Analysis and Design of Object-oriented Software using Multidimensional UML

Lukas Gregorovic
Faculty of Informatics and Information Technologies
Institute of Informatics and Software Engineering,
Slovak University of Technology in Bratislava
Ilkovičova 2, SK-84216 Bratislava 4
xgregorovic@stuba.sk

Ivan Polasek
Faculty of Informatics and Information Technologies
Institute of Informatics and Software Engineering,
Slovak University of Technology in Bratislava
Ilkovičova 2, SK-84216 Bratislava 4
ivan.polasek@stuba.sk

ABSTRACT
This paper introduces our method of the UML diagrams visualization in 3D space. It uses the layers for particular components and modules in class diagram, alternative and parallel scenarios in sequence and activity diagrams with modern combined frameworks. The herein presented approach contains also automatic generating of the object diagrams and final class diagram from the sequence diagrams of the use case scenarios. It applies force directed algorithm to create more convenient automated class diagram layout using semantics by adding weight factor in force calculation process.

Keywords
3D UML; Analysis and Design; Sequence Diagram; Class Diagram; Fruchterman-Reingold

1. INTRODUCTION AND RELATED WORK
Unified Modeling Language (UML) is standardized [3] and widely used for the creation of the software models describing architecture and functionality of the created system. Moving UML diagrams from standard 2D to 3D space we allow visualization of the large diagrams in modern three-dimensional graphics to achieve more readable schemas of complex models to decompose behavior and functionality to particular scenarios of the system, alternative and parallel flows.

There are some existing alternatives how to visualize UML diagrams in 3D space. One of the first prototype VOGUE [14][15] used 3D space for the process of visualization. M. Gogolla et al. concern on 3D class and sequence diagrams [12][13].

Interactions of the classes in the shape of sphere was described in 2001 by T. Dwyer[10]. S. Kent and J. Gil suggested notation for 3D dynamic diagrams [11] and hierarchical embedded parts similar to composite fragments in the modern UML sequence diagrams. In our institute, we have also started with the 3D visualization and animation of the messages in the sequence diagrams [16][18]. In contrast, Paul McIntosh worked on new State Machine Diagram [8]. This implementation was compatible with the X3D ISO standards used for representing 3D computer graphics and allows diagrams created by this implementation to be shared across X3D viewers. This X3D-UML approach [2] (combination of X3D standard and UML diagrams) displays the state diagrams in movable hierarchical layers in which also filtering can be applied.

GEF3D [4] is a 3D framework based on Eclipse GEF (Graphical editing framework) developed as Eclipse plugin. Using this framework (or multi-editor), existing GEF-based 2D editors can be easily embedded into 3D editors for visualization connections between two-dimensional diagrams in the planes.

Another concept in the field of 3D UML visualization uses virtual boxes [7] where diagrams are placed onto sides of the box allowing them to arrange inter-model connections, which is easily understandable. GEF3D is in the Eclipse incubator and does not allow users to make modifications in displayed models and there is no official release of GEF3D available.

UML diagrams can be very complex and difficult to remember and geon diagrams [6] use different geometric primitives that are easier to memorize for human readers and it can be analyzed more rapidly.

2. ANALYSIS AND DESIGN WITH 3D UML
Our method proposes the process of analysis and design in 3D space framework. It does not support only 3D visualization as many approaches mentioned in *related work* section above, but also allows to create and update 3D UML diagrams in separate layers arranged in 3D space (see Figure 1 and 4).

It is not just the method for reducing complexity and comparison of modules and scenarios, but also for better understanding the context and interconnections of the processes and modules as well as the new way of inception and consistency maintenance of diagrams. Currently, it offers creation of the use case scenarios in the UML Sequence Diagrams. These diagrams could be automatically transformed in to the object diagrams and then to the class diagram to visualize required structure. We are developing tools and environment also for UML Activity Diagram editor as an alternative method for scenarios and system functionality. Our editors support the latest UML standard: combined fragments in the sequence and activity diagrams (Figure 2 and 3).

Our first prototype is a standalone system implemented in C++ language using Open Source 3D Graphics Engine (OGRE) and we are preparing to alternate it in the future with the OpenSceneGraph and Open GL as an open source high performance 3D graphics.

