UDC: 004.415:681

В.В. Євсєєв
Харківський національний університет радіоелектроніки

ГРАФІЧНЕ ПОДАННЯ КОНЦЕПТУАЛЬНОЇ СХЕМИ І ОСНОВНИХ БАЗОВИХ ПОНЯТЬ ОПИСУ ВІЗУАЛЬНИХ ЕЛЕМЕНТІВ ПРОГРАМНИХ ПРОДУКТІВ І МОДУЛІВ В РІШЕННІ ЗАДАЧ АВТОМАТИЗАЦІЇ ПРОЕКТУВАННЯ КІС ТПВ

У даній статті пропонується новий підхід до графічного подання структури опису програмних продуктів і модулів для КІС ТПВ в рішенні задачі автоматизації проектування. На базі розробленого математичного опису властивостей і якостей елементів інтерфейсу користувача, а також їх взаємодії для мов високого рівня програмування була вдосконалена методологія структурних карт Константайна (СКК), фундаментальні елементи якого відповідають міжнародним стандартам IBM, ISO і ANSI. Розроблене графічне подання дозволить спростити складність опису параметрів візуалізації інтерфейсу користувача, функціонала програмного продукту, що проектується, задати всі необхідні вимоги,що пред'являються замовником, прив'язати до графічних елементів необхідні властивості та події з використанням «лінгвістичних змінних», які посилаються на певний «контейнер рішень» з уже реалізованими фрагментами програмного коду. Запропонована концепція дозволить скоротити час розробки програмного продукту або модуля для КІС ТПВ за рахунок зменшення ймовірності внесення змін на етапах тестування і впровадження, а також дозволить розрахувати вартість і час розробки на стадії складання ТЗ.

Ключові слова: *карти Константайна, автомаизація, КІС ТПВ.*

В.В. Евсеев
Харьковский национальный университет радиоэлектроники

ГРАФИЧЕСКОЕ ПРЕДСТАВЛЕНИЕ КОНЦЕПТУАЛЬНОЙ СХЕМЫ И ОСНОВНЫХ БАЗОВЫХ ПОНЯТИЙ ОПИСАНИЯ ВИЗУАЛЬНЫХ ЭЛЕМЕНТОВ ПРОГРАММНЫХ ПРОДУКТОВ И МОДУЛЕЙ В РЕШЕНИИ ЗАДАЧИ АВТОМАТИЗАЦИЙ ПРОЕКТИРОВАНИЯ КИС ТПП

В данной статье предлагается новый подход к графическому представлению структуры описания программных продуктов и модулей для КИС ТПП для решения задачи автоматизации проектирования. На базе разработанного математического описания параметров и свойств элементов интерфейса пользователя, а также их взаимодействия для языков высокого уровня программирования была усовершенствована методология структурных карт Константайна (СКК), фундаментальные элементы, которой соответствуют международным стандартам IBM, ISO и ANSI. Разработанное графическое представление позволит упростить сложность описания параметров визуализации интерфейса пользователя, функционнала проектируеммого программного продукта, задать все необходимые требования, предъявляемые заказчиком, привязать к графическим элементам необходимые свойства и события с использованием «лингвистических переменных», которые ссылаются на определенный «контейнер решений» с уже реализованными фрагментами программного кода. Предлагаемая концепция позволит сократить время разработки программного продукта или модуля для КИС ТПП за счет уменьшения вероятности внесения изменений на этапах тестирования и внедрения, а также позволит рассчитать стоимость и время разработки на стадии составления ТЗ.

Ключевые слова: *карты Константайна, автоматизация , КИС ТПП.*

V. Yevsieiev
Kharkiv National University of RadioElectronics

CONCEPTUAL SCHEME AND BASIC CONCEPTS GRAPHIC REPRESENTATION OF SOFTWARE AND MODULES VISUAL ELEMENTS DESCRIPTION IN CIS TPP DESIGN AUTOMATION PROBLEM SOLUTION

In this article, we propose a new approach to the graphical representation of software and modules description structure for CIS TPP for design automation problem solution. On the basis of the developed user interface elements parameters and properties mathematical description, as well as their interaction for high-level programming languages, the methodology of the Constantine's structural maps (CSM) was improved, the fundamental elements that correspond to the international standards of IBM, ISO and ANSI. The developed graphical representation will simplify the complexity of describing user interface visualization parameters, projected software function, set all the necessary customer's requirements, link the necessary elements and events to the graphic elements using "linguistic variables" that refer to a certain "solution container" with already implemented program code fragments. The proposed concept will allow to reduce the time for software or module development for CCI TPP by reducing the possibility of introducing changes at the testing and implementation stages, and also will allow calculating the cost and development time at TT compilation stage.

