is a bit simpler. Java has a massive library while PHP has a small but quite useful library. Generally speaking JAVA developer earns more than PHP counterpart. C++ is more consistent as a language and during learning is clearly see the difference between the very languages and libraries that come with it. Computational Geometry Algorithms Library (CGAL) in C++ is used in various areas needing geometric computation, such as: *computer graphics, scientific visualization, computer aided design and modeling, geographic information systems, molecular biology, medical imaging, robotics and motion planning, mesh generation and numerical methods* [13].

2.1 Related works and research in solving of algorithms for computational geometry

In this section we will list some works and research of authors and their results in the application of programming languages in the field of computer graphics. Authors the paper [1] discuss the design of several Java applets that visualize how the Voronoi diagram for triangulation polygon of *n* points continuously changes as individual points are moved across the plane, or as the underlying distance function is changed. Moreover, authors report on some experiences made in using these Java applets in teaching and research. In paper [2] specified is studied profoundly that the compound 3D visualization modeling system can be built up with the open-sourced graphic libraries PYOpenGL and VTK (*Visualization Tool-kit*)**.** In this process, the computational models written by Python. Then based-on this data structure, spatial discrete points Delaunay triangulation in any plane is accomplished through the setting of projection plane.

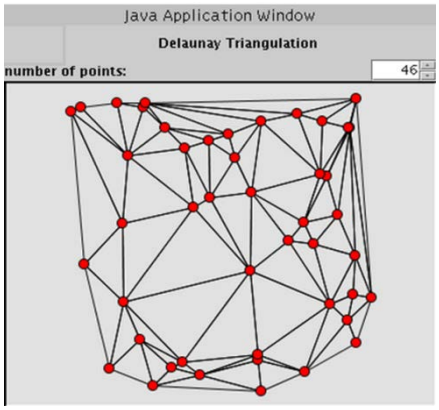


Figure 2. Java application for Delaunay Triangulation

Taking full advantage of the topology characteristic, a kind of algorithm which can search interrelated triangles, segments and vertexes efficiently is obtained and the object of inserting any constraint is reached.

In paper [3] specified PHP image processing and GD graphic extension, the energies calculated in active contour algorithm are adapted so the continuous energy is approximated with the distance between two consecutive points, the curvature energy is normalized using the same distance and the vicinity of each point are calculated until an edge is detected and that point will not move anywhere. The paper [4] is the first attempt to compare Web services engines implemented in PHP and Java. In this paper authors perform a thorough study of the capability of PHP as a Web service engine in both qualitative and quantitative aspects while comparing it with other Web service engines implemented in Java [12].

The paper [5] describes a new working implementation of the Python language, built on top of the Java language and run-time environment. Implementing Python in Java has a number of limitations when compared to the current implementation of Python. The advantages of Java as an implementation language include portability of binary executables, object-orientation in the implementation language to match object-orientation in Python and the ability to automatically generate wrapper code for arbitrary Java libraries.

In paper [6] specified the two language approach to software development has been investigated by several language designers. Java is a static type-checked language which offers performance,

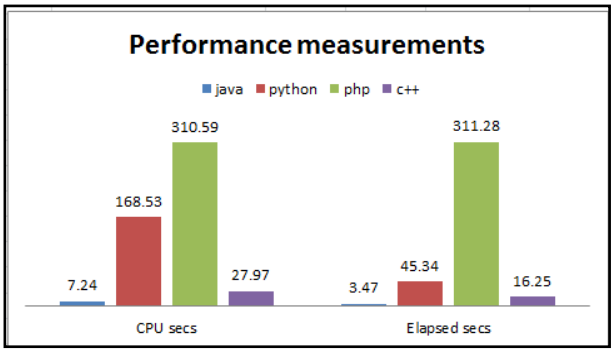
robustness and modularity as such, while Python is a run-time type-checked language which offers rapid prototyping, dynamic run-time modification, and delayed evaluation. In [11] is given advantages programming language C++ in computational geometry and computer graphics.

