Algorithms, Mechanisms and Procedures for the Computer-aided Project Generation System

Anton O. Butko¹*, Aleksandr P. Briukhovetskii², Dmitry E. Grigoriev¹² and Konstantin S. Kalashnikov³

¹Department of Technologies of the Integrated Computer-aided Systems, Moscow Aviation Institute (National Research University), Volokolamskoie highway, 4, Moscow 125993, Russia.
²Department of Radio Engineering Devices and Antenna Systems, National Research Institute "Moscow Power Engineering Institute", Krasnokazarmennaya st., 14, Moscow 111250, Russia.
³Department of Radio Physics, Voronezh State University, Universitetskaya sq., 1, Voronezh 394018, Russia.

*Corresponding author
²Orcid: 0000-0002-0160-0724

Abstract

We suggest a simple, fast algorithm for the automated project generation based on the data integrated in the invariant structural-parametric model. The sample system is built (based) on this algorithm, and it includes project management software and interface module. It allows user to prepare and compose fully described project within the Microsoft Project application. There is no manual data input or conversion needed. This solution can be used while dealing with any task requiring process modeling and optimization, for example, data compression, etc.

Keywords: Project management, Automation, programming, Data conversion, Data compression, Data transformation, Modeling, Structural modeling, Parametric modeling.

INTRODUCTION

System for the automated generation of projects is a software product designed for computer-aided drafting (compiling lists and tasks) for the purpose of calculation and optimization of any process, including the production ones (organizational and technical solutions – OTR). The system includes several modules and is based on the invariant model provided by a structure-parametric modeling (SPM). To describe the model, one uses linguistic capabilities and translators. The output is a complete set of descriptions of the project, which can be loaded, for example, into Microsoft Project. This eliminates the need for manual data migration and settlement. A key element of the system is a software interface for converting data from the structural-parametric model. The results of this development are intended to be applied for both composition and description of the varied data compression processes.

THE MATHEMATICAL INTERPRETATION

Let us consider the mathematical interpretation of the functional aspect of the system interface. The formalization is carried out using the fundamental foundations of mathematics [1, 2] and set theory [3, 4]. The results of these studies lead us to the conclusion that the process of the project generation may be presented at a set-theoretic level, by the model combining the three main interrelated mechanisms:

$$\Pi(C) = \Pi^I(C) \cup \Pi^II(C) \cup \Pi^III(C),$$

where

$$\Pi^I(C)$$ produces analysis of the object and extracts the design data on the basis of the invariant product model;

$$\Pi^II(C)$$ converts the object structure on the basis of the technical solution;

$$\Pi^III(C)$$ performs both processing and transformation of resources (labor and material) and loads the obtained conversion results into the project management system in order to complete the project.

Let us present a description of the mechanisms.

$$\Pi^I(C) = \{A, N, B^E, R^K, G^C, \rho^I\}.$$  

Here A is a set of model elements, N is a set of model parameters; $B^E$ – set of link sets (may be interpreted as $R^K$); $R^K$ are the relations which determine the affiliation of parameters to elements; $G^C$ is a process protocol; $\rho^I$ is a set of procedures that support processing elements, parameters and relations in the model. We can see that the procedures to be determined will include software tools for the operation with the structural-parametric database, solving the following crucial tasks: extracting elements, extracting parameters, processing links.

$$\Pi^II(C) = \{A, N, B^E, R^K, G^C, \rho^II\}. \quad (1)$$

Here A is a set of model elements; N is a set of model parameters; $B^E$ is a set of link sets; $R^K$ are the relations which determine the affiliation of parameters to elements; $G^C$ is a process protocol; $\rho^II$ is a set of procedures that support the mechanisms of transformation of the collected project data into the import format of Microsoft Project. The composition of the procedures defined in the

$$\rho^II = (f_5, f_c). \quad (2)$$
FORMING THE TASK LIST

The SPM apparatus analysis shows that the structural-parametric model in its relation to its conditions and tasks can be represented as this set of components:

\[ M_{SPM} = (A, N, R^{AA}, R^{AN}) \]

(4)

where \( A \) is a set of model elements, \( N \) is a set of model parameters, \( R^{AA} \) are the relations (links) between the elements, \( R^{AN} \) are the relations determining the affiliation of the parameters to the elements.