Keywords: *Constantine's structural maps, automation, CIS TPP.*

Problem formulation. Development and improvement of computer-information systems of technological preparation of production (CIS TPP) for the specificity of each enterprise is a complex scientific and technical problem. At this time, much attention is paid to CIS TPP developing process,

during which it is necessary not only to develop a system, but also to determine at an early stage its cost, risk level and constructing complexity. Also at technical task (TT) preparation early stage, it is necessary to organize development correctly for software and software modules (SM). As a consequence, a complex scientific task arises to develop new methodologies for software and modules for CIS TPP design automation. Currently visual programming methods are widely used, but these methods do not solve the issues of calculating the cost at TT compilation stage and is determined experimentally by each developer, and therefore has a large error and statistically exceeds the declared minimum by 25% or more. Proceeding from the above, there actual task arises for design automation system development, that will enable software prototype development with the functional implementation and its structure it will allow to take into account all the customer's wishes and production specifics and TT compilation stages and to minimize the error in the software and SM estimated cost for CIS TPP.

Last Reseaches and Publications Analysis. Software and SM graphical representation existing methodologies and methods studies showed that to solve posed task following methods can be used:

- Heine-Sarson Structural Analysis (order of construction procedural oreiented, using frequency 20,2 %)[1];
- Yodana De Marco Structural Analysis and Design (order of construction procedural oreiented, using frequency 36,5 %)[2];
- Jackson Systems (building order oriented on data, using frequency 7,7 %)[3];
- Varnier-Orré Structural Systems (building order oriented on data, using frequency 5,8 %)[4];
- Martin Information Modeling (information-oriented building order, using frequency 22,1 %)[5];
- Constantine Methodology, signal-code structure (SCS) information (procedural-oriented building order of construction, using frequency 10,6 %)[6].

Based on the analysis, it was suggested to improve the method of Constantine, which makes it possible to simplify the complexity of the software or SM description for CIS TPP graphical representation.

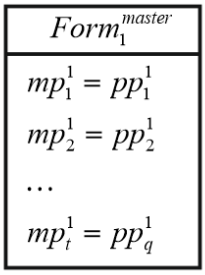
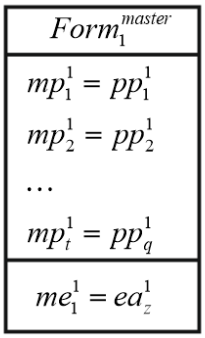
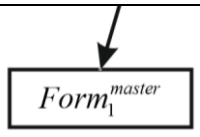
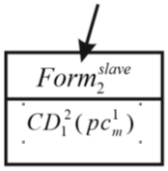
Tasks Set. This study main objective is to improve graphical representation method on the basis of Constantine's structural maps (CSM) methodologies for software and modules design automation for CIS TPP that will simplify main parameters describing process, properties and interrelationship between visual elements and will enable to graphically develop the structure of the developed software using "linguistic variables" and "solution containers" at TT compilation early stage.

Software and Software Modules for CIS TPP Basic Visual Elements Graphical Representation Development. Based on the fact that the software developed for CIS TPP in most cases is a multi-window solution, since problems solution specificity requires working with large amounts of data, hence for convenience in modern CAD / CAM / CAE / PLM systems need their grouping according to common logical characteristics, in accordance with this, a multi-window interface is used, which requires the development of a mathematical method for relationships between forms description. Based on the proposed mathematical descriptions of the main entities, rules and relationships [7,8], it is necessary to develop a graphical representation that will fully allow to display all the necessary information about the parameters and events that belong to each window, as well as the conditions for occurrence of events when the user interacts with one or another component of the interface. Let us denote by P developing software and SM for CIS TPP, that contains graphical forms $(Form)$, in which user interface visual representation is implemented, and is structured according to the necessary logical principles, depending on the requirements for the presentation of information to the user. Let us denote by $Form_1^{master}$ main window existence for developing software or SM, and by $Form_n^{slave}$ possibility for additional windows existence, that obey $Form_1^{master}$, which allows to implement a multi-window user interface. Based on the methodology of object-oriented programming for the development environments of Windows-centric applications (RadStudioXE5, Visual Studio, etc.) $Form_1^{master}$ may be represented by sets $Form_1 PE$, that represent main visual parameters $MP_1(mp_1^1,...,mp_t^1)$ of form display, and their values $PP_1(pp_1^1,...,pp_q^1)$ that they can take and events set $ME_1(me_1^1,...,me_h^1)$ which are inherent in each form and are regulated by the development environment. Event me_1^1 may have "linguistic variables" set $EA_1(ea_1^1,...,ea_z^1)$, which are set with the help of natural language, each "linguistic variables" (ea_z^1) refers to the "solution container" containing the program code (cod_n) or its fragment in the knowledge base, and solves the task

