UNIVERSITY OF ZIELONA GÓRA
Faculty of Management
and Faculty of Mechanical Engineering
In association with the Design Society

FOURTH INTERNATIONAL SEMINAR AND WORKSHOP
7th - 9th October 2004
Zielona Góra - Rydzyna
POLAND

Engineering Design in Integrated Product Development
Management of Design Complexity

IMPLEMENTATION OF THE DEPENDENCE MATRIX IN CONCURRENT ENGINEERING PROBLEMS

R. KIELEC, W. BABIRECKI, M. SĄSIADEK
University of Zielona Góra
Department of Mechanical Engineering
e-mail: r.kielec (w.babirecki, m.sasiadek)@iizp.uz.zgora.pl

Keywords: dependence matrix, design process, engineering calculations, assembly sequence, planning

Abstract: *In the paper applications of dependence matrix in the problems of design process planning, setting algorithms of engineering calculations, and determination the best assembly sequences are presented. In each application effectiveness of the matrix representation and its proper transformation has been shown. The illustrative examples include: rationalization of jack-screw calculation, planning of the design process, and generating feasible mechanical assembly sequences.*

The rapid development of technology and severe market competition, contribute to shortening of a product's life cycle and to speeding up the introduction of it to the market. Traditionally, the sequential realization of stages of product design and manufacturing should be carried out as quickly as possible, while taking into account specific features of the subsequent stages. The human potential should be efficiently employed to meet demanding and often conflicting requirements. To do this it should be aided with a suitable computer system. Such a system should allow for the fast generation of partial solutions, should assist their evaluation, and should make possible the assessment of their influence on the organization of the whole process. Thus a proper method is sought for improving the designed process, by means, among others, elimination of unwanted feed-backs because they increase the total time and cost of a product development.

The authors believe that the matrix representation of the problem to be solved ensures convenient and relatively simple way of recording necessary information and a useful tool for finding out the best solution of the problem.

In this paper three kinds of engineering problems are considered as examples of the matrix method implementation: organization of the design process planning, algorithmization of engineering calculations, and generation of the sequences of assembly operations.

1. IMPLEMENTATION OF THE DEPENDENCE MATRIX FOR ALGORITHMIZATION OF ENGINEERING CALCULATION

One of many problems, which mechanical engineers encounter in everyday work, is calculation of individual or assembled elements. Relevant literature presents sets of equality and inequality relations as well as different kinds of data in form of tables and graphs that aim at this problem. Also solutions of exemplary tasks in ready-to-use forms are available. Textbooks and manuals do not, however, contain every possible example that may appear in engineering practice. Many times an engineer copes with a problem of applying a theoretical algorithm into a real-life situation, because the set of input and output variables differs from that, which can be found in the literature. Such a situation brings about a serious difficulty which results in the necessity of searching for a suitable sequence of relations. This results in extended time of the task realization. Therefore there is a need to provide a user with a tool to face this difficulty. This section of the paper is aimed at presentation of the dependence matrix application for creating calculation algorithms ready to use in the real-life situations. On this example, the usefulness of the proposed method to control the data flow and variables interdependence will be presented.

1.1. Application of the 0dependence matrix for jackscrew calculation

Let's consider a problem of designing a simple jackscrew. An engineer facing such a problem applies, in general, a set of relations from relevant literature and a ready-to-use calculation algorithm. If the real design problem is the same as that solved in the handbook then the task of the engineer is easy. However if it is not the case, it is required that other calculation algorithm must be found out, which will be adapted to the real conditions. The flow of data and order of calculations may be essentially different from the example at hand. For creation of a suitable calculation algorithm application of the dependence matrix is very helpful. The procedure incorporates the following five stages:

Stage 1

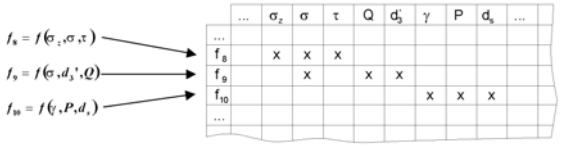
Building a matrix that maps dependences between relations and variables relevant to the calculation task. At the beginning all relations appearing in the task are recorded in a general form, e.g.:

relation $\sigma_z = \sqrt{\sigma^2 + 3\tau^2}$ is transformed to the form $f_8 = f(\sigma_z, \sigma, \tau)$

relation $\sigma = \dfrac{Q}{\dfrac{\pi d_3'^2}{4}}$ is transformed to the form $f_9 = f(\sigma, d_3', Q)$

relation $\gamma = arctg\dfrac{P}{\pi d_s}$ is transformed to the form $f_{10} = f(\gamma, P, d_s)$

Next, relations and their variables are recorded in the form showing the mutual dependences as shown in the figure below:



Stage 2

Arranging the matrices in the following ways:

- grouping the variables according to their types (functional, material, geometric, technology),
- grouping the relations according to their types (equality, inequality, table relation, functional relation),
- determining the variables status (input, output, intermediate) and separating the input variables.

