Development of autonomous software solutions based on geometric cores to create digital assembly models of machine tools

A N Yurov\textsuperscript{1}, M I Chizhov\textsuperscript{1}, V V Vetokhin\textsuperscript{1} and V V Lavlinskiy\textsuperscript{2}

\textsuperscript{1}Voronezh State Technical University, 14, Moscow ave., Voronezh, 394026, Russian Federation
\textsuperscript{2}Voronezh State University of Forestry and Technologies named after G.F. Morozov, 8 Timiryazeva str., Voronezh, 394000, Russian Federation

E-mail: yurovalex@mail.ru

Abstract. The article considers approaches and solutions aimed at improving the productivity of engineering departments, where CAD software solutions are actively used. In the manufacture of parts of machinery there is a need in the design of tooling. Elements of machine tools can be prepared and distributed in electronic libraries, which in an automated mode will create a set of parametric models of the required tooling of machine equipment. The accuracy and quality of modelling will be achieved by eliminating the human factor from the design process. The development of such software solutions is implemented by means and components of the open geometric kernel Open Cascade, which has great potential in the field of modelling and scientific calculations.

1. Introduction

Solutions in the field of automation of production processes allow to accelerate the production of new products, to optimize the work of engineering divisions. Preparation of application libraries of digital elements significantly relieves the designer from performing routine operations in the process of work on the creation of new products. Models the use of standard elements, typical blocks of tooling and tooling can be avoided if you purchase as part of CAD solutions (NX, Solid Edge, Catia, Creo, T-Flex) additional modules in which these elements are present. Or to pay for the development of vendors. However, the developed modules implement only the current projects of the enterprise and do not take into account the specifics of production as a whole. However, the use of software APIs requires high qualification of the employee both in the subject area of production and professional knowledge in the field of SOFTWARE development. In addition, some CAD solutions can be, due to various factors in the enterprise, replaced by others-import substitution processes (for example, in the case of support of the domestic manufacturer, the requirement to use 3D Compass instead of Solid Edge). In this case, it is necessary to deal with the API of the new design system.

As noted earlier, you can develop an application library through the API interface and then connect it to the CAD solution. However, in case of transition from the existing CAD to the new one, all the achievements are lost. Therefore, when automating processes, it is easier to have a database with models inside the system and use parameterization when creating subassemblies.
You can also go the other way and develop a universal application solution that will conduct modelling, providing a high level of automation, while not depending on the basic CAD at all. Such modules can be created by using the functionality of geometric cores, which allow you to obtain models of geometric objects not only typical elements, but also complex in its topology with the ability to transfer the results in the process to other CAD systems for later use. Open geometric Open Cascade can be used as an application library development tool.

The Open Cascade software package has extensive functionality for the development and support of projects in any engineering field. Design of application libraries and stand-alone solutions of Open Cascade allows to realize production tasks without resorting to significant financial investments. The software set of the geometric kernel contains not only classes and methods of modeling bodies and objects, but also solutions for analyzing the geometry of these components, support for various export data formats. The achieved results can be transferred to third-party CAD systems for further use in the design of engineering products.

2. Geometric core capabilities of Open Cascade

Before proceeding to the approaches and recommendations for creating stand-alone programs based on the geometric core, it is necessary to list the functional elements of Open Cascade and a set of classes for application development. This open geometric core is used not only for modelling, but also for the development of applications for engineering analysis, the preparation of solutions for CNC machines, as well as the implementation of other tasks.

The open Cascade geometric package provides tools for working with basic geometric objects, topology, algorithms for the analysis of models and their visualization. The topological model of the geometric kernel is based on the representation of boundary bodies, solid objects are represented by some multiple structure or can be performed by one constructive primitive. The geometric core allows you to create and work with any known model: frame, surface, solid, as well as all listed in one application.

In a model, you can create topological features that combine zero-geometry features (points), segments and lines (curves), surfaces, and solids, which can also contain embedded points, curves, and surfaces.

Geometric algorithms are used to construct and modify geometric objects and solve geometric problems. The geometric core of Open Cascade has a significant set of geo-metric methods for modelling and calculations. The open Cascade geometric package includes basic methods related to the simplest algorithms for creating lines, constructing parallel lines, and constructing lines at a given distance from each other. When filling a certain array, operations are available to check intersections, generate, transform, and trim objects on the plane. More complex methods for constructing Bezier and NURBS curves are implemented. With the help of these algorithms it is possible to solve the following common geometric problems:

- construction of geometric primitives (in the form of a set of points, lines, circles, ellipses) with the use of restrictions;
- realize intersections and projections;
- prepare the creation of curves and surfaces for approximation and interpolation;
- to create surfaces by sweeping the area on a curved path;
- create a construction of ruled surfaces;
- to prepare the filling surfaces.

The geometric core of Open Cascade contains its own mathematical library, which includes some well-known methods of calculation for numerical integration, solving systems of linear and nonlinear equations, optimization.