We describe the sets included in the SPM as

\[ A = \{ a_1, \ldots, a_n \} \]

(5)

where \( a_i \) is a single model element defined in the ELEMENTS block (of the model);

\[ N = \{ n_1, \ldots, n_n \} \]

(6)

where \( n_i \) is a single parameter of the model defined in the PARAMETERS block (of the model).

The model of the project resulting from the interface operation can be described by the set of the following components:

\[ M_p = (P, Q, R^{pp}, R^{pq}) \]

where \( P \) is a set of project tasks, \( Q \) is a set of resources, \( R^{pp} \) are the relations (links) between the tasks, and \( R^{pq} \) are the relations determining the belonging of the resources to the tasks.

In order to perform correct work of mechanism \( \Pi^I \), the item converted into a project task necessarily differs from the rest of the model elements. Firstly, it has to be a part of the solution, and, secondly, it is to be marked with a parameter marker. These are the properties needed.

Figures 1 and 2 show how the elements of the model are selected for their subsequent inclusion in the list of tasks. Let now describe the set of properties with the help of the table from Figure 1. This is example for a model with 5 elements and 7 possible properties (the dot indicates whether the element has a property). So, there we can find the possible model elements properties, the two of them marked as the ones whose presence makes the occurrence of the element in the future tasks list possible.

The tables \( a \) and \( b \) in Figure 2 present the approximate set of elements and their properties. The table \( a \) presents all the elements, and the table \( b \) implements the elements selection.

\[
\begin{array}{cccccccc}
  f_1 & f_2 & f_3 & f_4 & f_5 & f_6 & f_7 \\
  F(A) & & & & & & \\
\end{array}
\]

\[ Figure 1: Characteristics of the task-element \]
From here we select the condition for the occurrence of a model element in a future project as a task: 

\[ F(A) = (F_3, F_4). \]

The solution produced by means of the specified tables is 

\[ A = (a_2, a_3). \]

That each element possesses some specific properties can be described by a Boolean matrix of the following form:

\[
\begin{bmatrix}
F_1 & F_2 & \ldots & F_m \\
C_{11} & C_{12} & \ldots & C_{1m} \\
C_{21} & C_{22} & \ldots & C_{2m} \\
\vdots & \vdots & \ddots & \vdots \\
C_{n1} & C_{n2} & \ldots & C_{nm}
\end{bmatrix}
\begin{bmatrix}
d_1 \\
d_2 \\
\vdots \\
d_n
\end{bmatrix}
\]

The parameter \( c \) takes the value 0 or 1 depending on the properties of the element:

\[
c_{ij} = \begin{cases} 
1, & \text{if } F_i \in F(a_j) \\
0, & \text{if } F_i \notin F(a_j)
\end{cases}
\]

As the selected properties may belong to the multiple elements, the collections of items for each property should be represented as a separate set 

\[ K = \{ k_1, \ldots, k_a \}, \]

where we can see that \( k_i \) has a parameter marker. Then 

\[ \forall k_i : k_i \in A, \text{ but it may be that } a_i \notin K, \]

\[ A : A \supset K, \quad K : F(A) \subset F(K). \]

And the set of the elements included in the solution is 

\[ E = \{ e_1, \ldots, e_n \}, \]

where \( e_i \) is the item, for which the following condition is fulfilled: \( e_i \in ELRESH \) (the list of solution items). Then 

\[ \forall e_i : e_i \in A, \text{ but it may be that } a_i \notin E, \]

\[ A : A \supset E, \quad E : F(A) \subset F(E). \]

As we see, \( F(A) \) contains all the model elements, including support and geometric elements, their transmission into the project management system not required. Thus, \( F(E) \) contains only those elements that need to be transferred to the project for planning.