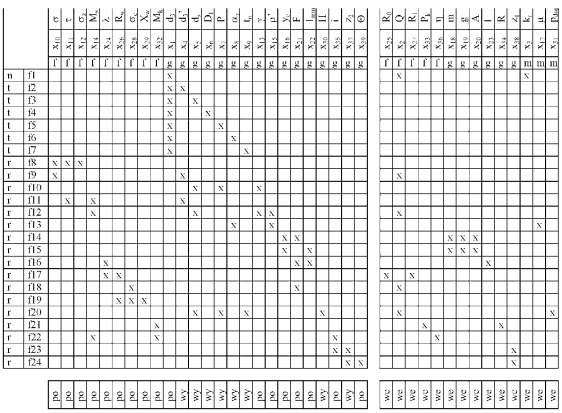


Fig.1 *The ordered dependence matrix example*

The created and well-ordered matrix for this illustrative example is presented in figure 1.

3 Stage

The dependence matrix decomposition.

Decomposition of the complex task usually results in obvious benefits. The task decomposed into a smaller number of possibly independent sub-tasks is simpler and faster to solve, especially in the case of many complicated relations between variables. Out of many available decomposition methods, the one described in [10,11] was chosen for this example. The decomposed matrix of the discussed example is presented in figure 2.

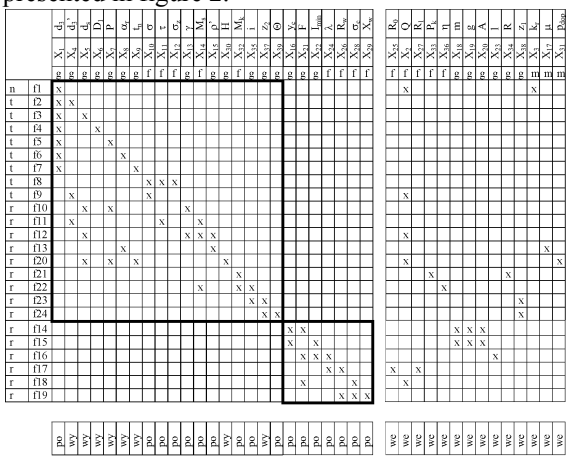


Fig.2 *The decomposed dependence matrix of the example*

Stage 4

Creation of the calculation algorithm.

This process is carried out separately for each block of the decomposed matrix. Once all available input variables are shifted to the right side columns of the matrix, then an extra column is added, in which a so-called row sum is recorded. This number denotes the number of unknown variables to be solved (output or intermediate) that appear in a given relation. If the set of input variables was entered correctly, the process of creation of the algorithm is, in the example considered, relatively simple. The relation, for which the value of the row sum equals 1, is taken out of the matrix and is recorded as the first relation of the created algorithm, because if the row sum is 1, the relation has only one unknown variable. In this example these are: relation *f1*, from which variable x_1 can be directly calculated, and relation *f21* from which x_{32} is calculated. If these relations together with just calculated variables are taken out from the matrix, the row sums of the consecutive relations equal 1. Now these relations are *f2, f3, f4, f5, f6* and *f7* and the resulting variables are accordingly: x_1,x_1,x_1,x_1,x_1,x_1. They are dealt with similarly and recorded in order of their disappearance from the matrix. Then they are cancelled along with variables that were calculated from them. The described activities are repeated until all variables are deter-

mined. First two steps of the algorithm creation for the first block from figure 2 are presented in the figure 3.

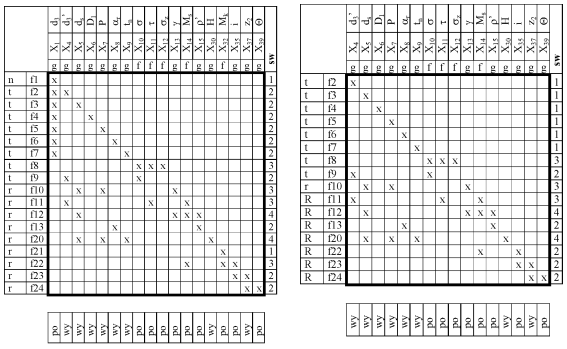


Fig.3. *Two steps of the algorithm creation*

Stage 5

Recording the algorithm.