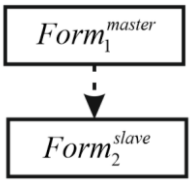
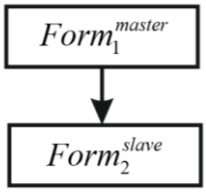
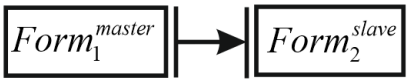
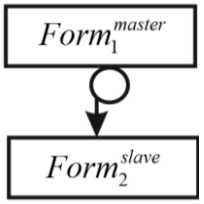
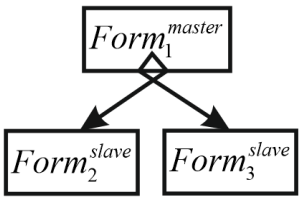
set by the customer in accordance with the TT requirements. Set CF_n describes presence of visual components CD_1^1 (Button, Grid, etc.) necessary for the implementation of the required graphical user interface. Performance CD_1^1 by such sets $PC_1^1(pc_1^1,...,pc_m^1)$ - main components visual parameters, and their values $PP_1(pp_1^1,...,pp_q^1)$, and events set $CE_1^1(ce_1^1,..,ce_w^1)$, where to each ce_w^1 belongs "linguistic variables" set $EA_1(ea_1^1,...,ea_z^1)$ which are set with the help of natural language and its description presented above [9]. For required properties graphical representation $(Form)$, it is proposed to improve the Constantine method, the main blocks are presented in the Table 1.

Table 1

Visual Elements Description Basic
Concepts Graphical Representation Basic Blocks

Graphical Representation	Description
1	2
	$Form_1^{master}$ - main user window without events and components; $mp_1^1, mp_2^1,..., mp_t^1$ - set of parameters, where 1 in uppercase denotes the membership of this parameter $Form_1^{master}$, and in the lower case its numbering in the set MP_1; $pp_1^1, pp_2^1,..., pp_q^1$ - the value assigned to a particular mp_t^1 (the value depends on the parameter and on the development environment).
	$Form_1^{master}$ must process some event me_1^1 (closing a window, activating or creating) to which "linguistic variable"is assigned ea_z^1 which refers to a specific "solution container" that contains a code fragment that performs certain actions.
	The knowledge base of all possible existing "Linguistic variables" (set $EA_1(ea_1^1,...,ea_z^1)$), which are set by the user of software and SM for CIS TPP automated design system
	A set of global variables (digital or logical) that transfer the necessary data between $Form_1^{master} \xrightarrow{\text{Domain}} Form_2^{slave}$ provided that $(Form_1^{master}, Form_2^{slave}) \in P$, necessary for further work
	Initialization $Form_1^{master}$ provided that no data is transferred to it.
	Transferring the result of the "Solution Container" to the graphical element CD_1^2 on the parameter pc_m^1, that is situated at $Form_2^{slave}$. Example output to a graphic element Edit (CD_1^2) to parameter Text (pc_m^1) "Solution Container" result execution.
	Communication by data (variables)
	Communication on the control of events CD_1^2 visual elements through the "solution container"