Finally, we select a set of elements possessing both properties: 

\[ D = \{ d_1, \ldots, d_n \}, \]

where \( d_i \) is the element with the parameter marker and is included in the solution. Then 

\[ \forall d_i : d_i \in A, \text{ but it may be that } a_i \notin D, \]

\[ A : A \supset D, \quad D : F(A) \subset F(D). \]

There \( F(A) \) contains all the model parameters, including support and geometric parameters, their transmission into the project management system not required. Thus, \( F(D) \) contains only those parameters that need to be transferred to the project for planning.

Hence, the set \( D \), as well as \( K \) and \( E \), is a subset of \( A \) and is formed by the intersection of the sets \( K \) and \( E \):

\[ D = K \cap E, \quad F(D) = F(K) \cap F(E). \]

Now we define the condition for the selection of the elements to be included into the project:

\[ \forall a_i \in D \left( (a_i \in E) \cap (a_i \in K) \right) \quad \text{or} \quad \forall a_i \in D \left( F(a_i) = F(a_i) \land F(a_i) \right). \]

As a result, the task of producing the project task list for the interface is reduced to the description of the function \( f_i \):

\[ p_i = f_i(d_i). \]

This feature is to be included into the mechanism of the SPM data conversion into the Microsoft Project data.

**CREATING A LIST OF RESOURCES**

The next task is a transfer of resources. It involves the decision similar to the presented above. As resources in the SPM are represented by the parameters of a certain type, we implement additional set \( V \):

\[ V = \{ v_1, \ldots, v_n \}, \]

where \( v_i \) is the parameter of the "resource" type. Then
∀ v₁; v₂ ∈ N, but it may be that v₁ ≠ N, N: N ⊆ V, V:F(N) ⊂ F(V).

We define all the types of the parameters via the characteristics table (Fig.3).

<table>
<thead>
<tr>
<th>f₁</th>
<th>f₂</th>
<th>f₃</th>
<th>f₄</th>
<th>f₅</th>
<th>f₆</th>
<th>f₇</th>
</tr>
</thead>
<tbody>
<tr>
<td>F(N)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Figure 3**: Characteristics of the task-element

The properties we are interested in are the parameter types "working resources" and "material resources".

Thus, we can determine the condition for sampling the parameters to be included into the project as a certain type of resources in the following way:

∀ nᵢ ∈ V (F(nᵢ) = F₄(nᵢ) ∨ F₅(nᵢ)).

As a result, the task of forming project resources for the interface is reduced to the description of the function fᵢ:

qᵢ = fᵢ(vᵢ).

This feature will be included in the conversion mechanism of SPM data into the project data for Microsoft Project.

**THE MODEL AND ALGORITHM**

In general, we can conclude that the task of developing an interface can be reduced to the implementation of the mechanisms presented above and to the application of the modern software development tools. Thus, our three mechanisms will work in the following way:

Γ₁(C) – the mechanism is based on the built-in procedures for the SPM operating, allowing accessing the elements and parameters of the model in order to retrieve the necessary data;

Γ₂(C) – the mechanism is based on the algorithms for the data transformation implemented in the environment of Microsoft Visual Studio C++, its basic scheme consists in an iterative processing of the arrays of elements extracted from the model;

Γ₃(C) – this mechanism works on the basis of the algorithms for the data transformation, implemented in the environment of Microsoft Visual Studio C++. The basic scheme consists in an iterative processing of arrays of parameters (resources), extracted from the model. The transfer of resources into the project management system provides a macro created by means of VBA.

Therefore, the system interface will include three key algorithms:

1. Algorithm for data analysis and extraction; the algorithm represents a cycle with sequential selection of elements and parameters and their further analysis (verification conditions), aiming at the generation of a project structure.

2. Algorithm for data conversion and transfer; the algorithm represents a cycle of the serial conversion of the obtained structure of the project into a format suitable for its input into the project management system.

3. Algorithm for the generation of the list of resources; the algorithm is a cyclical processing of resources parameters for resources allocation (by marker validation) and the generation of their list in the format of the project management system.