The final stage of the calculation algorithm creation process is its proper recording. It is recommended at first to set up a table, in which consecutive calculation steps are in the rows, and the pertinent relations and variables, which are to be calculated, are placed in the columns. Alternatively, the algorithm can be represented in the form of a tree. Both forms of the calculation algorithm are presented in the figure 4.

etap	obliczyc	z relacji
I	$x_1=d_3$	$d_3 \geq 1{,}27\sqrt{\frac{Q}{k_r}}$
II	$x_4=d_3'$	$d_3 \Rightarrow d_3'$
.	.	.
.	.	.
.	.	.
.	.	.
.	.	.
.	.	.
VII	$x_{35}=i$	$i=\frac{M_s}{M_s\eta}$
VIII	$x_{37}=z_2$	$z_2=z_1 i$
	$x_{39}=\Theta$	$\Theta=arctg\frac{z_1}{z_2}$

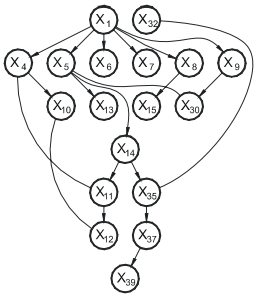


Fig.4 *A part of the calculation algorithm in the form of a table and a tree.*

1.2. Analysis of the dependence between the variables

The dependence matrix is very useful for the analysis of the dependence between the variables appearing in the calculation process. Very often the designer needs information like: which output variables are influenced by specific input variables, or from which input variables the sought output variables are dependent on, so "what depends on what" and "what influences what". It will be shown how to resolve this problem with the use of the dependence matrix.

Let us suppose that we want to determine which output variables are influenced by the input variable x_{33}. Based on the analysis of the algorithm described in section 1.1 we can infer from the dependence matrix that: input variable x_{33} influences output variables x_{32}, x_{35}, x_{37}, x_{39} (Fig. 5).

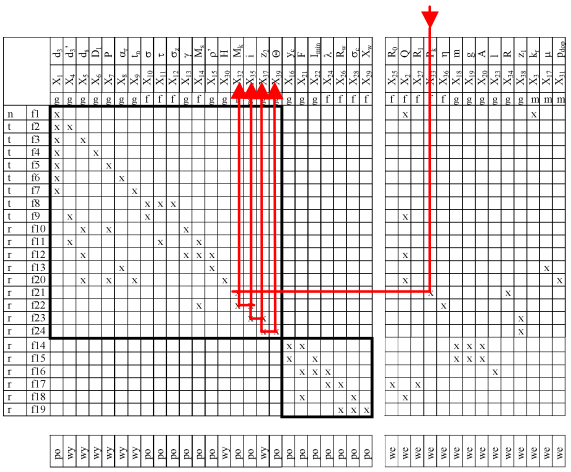


Fig.5 *Influence of an input variable on output variables (forward dependence)*

Similarly, by using the same matrix it is possible to determine on which input variables any output variable depends. Assume that the output variable x_{30} is considered. By means of inspection of the dependence matrix, it can be realized that the output variable x_{30} depends on input variables: x_2, x_3 and x_{31} (Fig.6).

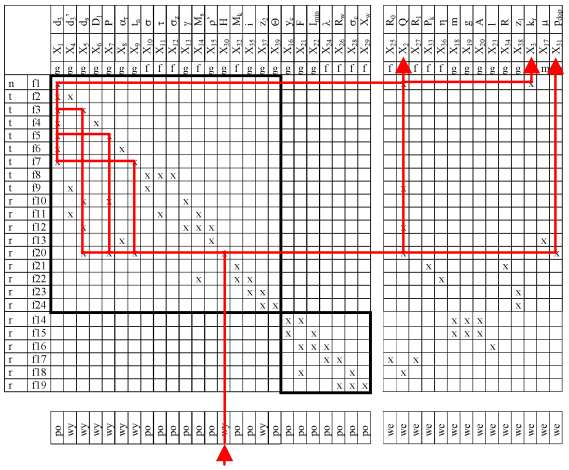


Fig.6 D*ependence of an output variable on input variables (backward dependence)*

To summarize, it should be stated that the use of dependence matrices in the representation of relations and variables appearing in the calculation problems benefits in:

a) creation of algorithms "accurately matched" with the actual design situations,

b) identification of the influence of input variables on output variables and their mutual dependence,

c) considerable simplification of the calculation task by decomposing it into smaller sub-tasks, which results in shortening the solving time [2, 3, 4,13]. Because the example presented in the paper is relatively simple, so the benefits from using the dependence matrix may seem not to be impressive. Because of lack of space this example has been chosen deliberately for the demonstration only.