On portal - *The Computer Language Benchmarks*[1] is stated testing results of several programming languages for more algorithms. Some of them are: *n-body, spectral norm, thread ring, reverse-complement and etc*. In this part of the paper is listed research which was conducted for the algorithm *BinaryTrees*, in order to compare testing results for the same programming languages for our algorithm. Each table row shows performance measurements for Java, Python, PHP and C++ program for the same algorithm for *BinaryTrees*.

n	CPU secs				Elapsed secs			
	java	**python**	**php**	**c++**	**java**	**python**	**php**	**c++**
12	0.16	0.95	1.14	0.21	0.12	0.98	1.15	0.16
16	1.01	20.88	35.26	4.06	0.57	5.73	35.29	2.33
20	20.54	483.8	895.4	79.6	9.73	129.3	897.4	46.3

Table 1. Performance measurements for algorithm

As for the CPU Load (testing on this portal is carried out on Intel® Q6600 quad-core) and in the most demanding case where is n=20, highest CPU occupation is registered by Python (93% , 92% , 98%, 93%). The values in parentheses refer to the core CPU. In the case for C++ (27% ,98%, 25%, 24%), Java (55%, 27% , 76% , 57%) and PHP (7% ,81% ,10% ,3%).



Graph 1. The ratio of all four programming languages in both criteria (average value)

3. Triangulation convex polygon in JAVA and UML environment

In this part of the paper we present procedure of object-oriented analyze and design for algorithm for triangulation polygons. The tool that is used for object-oriented analysis is the UML.
UML standard that is applied to the object-oriented approach provides appropriate views of the system, so that are in all terms system can be described from a static (structural) and dynamic aspect. It is used to design software which needs a plan, offers the possibility of visualization in multiple dimensions and levels of detail and is suitable for upgrading old systems. Generally, this is a good way to resolve problems, because the UML describes the source code, models help to visualize the system as it is or what it should be and allow you to determine the structure and behavior of the system.
This paper presents modeling first through the static and then dynamic diagrams. This is a good way to resolve because the UML describes the source code, models help to visualize the system as it is or what it should be and allow you to determine the structure and behavior of the system. Models document the decisions that we were making and provide solutions to guide us during the construction of the specific applications. In the following we present some of the UML diagrams that are crucial to the process of implementation of the algorithm, specifically in this case in the Java programming. The complete analysis is developed through three models of the system:

A. static,
B. dynamic and
C. physically.

In our project we develop:

- 9 use case diagrams,
- 17 activity diagrams,
- 1 class diagram,
- 9 state diagrams,
- 18 sequence diagrams and
- 7 communication diagrams.

Implementation of algorithms for triangulation convex polygon is realized through the same class (Diagram 1). Class diagram describes structure of the system. Classes are being modeled and they connect mutually through class diagrams and classes are described by names, attributes and operations. Connections provide communication between classes. In the shown class diagram two types of connections are defined (generalization and dependency).

Generalization is a kind of dependency which is

[1] (http://shootout.alioth.debian.org/),

established between subclasses and main class.

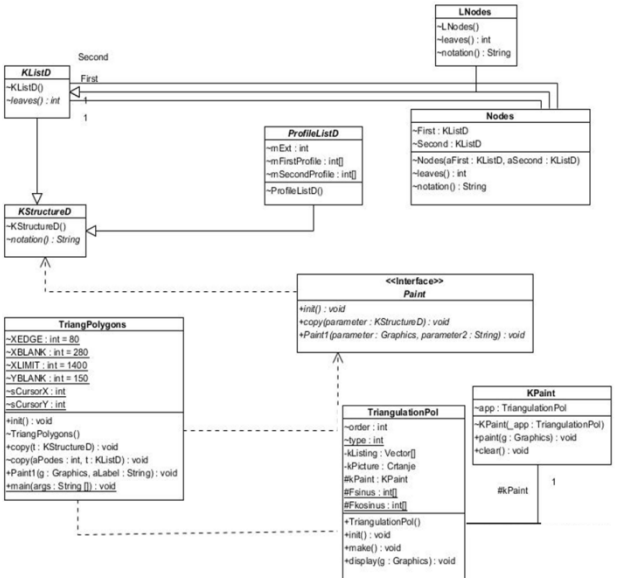


Diagram 1. UML Class Diagram

Class diagram representing all the classes that participate in creating applications for the triangulation of the polygon. *TriangulationPol* standard class inherits Panel class and contains the *init()* method, which is responsible for defining graphic elements.