Continuation of Table 1

	Flow - call $Form_1^{master}$ in $Form_2^{slave}$. Used to graphically mark the execution of asynchronous and synchronous operations, which allows the execution of long operations and in parallel with this work with the interface $Form_1^{master}$
	Parallel call - work with asynchronous operations, which allows for long-term calculations to interact with $Form_1^{master}$, and the result of the execution will be displayed in $Form_2^{slave}$.
	Serial call - working with the graphical interface $Form_1^{master}$ is blocked for the user until a long operation is performed, the result is displayed in the $Form_2^{slave}$, after the end of the user's work with $Form_2^{slave}$ by results of which it will be closed, the user can continue working with $Form_1^{master}$.
	$Form_1^{master}$ calls $Form_2^{slave}$ as a coroutine that uses the principle of multi-threaded code, without requiring the presence of multiple threads, and can be performed within a single thread. Within these studies, when developing automated control system of technological production preparation, coroutines are useful for implementing finite state machine methods, actors model, and generators (iterators).
	The loop is intended for instructions set repeatedly execution organization until the conditions are met. "Solution container" which implements a cycle belonging to an event (me_1^1) $Form_1^{master}$ or event (ce_w^1) of an element CD_n^1 on its implementation initializes the transfer of the result (just a call) $Form_2^{slave}$.
	Branching "Solution container" provides execution of certain commands sets (command) under conditions of truth of some logical expression. It can exist with one branch, with two branches, with several conditions. When "Solution Container" is run, which belongs to the $Form_1^{master}$ depending on the fulfillment of the truth conditions, can initialize / transmit the result $Form_2^{slave}$ or $Form_3^{slave}$

Based on the proposed descriptions in Table 1, the main blocks of the graphical representation of the software and SM visual elements for CIS TPP description basic concepts, we give an example of the fragment of the development of the main window of the "Automated system for accelerometer production technological process design "AcCAM" copyright certificate of Ukraine No. 65348 of 16.05.16 with the using of "Automated System for Designing Software for the CIS TPP "CAD-Programming Code", author's certificate of Ukraine No. 65348 of 09.11.2017, in which a new methodology and concept for solving the automation task were implemented.

Let us represent developing system "AcCAM" as a visual windows interaction logically grounded structure, which is necessary for the task set by the customer solution. The enlarged graphical representation of the conceptual scheme and the main visual elements of the "AcCAM" program are presented in the Figure 1.

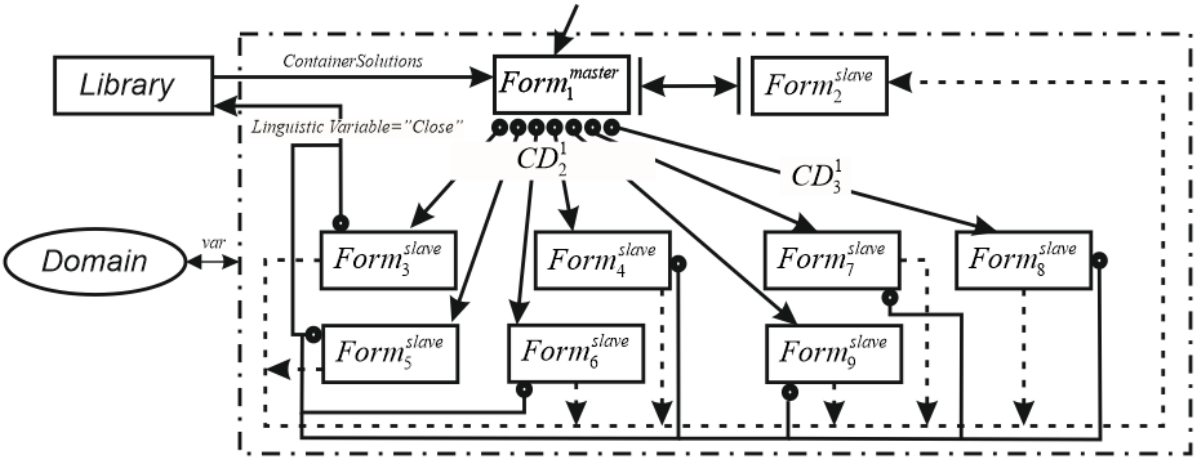


Figure 1 – **Enlarged Graphical Representation of the Conceptual Scheme and the Main Visual Elements of the "AcCAM" Program**

Let us describe the main graphic elements of the conceptual scheme:

$Form_1^{master}$ – the user initializes the main window, without data transmission, the fragment of the visual components structure is shown in Figure 2, and the description of the main parameters and events in Figure 3.