In more complicated calculations consisting hundreds of relations and variables, the benefits are significant.

2. IMPLEMENTATION OF THE DEPENDENCE MATRIX IN THE DESIGN PROCESS PLANNING

In the real-live engineering design processes some tasks have to be carried out sequentially, but there are many, which may be executed in parallel, and also such, which need to be repeated i.e. they can be solved by iterations only [19, 20].
A suitable approach for dealing with such processes was suggested by D. Steward [22] and then developed by others. This approach has led to successful applications in many fields.
This method is based upon the record of sub-tasks in the form of dependence (structural) matrix in which numbers situated on the main diagonal represent sub-tasks.
Relations of information between the sub-tasks are recorded in the form of connection lines. Each sub-task, to be solved, need to be fed by a specific information from another sub-task and its solution generates some specific data of engineering value. An example of such a matrix for 17 partial sub-tasks is shown in figure 7.

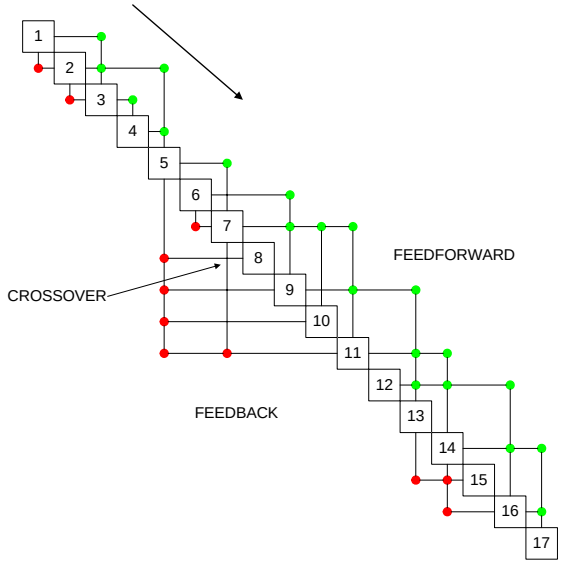


Fig.7 *Example of the Design Structure Matrix*

The design process mapped there begins with the sub-task in the left upper corner and proceeds towards the final sub-task, in the right bottom corner.
The informational connections are denoted with lines: horizontal lines denote outputs whereas the vertical ones represent inputs. Thus, lines below the main diagonal represent feed-backs.
Realization of the process shown in figure 7 begins with the sub-task 1, which is followed by the fulfillment of subsequent sub-tasks. It can be noticed that the sub-tasks may be carried out sequential, parallel or iterative. For instance, sequential realization takes place in the case of sub-tasks 3, 4, 5 and 7, 9, 11, 13. In the first case sub-task 3 takes in data from sub-tasks 1 and 2. After realization of task 3 the results are transmitted to the sub-task 4. Sub-task 4, on the other hand, delivers data for the performance of sub-task 5. Then the process proceeds similarly.
Simultaneous realization takes place in the case when two (or more) sub-tasks are carried out independently one from another, for instance sub-tasks 9 and 10.
Iterative realization means that one or more tasks have to be repeated, e.g. tasks 5 to 11. Under some circumstances iterated tasks can be carried out concurrently by permanent information exchange between these sub-tasks. The sub-tasks realized concurrently are both progressively and feed- backed coupled. For instance, in the matrix shown, sub-tasks 2 and 3, or 14 and 16 could be carried out concurrently.
The above-mentioned types of basic tasks structures are shown in figure 8.

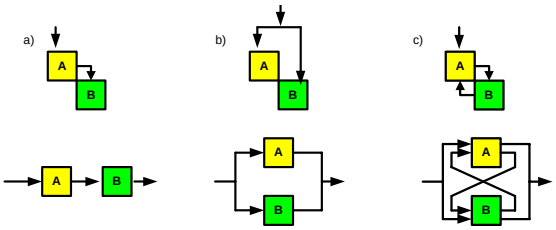


Fig.8 *Basic task structures a) sequential, b) simultaneous, c) concurrent.*

It is important to emphasize that the connection lines below the main diagonal indicate necessity to carry out the coupled activities in an iterative way, which is unfavorable. Therefore, if possible, they should be avoided by means of appropriate task re-arranging.