Activity diagram (Diagram 2) consist of rows of sub activities (action) which gives suitable value. Order of performing actions is defined so after one action is being finished the other one continues but there is a condition that the suitable value is obtained. All activities are further decomponated into sub activities.

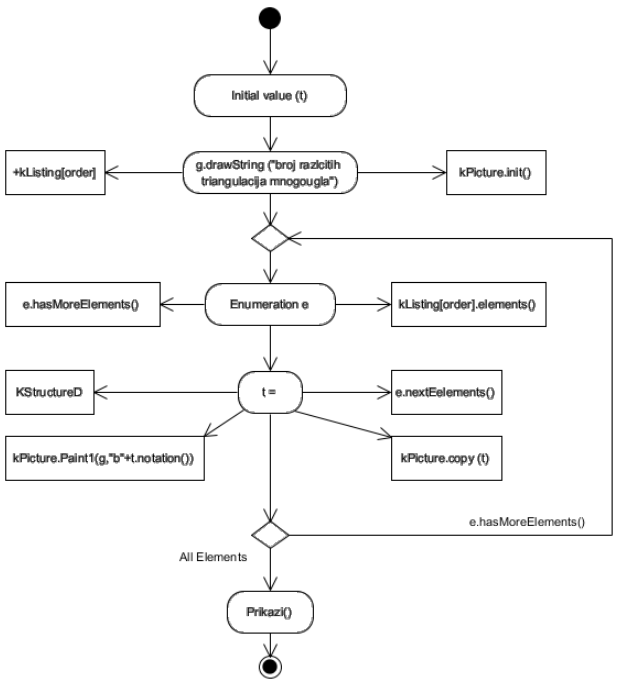


Diagram 2. Activity for display method in Java application

Java source code for method for displaying based on UML diagram activity:

```
public void display(Graphics g){
KStructureD t;
g.drawString("number of triangulation on
display=" + kListing[order].size(),480,20);
kPicture.init();
for(Enumeration e=kListing[order].elements();
e.hasMoreElements();){
   t = (KStructureD) e.nextElement();
   kPicture.copy( t );
   kPicture.Paint(g);}
  }
```

TriangulationPol class is responsible for drawing a convex polygon triangulation. This class provides the verification of all the vertices of the polygon and to take order those combinations of internal diagonal which form the triangles within the polygon under the condition that do not intersect (Algorithm).

Algorithm 1: Finding all internal diagonal of simple convex polygons [9]

- Step 1: Set the counter $i = 1$,
- Step 2: *i-th* point connect with (*i*+2)-*nd* point,
- Step 3: Is the new diagonal internal?
  - **Yes:** Add it in the list and eliminate (***i***+1)-*th* point of the polygon.
  - **No:** ***i***=i+1
- Step 4: Return to step 2

The method *make()* creates an internal diagonal. The method *Display()* ensures that any combination of triangulation and the records appear on the panel. Class *TriangPolygons* implements class *TriangulationPol* and contains the method *Paint()* which is tasked to do the drawing of a combination of triangulation of the polygon.