2.1 Required structure discovering and visualization

Analysis and design of the large software system can start with use cases and their scenarios describing and capturing system behavior in the UML sequence diagrams. In sequence diagrams we can identify essential objects and their methods that are necessary for functionality of the system. UML sequence diagrams are created by analysts in separate layers all at once in 3D space, but the object diagrams (again in separate layers) and final class diagram with real association could by automatically transformed. Thanks to element similarities between sequence diagram and object diagram in the UML metamodel definition [3], it is possible to use same shared data representation and rendered objects and their interconnections from lifelines and messages in the sequence diagram.

Identical elements in object diagrams have fixed positions for easy visual comparison and projection to the automatically created class diagrams with classes derived from these objects. Their relationships (associations) are inferred from the interactions in the sequence

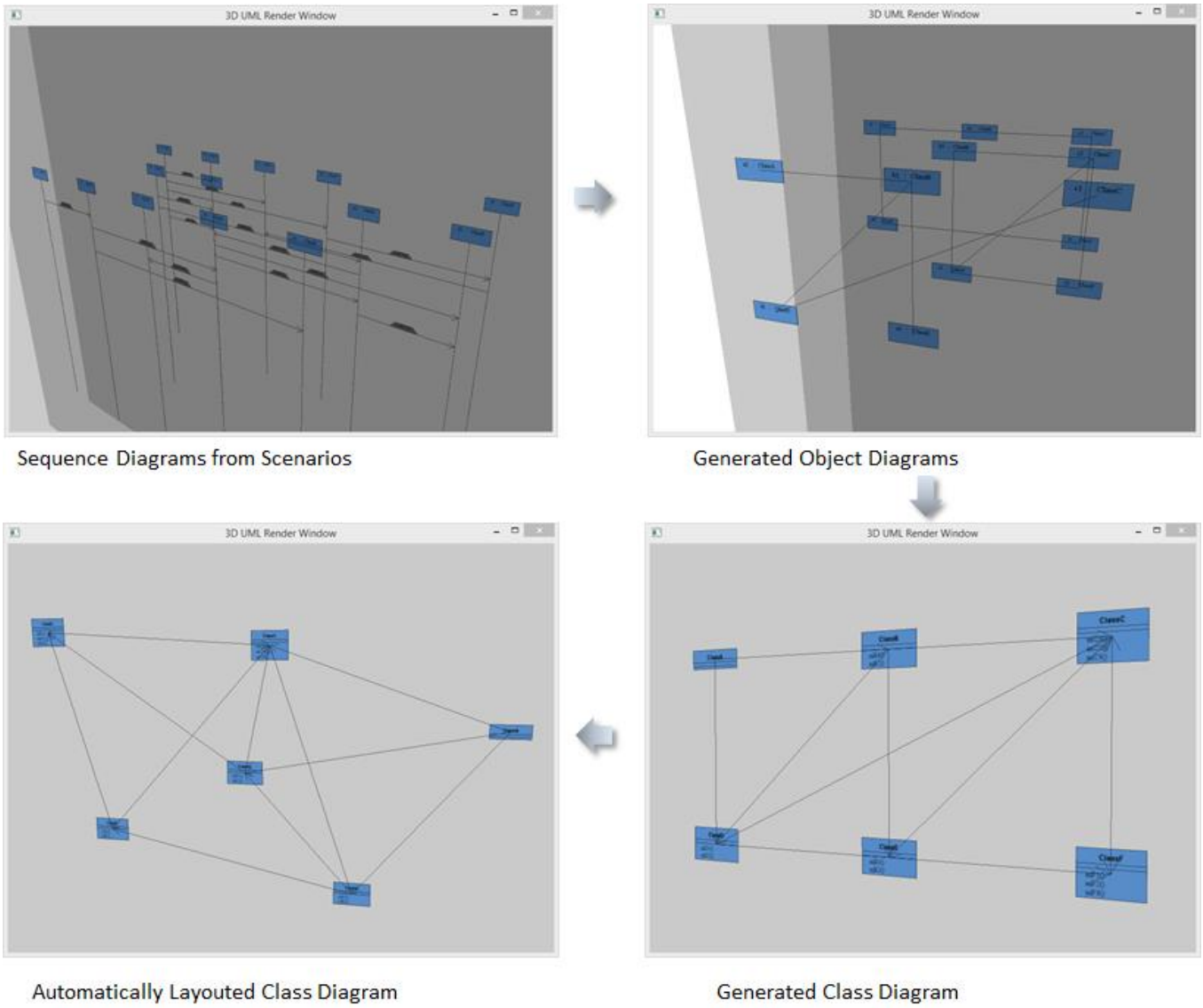

Figure 1. Generation and layouting of the class diagram

diagrams and class methods are extracted from the required operation in the interactions of these sequence diagrams. Transformation is visible in Figure 1.

In early phases of prototype we used simple layout algorithm where each unique element was placed on next available cell in imaginary grid. In the next step we used layout creation with force-directed algorithms described later in this paper. Primarily we have used F-R to change or rearrange the position of the elements for better visualization and minimization of the relationship intersections in generated class diagrams. Generation of the class diagram creates only associations, but we have tried in the next step of semantic layouting to reflect