Before a software system development, the study is conducted, in order to identify the possible problems and peculiarities, taking into account the available software products that can be used as the components of the system.

Depending on the a priori accepted conditions, there can be several design algorithms. We should consider the most important factors that, in fact, are the decisive criteria when choosing a suitable variant for the algorithm implementation.

First such factor is the presence or absence of tools to work with formats and internal functions of the automated system (AS) and detailed guidelines for their use.

In order to develop an algorithm, which works with the formats of different systems, first of all, you must either clearly represent the structure and the form in which technical information is stored in formats of AS, or have a fully described and extensively worked on software tools that works with this format. Quite often such tools and guidance on their application is provided by the developer together with the AS, usually the library functions are compiled into the standard Windows format DLL – Dynamically Linked Library. In order to work with these components programmer simply connect them to the project and immediately gets the opportunity to work with different internal functions of AS [5]. In this case, the task of implementing the interface becomes simpler, and the work is reduced mainly to the maintaining software data conversion.

Different situation arises, if the AS producer does not provide the documentation and other means necessary for external developers to operate the features and formats of AS and just gives a detailed description of all the formats and access methods. The system may support only its own specific and closed formats for information import/export and not the common ones. In this case, the programmer would have to write all the functions performing the needed information/data extraction and processing, the solution that may require the application of heuristic methods. This will inevitably affect the cost of the designed product, in some cases serving as a possible reason for the rejection of the choice of this particular AS, with subsequent revision of the desired system integration method.
The second factor is whether the AS supports one or more of common import formats, whose detailed content description and the structure, as well as their methods and tools, are available for developers. As a rule, for them there are already many third-party software processing tools, including the open source ones.

The next option is: the AS producer does not provide any supporting tools and data for external developers, but the system supports additional import formats, content and description of which are widely known and available for the programmer, as a DXF format for CAD systems, for example. In this situation, the most difficult work is reduced to the transformation of the data/information and providing its accuracy and integrity [6]. If the support of common formats is missing and there is no software capable of operating the internal functions of the system, i.e. it is closed as its format, one would have to make a difficult decision, or to select a different AC, or a different method of integration of such systems.

The third factor consists in the size of the budget and the time frame allocated for the project. They radically affect the choice of methods and means of integration, especially for Russian companies.

Now, on the basis of the above stated, we can present a general description of the basic options for the content of the algorithm constructing the software interface, aiming at the integration of the selected systems. These options differ from one another by presence or absence of certain operations and by the sequence of operations in the algorithms they provide.

Option 1. Direct conversion of the formats from one to another is carried out without running the AS. The interface works with the file of the system from which you import by means of the internal functions of this very system. The result is stored in the importing system’s format. In this case, no AS installed on a computer is required, which is one of the advantages of this option.

Option 2. The formats are converted from one to another through the intermediate format, without running the AS. This option presupposes that a certain external auxiliary converter is used, performing the transformation of data from the intermediate format into the importing system’s format. In this case, there is also no need to install AS.

Option 3. Data conversion triggers one of the AS and utilizes its internal functions. The interface operation is only possible on the computer with the certain AS and the particular API functions library installed (if the latter is not supplied with the interface already).

Option 4. The data conversion triggers both systems and utilizes their internal functions.

As it has been previously mentioned, as a system data source the SPM Modeler is selected, and its applicability is determined by the following reasons:

- SPM is invariant to the semantics of the data in the context of the specific object modeling;
- the Modeler allows generating geometric models of complex topology;
- SPM can be used in ”assembly diagrams” nodes models;
- SPM S(A) is based on the structural properties of the systems and can be used in data formulation for the elements of the simulation object A, the properties of F and the relations between the elements and the properties of R;
- matrix apparatus and system of relations are the algorithmic basis for the solution of a wide range of tasks by applying a modular principle of information management during the models processing;
- SPM provides extensive features for parameterization and structuring.

To illustrate the operation of the system, Microsoft Project is selected as the project management system, as it is characterized by a universal availability and a great variety of functions implementing the tasks regulated in the MRP-I and MRP-II.