$Form_2^{slave}$ – DataModule contains not visual elements of work with the database and is called as an integral part of $Form_1^{master}$;

$Form_3^{slave}$ – window for adding new stages, is called using a visual element PopupMenu, which is bound to the parameter PopupMenu by right-clicking on an element DBGidEh, which is on GroupBox parameter Caption:= "Stage name";

$Form_4^{slave}$ – window for adding new equipment, is called with the help of a visual element PopupMenu, which is bound to the parameter PopupMenu by right-clicking on an element DBGidEh, which is on GroupBox parameter Caption:= "Equipment name";

$Form_5^{slave}$ – window for adding new operations, is called with the help of a visual element PopupMenu, which is bound to the parameter PopupMenu by right-clicking on an element DBGidEh, which is on GroupBox parameter Caption:= "Operations";

$Form_6^{slave}$ – window for adding new transitions, is called using the visual element PopupMenu, which is bound to the parameter PopupMenu by right-clicking on an element DBGidEh, which is on GroupBox parameter Caption:= "Tansitions";

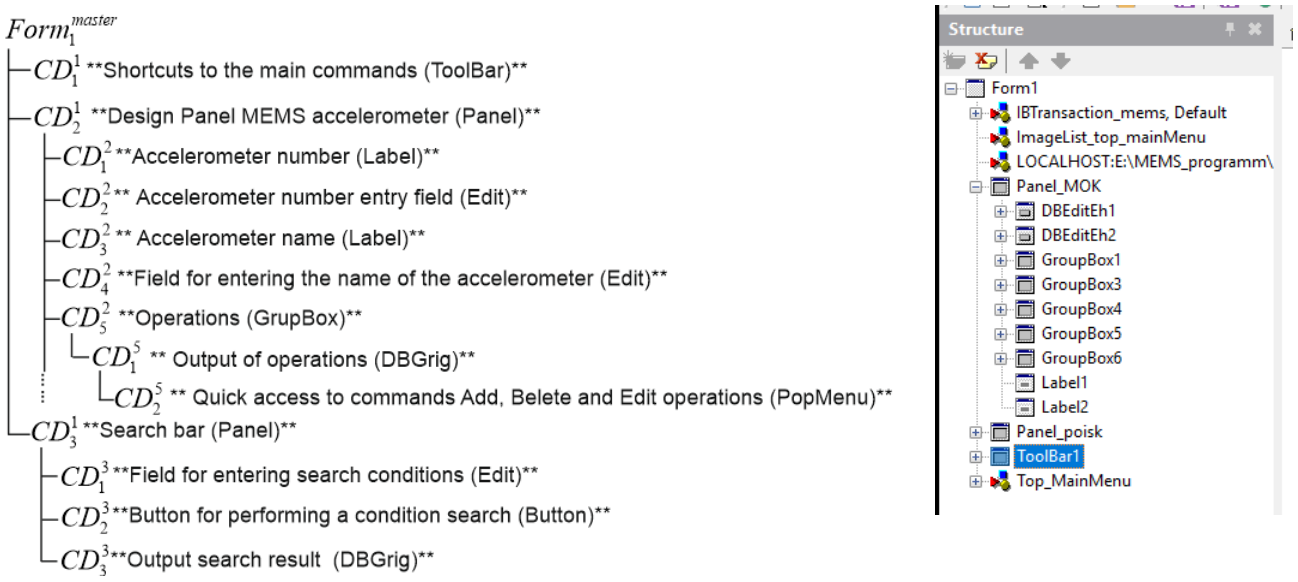
$Form_7^{slave}$ – window for editing transitions, is called with a visual element PopupMenu, which is bound to the parameter PopupMenu by right-clicking on an element DBGidEh, which is on GroupBox parameter Caption:= "Tansition parameters";

$Form_8^{slave}$ – called through the graphic element CD_2^3 (Button), which is on CD_3^1 (Panel Search bar), which belongs to $Form_1^{master}$.

After performing the actions in $Form_3^{slave}, Form_4^{slave}, Form_5^{slave}, Form_6^{slave}, Form_7^{slave}, Form_8^{slave}$ required by the user, the user performs the action with the help of visual elements Button1 and Button2, Button1 performs the "Save" action, and Button2 performs the "Close" action. Figure 1 shows the action LinguisticVariable with name "Close", which refers to the Library of CAD-Programming Code automated system and returns a ContainerSolution that contains the program code. Domain – contains all necessary variable names that are necessary for data transfer within the developed program «AcCAM».