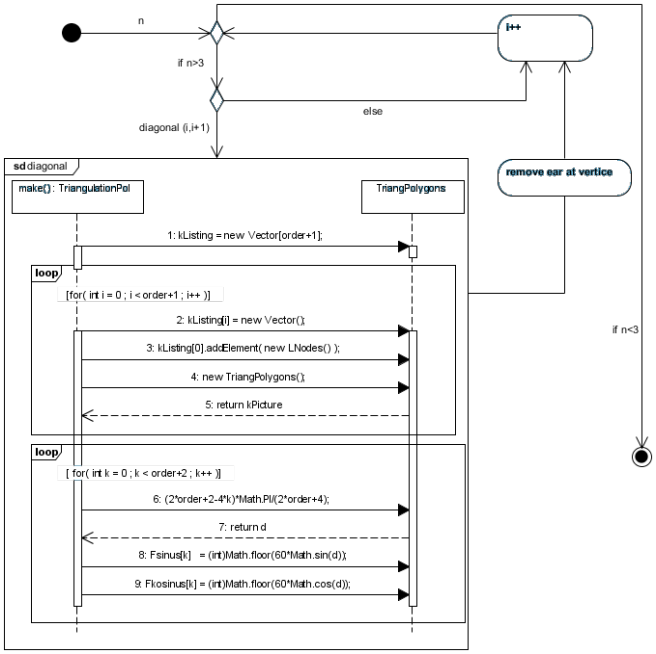


Diagram 3. Activity Diagram for *make* method

Java source code for *make* method based on UML diagram activity:

```
public void make() {
kListing = new Vector[order+1];
for( int i = 0 ; i < order+1 ; i++ ) kListing[i]
= new Vector();
kListing[0].addElement( new LNodes() );
kPicture = new TriangPolygons();
for( int k = 0 ; k < order+2 ; k++ ) {

//for diagonals
double d = (2*order+2-4*k)*Math.PI/(2*order+4);
Fsinus[k]   = (int)Math.floor(60*Math.sin(d));
Fkosinus[k] = (int)Math.floor(60*Math.cos(d));}

//for drawing picture
for( int n = 1 ;n<order+1 ; n++ ){
for( int i = 0 ;i<n ; i++ ){
for( int j = 0 ;j<kListing[i].size() ; j++ ){
for( int k = 0 ;k<kListing[n-i-1].size();k++){
kListing[n].addElement(new Nodes(
 (KListD)kListing[i].elementAt(j),
 (KListD)kListing[n-i-1].elementAt(k) ));}
                }
        }
}
```

Sequence diagrams (Diagram 4) presents actions which are described in activity diagrams, but here the accent is on the order and time duration the life length of some object (instance some class). This sequence diagram only shows the segment method *Draw()* which has responsibility for ensuring the rendering of all possible triangulation. This diagram shows only the first part, ie the possibility of drawing all possible combinations. This uses the changing *x*, *y*, *z* and *w* whose value is obtained by using methods from the class TriangulationPol. These methods are: *Fsinus()*, *Fkosinus()* and *Math.sin(d)*, which added to the position *sCursorX* and *sCursorY*.

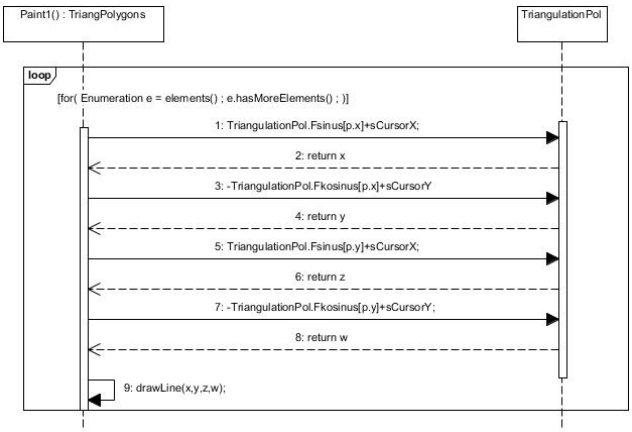


Diagram 4. Sequence for class *TriangPolygons* and *TriangulationPol*

Java source code based on UML diagram sequence:

```
public void Draw(Graphics g, String aLabel){
int x,y,z,w;
if ( XLIMIT<= sCursorX ) {
sCursorX = XEDGE; sCursorY += YBLANK;}
for(Enumeration e = elements();
e.hasMoreElements();){
Point p = (Point) e.nextElement();
//for drawing line in polygon
x=TriangulationPol.Fsinus[p.x]+sCursorX;
y=-TriangulationPol.Fkosinus[p.x]+sCursorY;
z=TriangulationPol.Fsinus[p.y]+sCursorX;
w=-TriangulationPol.Fkosinus[p.y]+sCursorY;
g.setColor( Color.black);
g.drawLine(x,y,z,w);
}
```