2.1. Example of industrial realization process improvement by means of design structure matrix

In table 1 sub-tasks of certain industrial realization process are listed and assigned with their realization times and cost. The staff of a production company has estimated these data. Information connections between the sub-tasks are essential part of the process. They have a strong effect on the order of the sub-tasks execution. The dependence matrix in figure 9 shows the original realization process as it was carried out in the company.

Table 1. *Basic activities, their times and costs, and the order of their realization in a company*

Task Number	Task Description	Realization Time	Realization Cost
1	OFERTA	150	40
2	OPIOFER	16	80
3	KONCTCH	60	40
4	KALKULA	40	40

Task Number	Task Description	Realization Time	Realization Cost
5	OTWZLEC	30	40
6	DOKKONS	1619	40
7	MATTRUD	16	40
8	ZMIANYK	308	40
9	INSTROB	86	40
10	EMIDOKK	45	28
11	DOKTECH	324	40
12	LISTDOS	42	35
13	ZAMMAT	100	35
14	PRZYIWY	100	28
15	ZMIAMIK	5	35
16	LISKOOP	452	40
17	DETKOOP	1892	28
18	DETSECO	270	28
19	MONTZES	1622	28
20	MONTPGL	2387	28
21	MONTCAL	537	28
22	BADANIA	250	32
23	DEMMAL	347	28
24	TRANSPK	288	28
25	MONTKLI	353	84
26	URUCHO	259	32
27	SPOTKON	12	40

The 27 sub-tasks of the industrial product realization process represented in the structural matrix, figure 9, include 43 progressive connections and 8 feed-backs without of intersections. Two iteration blocks are marked in the figure. They should be subjected to careful inspection.

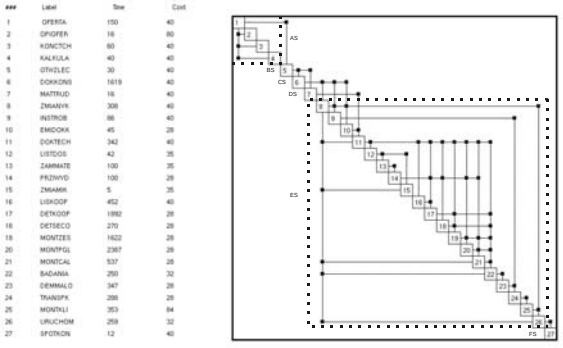


Fig.9 *Dependence matrix for the product realization process*

The total sum of realization times of all sub-tasks equals to 11 628 conventional units. If we assume that one feed-back causes one repetition of the looped sub-tasks, then the realization time of the process is equal to the sum of this time plus realization times coming from the repeated sub-tasks. Accordingly, the total duration of product realization will be equal to 40406 conventional units. Thus, feedbacks have caused the increase of realization time by 28778 time units. If the possibility of simultaneous realization of some sub-tasks is employed, then the total period of the product realization will be decreased to 39 912 conventional entities.
The total cost of the sub-tasks realization equals to 1025 conventional cost units. If feedbacks are taken into account then the process realization cost would equal to 3553 conventional units, which results in cost increase by 2528 units.

By means of the proper re-ordering of rows and columns of the dependency matrix the course of the process can be significantly improved towards minimal time and/or cost of the realization. During these operations all information links between sub-tasks remain unchanged; the only change is the order of their realization.
Some implementations of the method and effects gained have been described elsewhere [17, 19, 20, and 21].

3. USE OF DEPENDENCE MATRIX FOR DETERMINATION OF ASSEMBLY SEQUENCES

The assembly process is the last stage of product manufacturing. The goal of the assembly process is connecting the proper components into ore complex entities until the final product is received. The order of connecting the components i.e. assembly sequence strongly influences the assembly process. Establishing the best assembly sequence poses a difficult problem because the number of alternatives is usually great. It exponentially depends on number of parts to be connected. Effective order of component assembly should significantly reduce the product manufacturing time, therefore the generation assembly sequences methods and devices, as well as their estimation, should be helpful and should be used by a designer in the early stage of product designing and creating. There are many approaches to determine the order of assembly e.g. some methods directly describe and represent assembly sub-tasks [1, 5, 6, 9, 15, and 23] whereas others directly focus on determining a proper course of the assembly [14].
The method presented in the paper determines all possible assembly sequences. It belongs to "the advisory methods in designing" and in the authors' point of view it should be included in CAD systems.
The suggested algorithm for determining all possible assembly sequences makes use of the designed product geometrical data as well as order of the assembly operations. The geometrical data define all possible contact relations between product components. Order relations apply to every operation, and leave out operations, which can't be carried out earlier because of the next part of the assembly process course, that is because of the preventing the product from completion.