The choice in favor of this class of systems is the result of the suggestion accepted to introduce such AS as the component of the integrated solution, instead of the ERP, that, for the domestic enterprises, appears to be hard to implement and apply. The more so, as the structure of the information coming as the original data in SPM is almost equivalent to the one in ERP, or PDM, systems.

Theoretically, the choice of the AS is not a key factor, significantly influencing the applicability of the suggested scheme (PS (Modeler) → interface → PM (PDM, ERP in existing solutions)), due to the invariance of the latter relative to the specific systems. Thus, the solution (the procedure) is expected to have sufficient adaptability, at least within our field of study and research.

The results of this analysis based on the project operation tasks and the AS selected, a decision is made to develop the algorithm using the combined approach involving the following interface operation sequence: the source data is extracted by means of the SPM special internal functions, used for the external access to the SPM format without triggering the entire complex; and then the information is structured, verified and converted to the import format of the project management systems.

Part of the data is generated in intermediate format, for its subsequent loading into the system by means of a special macro integrated in the project management system [7]. The resulting import file is opened in Microsoft Project environment and the generation of the project framework begins, followed by the loading of resources and the
The results of the project design are presented as a high-level algorithm of the interface module operation, described following the IDEF0 methodology [8, 9]. The corresponding chart is shown in Fig. 4, 5.

Fig. 4b presents the deployment of the root diagram showing the basic stages of the project generation, implemented by means of the three interface mechanisms described above.

Fig. 5a, presents the deployment of the first block of the first level diagram, demonstrating the main stages of the extraction process. Fig. 5b presents the deployment of the second block of the first level diagram, showing the basic steps of the data conversion process. Fig. 5c shows the deployment of the third block of the first level diagram, illustrating the main stages the project data loading into the Microsoft Project takes.

**Figure 4:** Upper level diagrams: a) root diagram; b) first level diagram (shows usage of ПІПІІ mechanisms)
As we can see, the diagrams below present the charts that depict the most important stages of the process of the interface operation and, in fact, constitute together the conceptual scheme of the project generation procedure.

Fig. 6 shows high-level algorithm for the process of data transition from SPM to Microsoft Project:

- $\Pi_1(C)$ operates within block 6;
- $\Pi_2(C)$ operates within blocks 7, 10, 11;
- $\Pi_3(C)$ operates within block 13.

**Figure 5:** Blocks deployment for the first level diagram: a) diagram 1, second level; b) diagram 2, second level (shows usage of $f_s$, $f_P$, $f_C$ functions); c) diagram 3, second level
The diagrams present the main stages and blocks of the process of the interface operation. Most of these blocks include cyclic algorithms for the operation with the arrays of elements and parameters of the model to be converted into tasks and project resources.

The suggested solution was tested as a part of the corporate information system for management and planning, deployed at a space plant. The system is based on the complex modeling product (SPM), with the Microsoft Project environment managing the project applied. This complex was used for the technical products manufacturing planning. A trial period shows significant reduction of the data input time and errors quantity. The period needed for the data entry into the project management system is reduced almost by two times. But there are still some restrictions encountered during the testing: some versions of the Microsoft Project applications cannot support large models that contain huge numbers of elements (from 100 000 to 1 000 000). Given this, we have to test the models of the middle and small sizes. The suggested solution allows avoiding manual entry of data into the project management system, and thus the number of errors is reduced. We expect a significant reduction in the time it takes to build the projects for planning through the automated data transfer from the modeling system.

CONCLUSION

The suggested simple and fast digital algorithm converting data for project generation allows creating and managing various process models, in order to find optimal (balanced) solution. Projects can be loaded into any Project Management system for the subsequent analysis and optimization. Suggested solution provides the possibilities to avoid multiple errors produced by the manual data input and conversion.
ACKNOWLEDGEMENT

This work was supported by the Ministry of Education and Science of the Russian Federation (project No. 14.577.21.0163).

REFERENCES