Method *Draw* is located in class *TriangulationPol*. This method is responsible for making individual polygon triangulation. In Java with the command *drawLine* for appropriate number of vertices is formed regular convex polygon. One combination of internal diagonals forms a triangulation of the convex polygon. Method *DrawAll* also belongs to the class *TriangulationPol*, and she is responsible to provide iteration with the method *Draw*. The main executive method in Java is located in *App* class and is responsible for the basic input of *n* parameter (number of vertices of a convex polygon).

UML communication diagram (Diagram 5) emphasizes the structure of relationships between participants in the interaction. This diagram shows all who participate in the interaction in the implementation of the application and their interrelations.

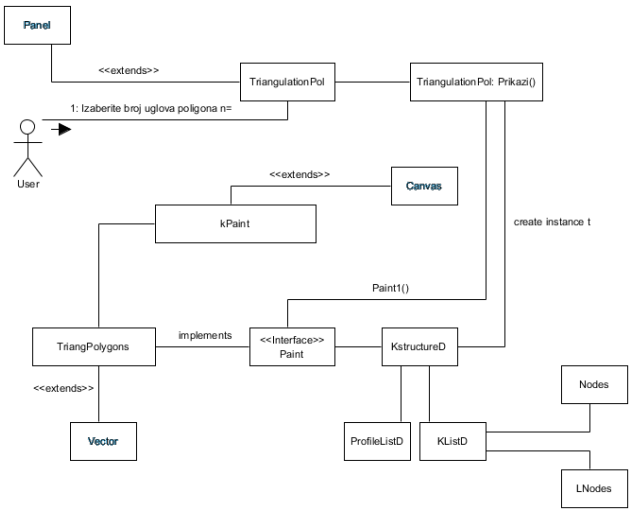


Diagram 5. Diagram of communication

During testing our application which programming in Java, we recorded a total time of execution of applications for corresponding value for ***n***, number of combinations per second. When executing Java application Java Virtual Machine (JVM) reserves more working memory for its object. JVM is compiled for many platforms and it comes in package with the standard Java libraries that together make Java Runtime Environment (JRE), and on any platform where it can be installed can run Java applications. This is a huge advantage, it shortens the time of porting to other platforms and has control in the execution of code and provides greater security. When testing of Java application is observed that

with increasing values of ***n*** increases the number of combinations per second.

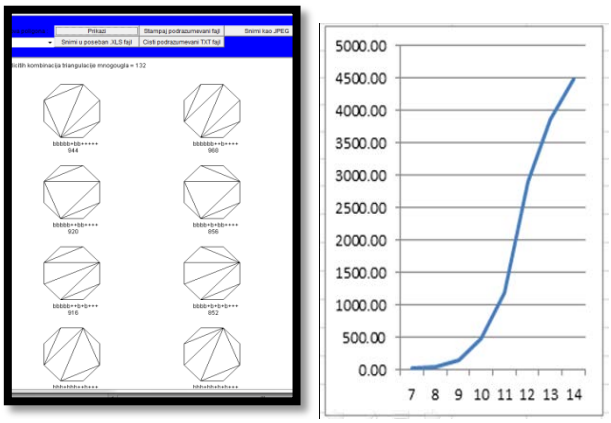


Figure 2. One part of Java aplication (Left) and diagram for result of testing application (Right)

* PC performance for testing results: *CPU - Intel(R)Core(TM)2Duo CPU, T7700, 2.40 GHz, L2 Cache 4 MB (Full-Speed), RAM Memory - 2 Gb, Graphic card - NVIDIA GeForce 8600M GS.*

4. Conclusion

The main task of our paper is to present visual planning for solving some algorithms in computer graphics and computational geometry. In this paper we give a proposal which programming language is most suitable to implement algorithms which are used in computer graphics in terms of speed, simplicity of syntax and clarity of the source code. The basic idea was that through the modeling tool graphically present the algorithm and based on the final UML models we implement in Java environment. The advantage of this way of resolving the problem is reflected in two aspects: aspect of the analysis and design and programming aspects.