the type of the relationships (association, aggregation, generalization) and the name of the elements in the common process of the creation or the upgrade. Part of this (especially the semantic of the names) could be implemented also backward to the layouting in the process of the generation.

Algorithm for class diagram creation was optimised with buffering and memorisation to create unique class types from these object collections and their methods from the message calls.

2.2 Semantic Layouting

The layout of generated class diagram was not good enough for reading an understanding. Users are looking for groups, collections, hierarchy, software patterns [17], relationships, semantics and other hidden aspects of the model. Assuming that the most important relationship between two elements in the class diagram from semantic view is the generalization, then the aggregation, and finally the association, it is possible to modify output of layout algorithm by adding the weight factor in attractive force calculation process.

We have tested two force-directed algorithms [20]: FM3 and Fruchterman-Reingold (F-R) algorithm. In the case of F-R, we can introduce the weight to the vertices or the edges and modify the original behaviour of the algorithm [1].

Time complexity of the algorithm is not evident according to the scale in which we use these algorithms: in class diagram with size of 10-100 classes is the layout calculation very fast. Modification of the algorithm is simple and implementation using FM3 can be realized in the future, if it will be necessary.

Our approach distinguishes different types of relationships between classes in the process of the calculating attractive forces and multiplies it by weight of corresponding type of the edge.

While prototyping phase, weights of relationship edges were experimentally set as follows [20]: generalization to 200, aggregation to 100, and association to 10. It allows to make semantics patterns of class diagram more visible.