3.1. Description of generation assembly sequences algorithm

Algorithm of producing assembly sequences will be described on an example that is often presented in publications [1, 14, and 15] i.e. on pen composed of 6 components, fig. 10. This example is simple but complex enough to describe the suggested algorithm use.

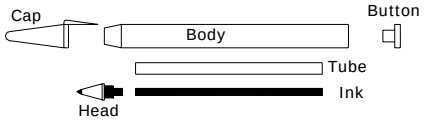
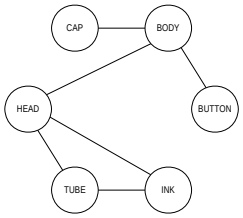


Fig.10 *Components of the pen*

The identification of all "contact" relations between product components is prepared on the basis of the product physical shape. The required contact relations can be recorded in the graph form and its matching matrix – named later construction matrix $\mathbf{M_k}$. (fig.11)



	CAP	HEAD	BODY	INK	TUBE	BUTTON
CAP			X			
HEAD			X	X	X	
BODY	X	X				X
INK		X			X	
TUBE		X		X		
BUTTON			X			

Fig.11 *Graph and matrix of pen components relation*

Construction matrix has n *x* n size, where n – is the number of pen's components. The matrix is symmetric one, which results logically from the condition of contact of two parts – if part A contacts part B then part B also contacts part A. Every contact relation is denoted in the matrix appropriate matrix cell as X.

On the basis of $\mathbf{M_k}$ all assembly operations of connecting two contacting parts can be determined. However, only some part of the theoretical operations is feasible. Moreover, the feasible operations must not be performed in arbitrary order. That's why all infeasible operations should be rejected. Also, for every specific operation it should be determined the ones, which cannot be preceded, so called blocking operations. A blocking operation is the operation, which makes it impossible or limits the complete product assembly. To identify blocking operations, a question should be considered concerning every operation: performing which operation(s) would make realization of a given operation impossible (or significantly difficult)? Such approach allows for elimination of the improper components connecting order. Generated assembly operations are recorded in the operation matrix form M_0 – fig.12. In the presented example 12 assembly operations can be determined. Firstly, operations 5, 6 and 12 should be cancelled because of their performance difficulty. Next, in the M_0 matrix for every feasible operation it is necessary to determine operations that block it or make it significantly difficult to carry it out – e.g. HEAD-BODY operation (or BODY-HEAD) is impossible to perform after the preceding performing operation 1 or 2, that is CAP-BODY (or BODY-CAP), operation 3 (HEAD-BODY) makes operations 7, 8, 11 significantly difficult to carry out, that's why it's also marked M_0 in the matrix. The size of assembly operation matrices results directly from M_k matrix and it is equal to the number of all contact relations between components. Therefore, every assembly operation describes connection only two parts.

	1	2	3	4	5	6	7	8	9	10	11	12	
1		X			X	X						X	CAP-BODY
2	X				X	X						X	BODY-CAP
3	1	2		X	X	X						X	HEAD-BODY
4	1	2	X		X	X						X	BODY-HEAD
5	X	X	X	X		X	X	X	X	X	X	X	HEAD-INK
6	X	X	X	X	X		X	X	X	X	X	X	INK-HEAD
7			3	4	X	X		X				X	HEAD-TUBE
8			3	4	X	X	X					X	TUBE-HEAD
9					X	X				X		X	BODY-BUTTON
10					X	X			X			X	BUTTON-BODY
11			3	4	X	X						X	INK-TUBE
12	X	X	X	X	X	X	X	X	X	X	X		TUBE-INK

Fig.12 *Matrix of operation - M_0*

It is assumed that the state S = 0 describes the beginning state to determine alternatives of components assembly. For this state the initial form of the matrix M_0 should be established.

The described operations are quite laborious but they allow one to determine all possible product assembly sequences. Next the generation of assembly order for pen from fig. 10 by the algorithm is presented.

The course of the algorithm:

On the basis of M_0 matrix starting operations are determined – that is such operations that can be firstly carried out. They are 7, 8, 9, 10, and 11.

1) We start from the operation 7 (HEAD-TUBE). On the fig. 13 in the $\mathbf{M_{k\_1}}$ matrix these parts are marked as a one sub-group.

	HEAD TUBE	BODY	INK	CAP	BUTTON
HEAD TUBE					
BODY	X			X	X
INK	X				
CAP		X			
BUTTON		X			

Fig.13 *$\mathbf{M_{k\_1}}$ matrix*

2) Afterwards there are two operations possible: BODY – (HEAD-TUBE) and INK – (HEAD-TUBE). From the conditions in the M_0 matrix it results that the following operation has to be INK – (HEAD-TUBE) operation.