Initially, the tree of automated system "AcCAM" main window all visual elements belonging was developed. At the request of the customer, the development environment was "RadStudio XE6". Main window tree structure fragment with visual elements designed in the "CAD-Programming Code

automated system" (a) and the result of generating the given structure in RadStudio (b) is shown in Figure 2



a) b)

a) Main window tree structure fragment with visual elements "AcCAM"
b) Resulting tree in development environment RadStudio XE6

Figure 2 – **Result of Main Window Given Structure Generating "AcCAM" Using "CAD-Programming Code"**

Fragment of main program window main parameters and events graphical description "AcCAM" ($Form_1^{master}$) without visual components is shown in Figure 3.

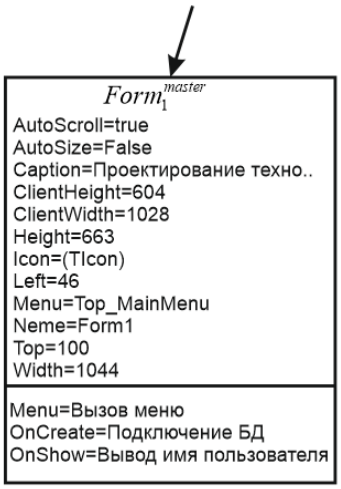


Figure 3 – **"AcCAM" Main Window Parameters and Events**

Figure 3 shows the minimum permissible parameters and their values set of for generation of an empty form with the name " MEMC Accelerometers Production Process Design" with an icon and a binding at $Form_1^{master}$ of visual component Top_MainMenu. This form has also to process events OnCreate, OnShow, Menu the program code of which is set with the help of linguistic variables (in this example the Russian language was used) "Call the menu", "Connect to the database", "Output user's name" which are set in natural language.

Solution container example (program code) for the linguistic variable "Connect to the database" generated by the "CAD-Programming Code" automated system and adapted by the programmer for the name of the components IBDatabase, IBTransaction, etc. is presented below.

```
procedure TForm1.FormCreate(Sender: TObject);
begin
IBDatabase_mems.Open;
IBTransaction_mems.StartTransaction;
IBDataSet_nac_etap.Active:=true;
IBDataSet_nac_oborud.Active:=true;
Panel_poisk.Visible:=False;
Panel_MOK.Visible:=True;
end;
```

Solution container example (program code) for the linguistic variable "Output user's name" is the following:

```
procedure TForm1.FormShow(Sender: TObject);
var
   UserName,result : string;
   UserNameLen : Dword;
begin
     UserNameLen := 255;
     SetLength(userName, UserNameLen);
  if GetUserName(PChar(UserName), UserNameLen) then
     begin
     Result := Copy(UserName,1,UserNameLen - 1);
     Form1.Caption:=Form1.Caption+result;
  end
else
   Result := 'Unknown';
end;
```

On an example the description of the basic parameters and events and also structure $Form_1^{master}$, generation of the developed automated system "AcCAM" program code was implemented for the environment RadStudio XE6. Figure 4 shows the result obtained in the "Designer window"