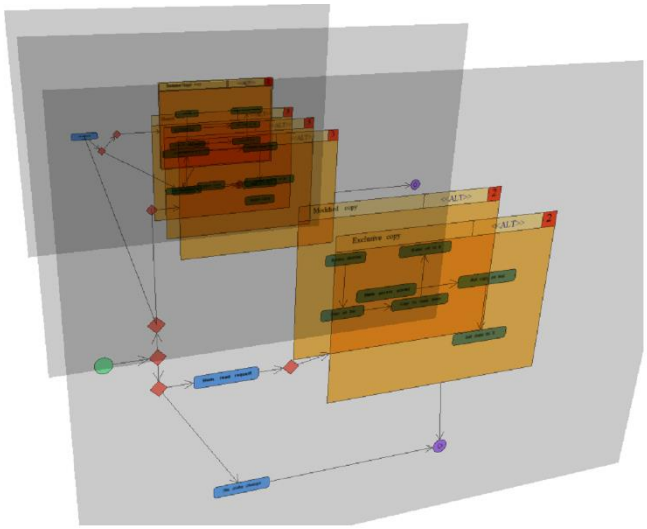

Figure 3. Activity Diagram with Combined fragments

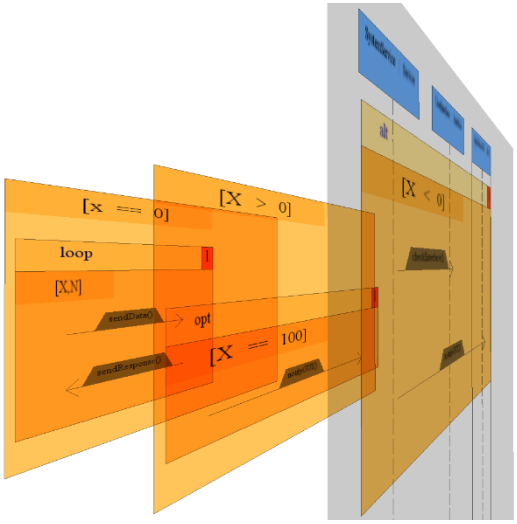

Figure 2. Combined fragments in Sequence diagram

Weighted F-R algorithm create more convenient layout. Elements are attracted to each other with proportion to relationship, which is apparent between each pair of elements. General strategy element is close to concrete strategy elements. Concrete strategy objects are closely related to their parent object, so it is crucial to group them near each other. Context element that contains strategy is the second nearest, as it is the second important in this semantics. On the other hand, client element that interacts with context object is most distant as it is the least significant in this semantics.

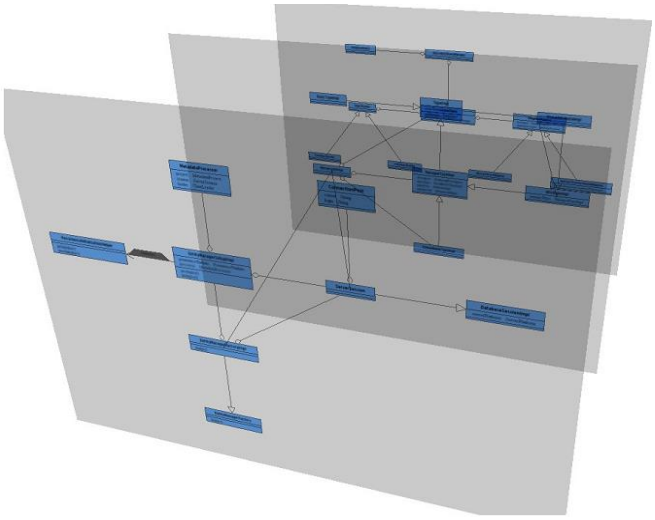

Figure 4. Class diagrams in separate layers [5]

3. CONCLUSION AND FUTURE WORK

We are implementing 3D visualization of the sequence, activity and class diagrams (see Figure 1, 3, and 4) in the first running prototypes. It is possible to create and update these diagrams (insert, update, move and delete the elements).

We are working on serialization diagram content to the XMI files. Our prototype can generate the class diagram from the object diagrams using our authentic idea [9]. These object diagrams are

automatically derived from the lifelines and message interconnections in the sequence diagrams (see the section 2.1 of this paper).

Inserted semantics by adding weight factor in force calculation process of the layout algorithm achieves significantly better layout. Type of relationship between vertices influence weight applied on the attractive forces. The method creates more readable organization of elements grouped by semantics: it puts relevant classes more together and creates smaller chunks of classes instead of one big mass as it was in the previous generated class diagram. This process supports better understanding and modifying of designed diagrams. The algorithm creates unwanted edge crossings sometimes, but it is an initial layout, which could be corrected by the user.

We are working on the algorithm enhancement to create layout reflecting not only relationships between elements, but also the other semantic attributes: element names, object types (GUI, Business and DB services), and software design patterns. This framework as a CASE system will support also collaboration of analysts and developers and monitoring their activities [19] for better understanding of their work.

4. ACKNOWLEDGMENTS

This work was supported by the Scientific Grant Agency of Slovak Republic (VEGA) under the grant No. VG 1/1221/12. This contribution is also a partial result of the Research & Development Operational Programme for the project Research of Methods for Acquisition, Analysis and Personalized Conveying of Information and Knowledge, ITMS 26240220039, co-funded by the ERDF.

5. REFERENCES

[1] T. M. Fruchterman T. M. and Reingold E. M. Graph drawing by force-directed placement. Software: Practice and experience, 21(11):1129–1164, 1991.