	HEAD TUBE INK	BODY	CAP	BUTTON
HEAD TUBE INK				
BODY	X		X	X
CAP		X		
BUTTON		X		

Fig.14 M_{k_2} *matrix*

3) The next possible operation in the first loop of the algorithm BODY – (HEAD-TUBE-INK)

	HEAD TUBE INK BODY	CAP	BUTTON
HEAD TUBE INK BODY			
CAP	X		
BUTTON	X		

Fig.15 M_{k_3} *matrix*

4) Then operations CAP – (HEAD-TUBE-INK-BODY) or BUTTON – (HEAD-TUBE-INK-BODY) can be performed in any order (or simultaneously). After eliminating part CAP and BUTTON the matrix is complete and forms one group representing the analyzed pen assembly sequence.

Consequently we proceed with every starting operation and as the consequence we receive 12 possible pen assembly sequences that match to the conditions presented in matrixes from fig. 11 and 12

(((((HEAD-TUBE)-INK)-BODY)-CAP)-BUTTON)
(((((HEAD-TUBE)-INK)-BODY)-BUTTON)-CAP)
(((((TUBE-HEAD)-INK)-BODY)-CAP)-BUTTON)
(((((TUBE-HEAD)-INK)-BODY)-BUTTON)-CAP)
(((((INK-TUBE) -HEAD)-BODY)-CAP)-BUTTON)
(((((INK-TUBE) -HEAD)-BODY)-BUTTON)-CAP)
(((BODY-BUTTON)-((HEAD-TUBE)-INK))-CAP)
(((BODY-BUTTON)-((TUBE-HEAD)-INK))-CAP)
(((BODY-BUTTON)-((INK-TUBE)-HEAD))-CAP)
(((BUTTON-BODY)-((HEAD-TUBE)-INK))-CAP)
(((BUTTON-BODY)-((TUBE-HEAD)-INK))-CAP)
(((BUTTON-BODY)-((INK-TUBE)-HEAD))-CAP)

All alternatives are presented in the graph, fig.15

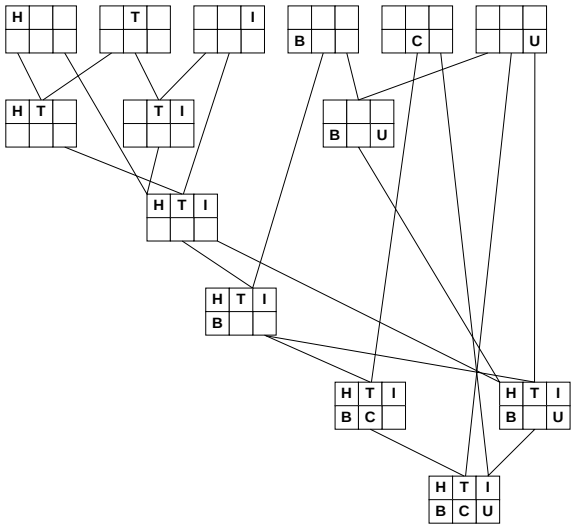


Fig.15 *Graphical representation of all pen assembly sequences*

When all possible variations of the assembly process have been generated they should be evaluated with regard to reasonable criteria. Then the best order of assembling can be chosen.
The research is in progress. The next step will be concerned with establishing the criteria and setting the algorithm for finding feasible assembling sequences of the high value.
Although not completed yet the authors believe that the presented study can be useful for the designers who tackle with design for manufacture and assembly.

4. GENERAL SUMMARY

In the paper the application of dependence matrices in solving selected problems of new product realization has been presented. In the first chapter, the task of algorithmization and rationalization of engineering calculations have been discussed. The way of the conceptual approach to this problem has been presented.
In the next chapter, the possibility of implementation of the dependence matrix to the complex realization problems has been presented. If the whole problem can be decomposed to the set of sub-tasks linked with informational couplings it can be represented as a matrix. Then by re-ordering rows and columns of the matrix while retaining the links it is possible to optimize the industrial realization processes for duration and cost.
In the third chapter it was shown how the matrix form could be used in designing of assembly process. An approach to generation of all the assembly sequences has been presented.
Of course, application of dependence matrices is not limited to those presented in the paper. The authors' intention was just to demonstrate a few selected examples of implementation in various stages of the product development process. According to the

authors' experience implementation of the dependence matrix in engineering design has proved its effectiveness.
Research on development and application of the dependence matrix in engineering problems is in progress.