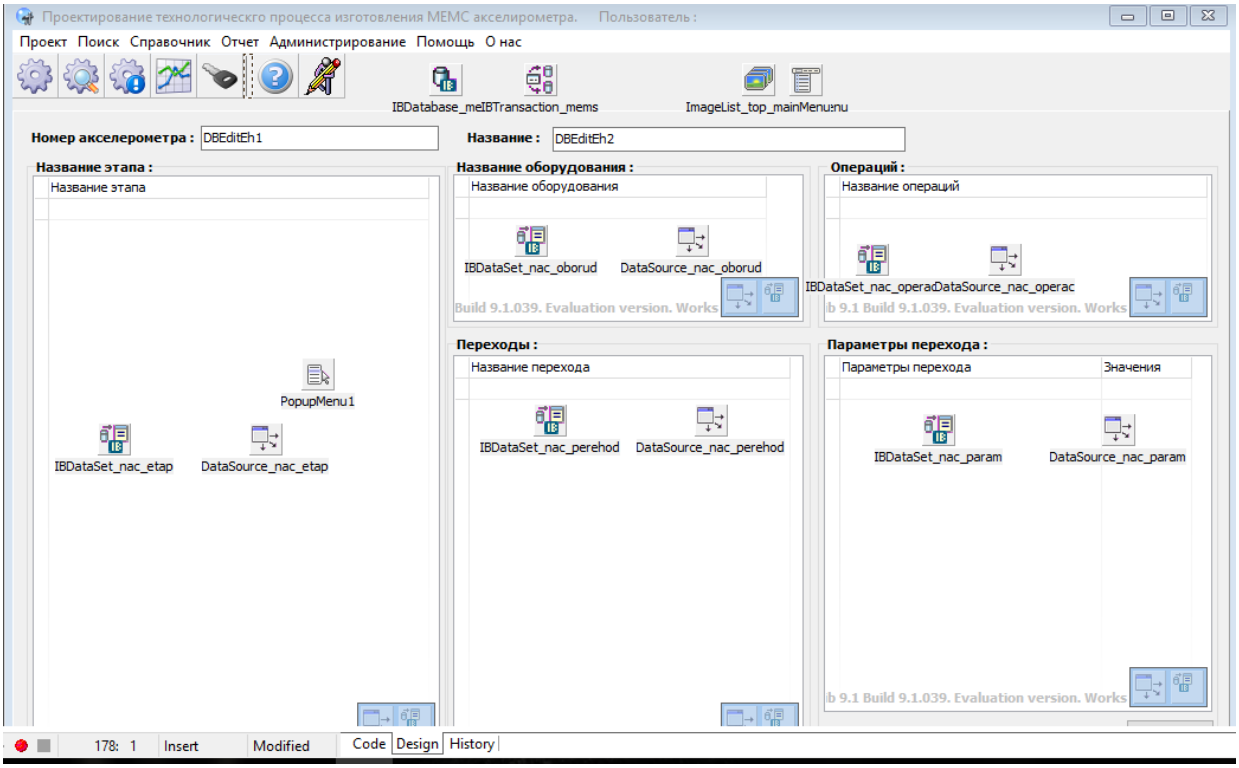


Figure 4 – **"AcCAM" Automayed System Main Window ($Form_1^{master}$) Obtained Result**

Conclusion. A graphical representation of software and modules visual elements description conceptual scheme and basic concepts for solving the problem CIS TPP design automation. It is implemented at "CAD-Programming Code". It allowed to obtain the following results with "Automated system for accelerometer production technological process design "AcCAM" design: reduce the user interface design time up to 15%, writing code up to 30% (it is possible to increase if the database is expanded with "Solution Containers"). At TT compilation early stage the user interface was shown and coordinated with the customer's preferences. In the future, it is planned to develop own language and syntax for describing software design for CIS TPP based on the use of natural language.

References.

1. Дэвид А. Марка, Клемент Л. МакГоуэн Методология структурного анализа и проектирования SADT. -М.:1993, 243с

2 Thomas A.Bruce. Designing Quality Databases with IDEF1X Information Models. - New York: Dorset house publishing, 1992. - 548 p.

3 Barker R. CASE Method. Entity-Relationship Modeling. - N.Y.: Addition-Wesley Publishing Company, 1991. - 431 p

4. Design/CPN User's Manual, Version 1.5: Meta Software Corporation. - Cambridge, Massachusetts. - 1989. - 387 p.

5. Jensen K., Rozenberg G. High-level Petri Nets: theory and application. - Berlin: Spingler-Verlag. - 1991. - 120 p.

6. Гейн К., Сарсон Т. Системный структурный анализ: средства и методы. – М.: Эйтэкс,2010.-545с.

7. V. Yevsieiev Program code automated system development at early stage of software life cycle // Наукові праці донецького національного технічного університету. Серія: «Обчислювальна техніка та автоматизація», №1(30)-2017. Покровськ .,-157с. с.69-78с ISSN 2075-4272

8. I. Nevlyudov, V. Yevsieiev , S. Miliutina, K. Kollesnyk Object semantic model for life cycle model “Jamp // CAD in Machinery Design. Implementation and Educational Issues. 25 Proceedings of Polish- Ukrainian Conference CADMD’2017, October 20-21, 2017, Bielsko Biala, 31-32 P

Стаття надійшла до редакції 20.03.2018