The Open Cascade package provides a wide range of topological algorithms that allow you to create and modify topological objects using various construction methods: low-level algorithms for directly constructing any topological objects from lower-level objects:

- creation of primitives (blocks, prisms, cones, spheres)
• a shift of the contours along complex profiles;
• Boolean operation;
• local Boolean operations (such as gluing solids along common faces);
• modification of the original object by functional features: mixing, different types of protrusions and depressions.

The geometric core of Open Cascade is focused on modeling the bodies of machine parts, which include chamfers, grooves, thin offset circles around or inside solids or passing through the entire model.

Open Cascade visualization algorithms provide a complete set of operations for visualization of geometric and topological objects, as well as grids in different modes: wireframe and solid, shaded and hidden lines with full control of various visualization attributes.

3. Goal setting and list of tasks
When designing a stand-alone application for modelling the device Assembly, it is necessary to prepare a software solution interface, in which the construction and visualization of the entire Assembly by the Open Cascade geometric core functionality is performed by selecting a given type of execution.

The list of tasks, the solution of which requires implementation in the creation of applied Autonomous solutions:

• preparation of data representation in the form of tables with standard sizes of all components, or entering data into a compact embedded DBMS (h-R, SQLite);
• preparation of structures that will provide data with dimensions for the design of Assembly components;
• execution of parametric models of Assembly elements;
• preparation of services that allow the use of export data formats;
• preparing the wood for Assembly, including interference with the part of the Assembly and space constraints;
• visualization of parametric Assembly models in a separate application window;
• preparation of the program interface, which provides access through dialog boxes to the selection of the required execution of the Assembly with subsequent modelling and unloading of the results of the project in an export format.

4. Scheme of the project solution
When designing the application interface, you should separate the dialog boxes from the basic template classes, and in the case of preparing tables with sizes for Assembly, you develop a separate dialog console. Visualization of all built objects is assumed in a separate application window.

After the Assembly is completed, additional functionality appears in the application. First of all, these are tools that allow you to change the color of the models that are part of the Assembly. Next, it should be possible to unload the Assembly through the export format to the media. A list of components must be associated with the Assembly and displayed in a separate window. Selecting items from the list will highlight the Assembly item in the main window.

According to the proposed recommendations, the following diagram of a stand-alone solution for modeling the device Assembly can be used (figure 1).

5. Scheme of the project solution
To develop a software tool, it is proposed to use a multi-file layout of the project, in which each file will be a separate class with a description and implementation of the application functionality.

Description of elements of functional parts of the automotive solution:

Wrapper class. Its tasks include the creation of all actions from the controls. Controls, in turn, can be placed in menus and toolbars. In this case, duplication of actions is allowed, since the user can refer
to those or objects of the interface according to personal preferences. The interface class also includes dialogs that provide easy access to data, as well as file I/o operations.

**Figure 1.** Application development diagram.

A class that provides storage of standard sizes for building an Assembly. The tasks of the class are to provide the acquisition and exchange of data that is needed to create a software Assembly model.

A common interface class that allows to coordinate all actions taking place in the application.

Class that provides work with the current collection. Its tasks include access to methods of modelling Assembly elements. Access to the goto-Vym elements of the Assembly, uploading data to an external storage medium.

Modelling class, whose methods are responsible for creating parametric models by means of Open Cascade.

Representation class, whose tasks include operations for modification and transformation of objects of geometric modelling.

All these classes must interact with each other through interface methods, provide and send data for modelling, and interact with the elements in the Assembly when the application has completed the stage of program modelling.

Based on the above, the implementation of the design solution according to the functional scheme, which is presented in figure 2, is being prepared.

Modelling of the Assembly with subsequent visualization of the Assembly is performed by the geometric kernel Open Cascade. Therefore, it is necessary to connect the functionality of the geometric modelling package to the software solution, including the necessary header files and static libraries. The use of parametric models will require the tabular data representation which can be displayed in the dialog boxes. It is more expedient to prepare dialog boxes by means of the interactive development designer. In case of preparation of non-standard interfaces any declarative language and the editor of forms for it can be used. On the basis of the template for creating a window application, an interface is formed, which provides a text menu, toolbars. Visualization is based on the use of OpenGL components, so it requires making dependencies in the design solution.
6. Implementation of a stand-alone module

Interface elements of the application are implemented by some set of classes that it is advisable to inherit from the base to create Windows. When using the Qt production development environment, the Windows application framework is used as a template, which Qt defines as QMainWindows. Next, you create a class group with the following functionality for each class:

- **class MainWindow**: public QMainWindow - a class that inherits from the main template class is required to build the design solution interface;
- **class Dialog**: public QDialog-inherited class from dialog. In the specified class, the implementation of the tabular data that is needed to build the components of the Assembly and the Assembly as a whole will be prepared;
- **class document**: public QObject-a class that provides capabilities for loading models and unloading data from the system, contains functionality for calling services for painting models, creating an Assembly and storing models in a dynamic container.
- **class models**: public QObject - the capabilities of the specified class allow you to model the components of an Assembly.
- **class VisualScreenContext**: public QGLWidget-class that allows you to display models on the application scene, manipulate objects, handle events from the manipulator "mouse" and so on.