Acknowledgements: The authors wish to express their sincere thanks to professor R. Rohatyński for his initiative to undertake this direction of research and for his invaluable advice in carrying out the research.

References

[1] A.C. Sanderson, L. S. Homem de Mello, H.Zhang, *Assembly Sequence Planning*, AI Magazine, 1990, pp.62-81

[2] Babirecki W., Kielec R., Sąsiadek M., Organizacja procesów obliczeniowych z zastosowaniem metody macierzowej, Wrocław 2003

[3] Babirecki W., Racjonalizacja algorytmów obliczeniowych elementów maszyn, Opole 2001

[4] Babirecki W., Rohatyński R., Zagadnienie algorytmizacji obliczeń elementów i zespołów maszyn**,** Warszawa, 2003

[5] D.E.Whitney, R. Mantripragada, J.D. Adams, S.J. Rhee, *Designing Assemblies*, Research in Engineering Design, Vol.11, 1999, pp.229-253

[6] D.F. Baldwin, T.E. Abell , Man-Cheung Max Lui, T. L. De Fazio, D. E. Whitney, *An Integrated Computer Aid for Generating and Evaluating Assembly Sequences for Mechanical products,* IEEE Transactions of Robotics and Automation, Vol.7, no.1, 1991, pp.78-94

[7] E. E. Adam, R. J. Ebert, *Produktion and operations management*, Prentice Hal, New Jersey, 1992

[8] Hillier F. S., Lieberman G. J., *Introduction to Operations Research*, McGraw-Hill Higher Education, New York, 2001

[9] J.Żurek, O.Ciszak, Modelowanie oraz symulacja kolejności montażu części i zespołów maszyn za pomocą teorii grafów, Wyd. Politechniki Poznańskiej, Poznań 1999

[10] Kusiak A. *Concurrent Engineering*, New York, 1992

[11] Kusiak A., Aang J., *Decomposition of the Design Process*, Journal of Mechanical Design, 1993 L. S. Homem de Mello, A.C. Sanderson, *Automatic Generation of Mechanical Assembly Sequences*, CMU-RI-TR-88-19, 1988

[13] Larson N., Decomposition and Representation Methods in Mechanical Design, Journal of Mechanical Design, 1995

[14] P.Łebkowski, Metody komputerowego wspomagania montażu mechanicznego w elastycznych systemach produkcyjnych, Wyd. AGH, Kraków 2000

[15] R.B. Gottipolu, K. Ghosh, A simplified and efficient representation for evaluation and selection of assembly sequences, Computers in Industry, Vol.50, 2003, pp.251-264

[16] Rogers James.L., *Reducing Design Cycle Time and Cost Through Process Resequencing*, International Conf. On Engineering Design, ICED 97, Tampere 1997

[17] Rohatyński R., Kielec R. Sąsiadek M., *Implementation of matrix method and evolution algorithm for reorganization of design processes*, Engineering Design in integrated Product Development - EDIPROD 2002: third international seminar and workshop. Zielona Góra - Łagów, Polska, 2002.- Zielona Góra: Redakcja WNT UZ, 2002

[18] Rohatyński R., Kielec R., *Artificial Evolution in Design Process Optimization*, Computer Integrated Manufacturing, International Conference on Zakopane, CIM-2001

[19] Rohatyński R., Kielec R., *Nowa metoda organizacji procesu projektowania systemów technicznych*, Ogólnopolska Konferencja nt. Sztuczna inteligencja CIR-13'98 (cybernetyka-inteligencja-rozwój). Siedlce-Warszawa 1998

[20] Rohatyński R., Kielec R., *Racjonalizacja komputerowego wspomagania projektowania współbieżnego – metoda i przykład*, VI Międzynarodowa Konferencja Naukowa, Computer Aided Engineering, Polanica Zdrój 2002

[21] Rohatyński R., Kielec R., *Sztuczna ewolucja w zastosowaniu do optymalizacji procesu projektowania*, Materiały IV Ogólnopolskiej Konferencji nt. Sztuczna Inteligencja, Szl-15'00, Siedlce-Warszawa, 2000

[22] Steward D. V., The Design Structure Systems: A Method for Managing the Design of Complex Systems, IEEE Transactions on Engineering Management. Aug. 1981

[23] T. L. De Fazio, D. E. Whitney, *Simplified Generation of All Mechanical Assembly Sequences,* IEEE Journal of Robotics and Automation, Vol.RA-3, no. 6, 1987, pp.640-658