[2] McIntosh P., Hamilton M., and van Schyndel R. X3d-uml: Enabling advanced uml visualisation through x3d. In Proceedings of the Tenth International Conference on 3D Web Technology, Web3D '05, pages 135–142, New York, NY, USA, 2005. ACM.

[3] OMG Unified Modeling Language (OMG UML), Infrastructure, Version 2.4.1, [Online], Available: http://www.omg.org/spec/UML/2.4.1/, Accessed: March 1, 2014

[4] Pilgrim J. and Duske K. Gef3d: A framework for two-, two-and-a-half-, and three-dimensional graphical editors. In Proceedings of the 4th ACM Symposium on Software Visualization, SoftVis '08, pages 95–104, New York, NY, USA, 2008. ACM.

[5] Škoda M. Three-dimensional visualization of uml diagrams. Diploma project, Slovak University of Technology Bratislava, Faculty of informatics and information technologies, May 2014.

[6] Casey, K. and Exton, Ch. A Java 3D implementation of a geon based visualisation tool for UML. Proceedings of the 2nd international conference on Principles and practice of programming in Java, June 16-18, (2003), Kilkenny City, Ireland.

[7] Duske, K. A Graphical Editor for the GMF Mapping Model. (2010). http://gef3d.blogspot.sk/2010/01/graphical-editor-for-gmf-mapping-model.html

[8] McIntosh, P. X3D-UML: User-Centred Design. Implementation and Evaluation of 3D UML Using X3D. PhD. Thesis, RMIT University, (2009).

[9] Polášek, I. 3D Model for Object Structure Design (In Slovak). In: Systémová integrace. - ISSN 1210-9479. - Vol. 11, No. 2 (2004), pp. 82-89

[10] Dwyer T. Three dimensional uml using force directed layout. In APVis '01 Proceedings of the 2001 Asia-Pacific symposium on Information visualisation - Volume 9, APVis Proceedings, pages 77–85. Australian Computer Society, Inc., 2001.

[11] Gil J. and Kent S. Three dimensional software modelling. In ICSE '98 Proceedings of the 20th international conference on Software engineering, ICSE Proceedings, 1998.

[12] Gogolla M. and Radfelder O. On better understanding uml diagrams through interactive three-dimensional visualization and animation. In AVI'00 Proceedings of the working conference on Advanced visual interfaces,AVI Proceedings, 2000.

[13] Gogolla M., Oliver Radfelder O., and Richters M. Towards three-dimensional representation and animation of uml diagrams. In UML'99 Proceedings of the 2nd international conference on The unified modeling language: beyond the standard, UML Proceedings, 1999.

[14] Hideki Koike. The role of another spatial dimension in software visualization. ACM Transactions on Information Systems (TOIS), 11, 1993.

[15] Hideki Koike and Hui-Chu Chu. How does 3-d visualization work in software engineering? In ICSE '98 Proceedings of the 20th international conference on Software engineering, ICSE Proceedings, 1998.

[16] Šperka M., Kapec P., and Ruttkay-Nedecký I. Exploring and Understanding Software Behaviour Using Interactive 3D Visualization. In: ICETA 2010: 8th International Conference on Emerging eLearning Technologies and Applications, pp. 281–287, 2010.

[17] Gamma, E., Helm, R., Johnson, R., and Vlissides J. Design Patterns, Elements of Reusable Object-Oriented Software, Addison-Wesley professional computing series, 1995.

[18] Grznár F. and Kapec P. Visualizing dynamics of object oriented programs with time context. In Proceedings of the 29th Spring Conference on Computer Graphics, SCCG '13, pages 065:65–065:72, New York, NY, USA, 2013. ACM.

[19] Bieliková M., Polášek I., Barla M., Kuric E., Rástočný K., Tvarožek J., Lacko P. Platform Independent Software Development Monitoring: Design of an Architecture. Theory and Practice of Computer Science - SOFSEM 2014, Lecture Notes in Computer Science Volume 8327, Springer, pp 126-137.

[20] Gregorovic L., Polasek I., and Sobota B. Software model creation with multidimensional UML. International Conference on Research and Practical Issues of Enterprise Information Systems, The 23rd IFIP World Computer Congress, Daejeon, Korea, 2015. (accepted)