Data export for transfer to other CAD systems can be done in the format. step. Code implementation is built on the basis of the geometric kernel API, allows you to create a list containing a set of interactive objects, fill it with objects already available in the context of the application window output, convert objects to a data format that can be transferred to an export format.
Other services for standalone solutions such as working with object color, indicator of completion of operations in the construction of Assembly elements, etc. can be implemented in a similar way.

The following are examples of developing stand-alone applications by means of the geometric kernel according to the previously proposed methods.

The step Cam application was implemented as a stand-alone solution that does not require the installation of additional components. The vertical dashboard in the application allows you to control objects on the scene: to rotate the Assembly, move it, get a frame or solid-state views. The top toolbar is responsible for selecting parameters for modeling the Assembly, loading ready-made models in export format, recording design results, removing the Assembly from the scene. The last two panel buttons allow you to override the Assembly color.

The modeling process begins with the selection of parameters in a separate dialog box. After selection, the system transfers the selected sizes to a dynamic container, clears the scene for modeling objects, clears the list responsible for the composition of the Assembly. Upon completion of all operations, geometric modeling of bodies is performed by means of the geometric kernel Open Cascade. The simulation results are shown in figure 3.

![Figure 3. Build an Assembly.](image1)

The application for the creation of the fixing device by means of the geometric core was developed as a separate module that allows you to simulate the Assembly of the clamping device using Open Cascade modeling methods. One of the toolbar buttons brings up a dialog box where the user is asked to select a given build execution.

![Figure 4. Dialog Building a fixture in a wireframe view.](image2)
After selecting the size, the process associated with modeling begins. The modeling process is accompanied by an indication of the design procedures. After building, the Assembly is displayed on the screen and displays a list of elements with names that are part of the Assembly (figure 4).

In each of the implemented projects there is a build tree, interface configuration and parameterization of application Windows for easy scaling of data output on different display devices.

7. Conclusion

Open Cascade geometric core and production development environments with functionality for the implementation of graphical interfaces allow you to create stand-alone solutions for modeling subassemblies and assemblies. Such applications are characterized by significant speed, flexibility in operation, as well as more cost-effective than developed to order from large manufacturers of CAD solutions. Given as examples, electronic libraries of machine tools create and visualize the elements of subassemblies, perform animation of work, which gives an idea of how the device functions in practice.

Complete software solutions allow to quickly and correctly build, depending on the specified sizes, devices, as well as to export the results of the work to other CAD systems for further use. Thus, the geometric core functionality can be used to develop application libraries of any complexity level, to create complex Autonomous solutions for solving production problems.

References

[1] Ryzhkov V A, Sabinina O V and Pak A A 2016 Development module creating typed elements for an offline client to Teamcenter Informatics: problems, methodology, technology: int. scientific-methodical conf. (Voronezh: Voronezh State University Press) 404-8
[2] Yurov A N and Eremin I A 2017 Software modeling and visualization of geometric bodies by means of the OCCT core Informatics: problems, methodology, technology: int. scientific-methodical conf. Voronezh: Voronezh State University Press) 340-5
[3] Yurov A N 2017 Development of Autonomous solutions based on functional capabilities of OCCT New technologies in research, design, management, production: proceedings of the Int. scientific and tech. conf. (Voronezh) 189-91
[4] Slyadnev S, Malyshev A and Turlapov V 2017 CAD model inspection utility and prototyping framework based on OpenCascade Proc. GraphiCon (Perm) 323-7
[5] Yuan G and Zhang Y 2008 Development of 3D modeling platform based on Open CASCADE Journal of Engineering Graphics 4 146-9
[6] Tsygankov D and Pokhilko A 2015 The Design Process Procedural Representation Based on Open CASCADE Technology Interactive Systems: Problems of Human-Computer Interaction (Ulyanovsk ) 232-5
[7] Yuan Y, Wang Y H, Jiang L, Lin C D and Wang Y 2013 Research on modeling technology based on Qt and OpenCASCADE Modern Electronics Technique 10
[8] Zhou Q et al. 2014 Research and Implementation of CAD-INP Integration System Based on OpenCASCADE Multimedia and Ubiquitous Engineering (Berlin: Heidelberg) 227-32
[9] OpenCascade - an open source library for BRep solid modeling https://www.opencascade.com/
[10] Cai D H and Tu L Q 2010 Speedy Interference Test Based on Open CASCADE Software Light Industry Machinery 2
[11] Pan G Y Han J and Feng H T 2015 Research on Parametric Modeling of Ball Screw Based on Open CASCADE Machine Building & Automation 3