From the aspect of the analysis and design it provides a clear definition of requirements and specific plan that we will later use to create applications. From the programming aspect UML describes the source code, models help to visualize the system as it is or what it should be and allow you to determine the structure and behavior of the system. The results of this paper are specific models in the form of UML diagrams and actual implementation in the Java environment. As a conclusion from the application programming and testing can identify that Java is a programming language with exceptional abilities when it comes to working with graphics and also when it comes to speed of execution.

Reference

[1]. C. Icking, R. Klein, P. Köllner, L. Ma, *Java Applets for the Dynamic Visualization of Voronoi Diagrams*, Algorithmica, Volume 22, pp: 477-482, 1998.

[2]. L. Zhen-ping, H. Huai-jian, L. Qiang, Z. Fa-hua1, *Study of the technology of 3D modeling and visualization system based on Python*, Changjiang Water Resources Commission, China, 2008.

[3]. Sabau, G., Costin, M.L., Posdarie, E., *Recognizing Face Features Using Points of Active Contour, PHP–Image Processing and GD Grafic Extension*, Comp.Tech.-ICCTD, pp:336–339, Romania, 2009.

[4]. Suzumura, T., Trent, S., Tatsubori, M., Tozawa, A., Onodera, T., *Performance Comparison of Web Service Engines in PHP, Java and C*, ICWS '08. IEEE International Conference, pp: 385 –392, 2008.

[5]. J. Hugunin, *Python and Java: The Best of BothWorlds*, Corporation for National Research Initiatives, in Proceedings of the 6th International Python Conference, 1997.

[6]. D. Cunningham, E. Subrahmanian, A. Westerberg, *User-Centered Evolutionary Software Development Using Python and Java*, Engineering Design Research Center, Carnegie Mellon University, Pittsburgh, PA, 2010.

[7]. Saračević M., Stanimirović P., Mašović S., *Implementation of some algorithms in computer graphics in Java*, TTEM - Technics Technologies Education Management, *paper accepted for publishing:* Volume 8, No.1, 2013.

[8]. Saračević M., Mašović S., Kamberović H. (2012), *Implementacija nekih algoritama računarske grafike u JAVA NETBEANS okruženju*, XVI Scientific and professional conference – Information Technology: IT2012, University of Montenegro.

[9]. Garey, M.R., Johnson, D.S., Preparata, F.P., Tarjan, R.E., *Triangulating a simple polygon*, Inform. Process. Lett. 7, pp: 175-180.

[10]. Ingmar Peter Stefan Gumhold, *Teaching computer graphics with java 3*, WSI/GRIS, University of Tübingen, Germany, 2010.

[11]. J. Mark Keil, Tzvetalin S. Vassilev, An Algorithm for the MaxMin Area Triangulation of a Convex Polygon, 15th Canadian Conference on Computational Geometry, Nova Scotia, 2003.

[12]. Masovic, S., Saracevic, M., Kamberovic, H., Kudumovic, M.,, *Java technology in the design and implementation of web applications*, TTEM – Technics Technologies Education Management, ISSN 1840-1503, Volume 7, No.2, 2012.

[13]. Laszlo, Michael J., *Computational geometry and computer graphics in C++*, Prentice Hall (Upper Saddle River, N.J.), Book, 1996.

Corresponding author: Camil Sukic.
Institution: Department for Computer Sciences, University of Novi Pazar, Serbia
E-mail: camil.sukic@gmail.com