project management, constraint programming, alternative projects

Marcin RELICH[*]

CP-BASED DECISION SUPPORT FOR SCHEDULING

Abstract

The paper presents the declarative approach to design of a reference model aimed at project prototyping. The reference model contains the finite set of decision variables, their domains and linking those constraints, i.e. can be seen as a kind of Constraint Satisfaction Problem. Consequently, the model considered can be treated as a knowledge base specifying both a class of enterprises and the projects that could be conducted on their base. So, the model provides a platform for rapid prototyping of alternative versions of project scheduling. The routine queries can be formulated in the straight or reverse way. In that context, the proposed reference model can be implemented in constraint programming (CP) techniques.

1. INTRODUCTION

In the activity of present organizations more and more importance concerns unique activities, so-called projects. A project is a sequence of unique, complex, and connected activities having one goal or purpose and that must be completed by a specific time, within budget, and according to specification [9]. On account of this, the demand arises for new knowledge that enables the problems occurring in the realisation of unique projects to be solved. In this case, of particular significance is knowledge of project management that identifies factors which have an influence on the success or failure of the project, and that uses special methods and techniques.

Many cases of projects indicate that fewer than half of projects met cost and schedule targets [6, 11, 13, 16, 19]. The findings of various other authors indicate that projects which overrun are more common than projects which complete within original time scales, overruns likely to be between 40% and 200% [12]; for instance, only one third of World Bank projects met their aims, with typical delays of 50%. Another survey showing only 17% of projects meeting all three aspects of the project triangle (cost, time, and scope), with typical cost overruns as high as 189% [7]. In the case of software projects, the surveys on estimation performance report that 60-80 percent of all software projects encounter effort overruns [8, 10, 17].

Project success or failure depends on many critical factors, such as factors related to the project, availability of resources, project management, and the external environment [2, 13]. The reasons for project failure can be generally considered in availability of resources (e.g. human, financial, raw materials) and changeability of the external environment.

[*] Ph.D., Faculty of Economics and Management, University of Zielona Góra, m.relich@wez.uz.zgora.pl

Moreover, unstable requirements, lack of well-defined scope, quality of management, and skill of the employees can cause project failure. Another factor is that an enterprise which carries out a few projects can change the priority of the project.

The project requires planning that supports, among other things, the estimates of effort, resources, time, etc., which are fundamental to guide the project activities. To reduce project overruns, there are two ways to approach the problem. The first way is to increase the accuracy of the estimates through a better estimation process and the second, to increase the project control.

It is unrealistic to expect very accurate estimates of project effort because of the inherent uncertainty in development projects, and the complex and dynamic interaction of factors that influence its development. However, even small improvements will be valuable, especially if a project is connected with the large scale. More accurate forecasting supports the project managers in planning and monitoring the project, for instance in the project price set, resource allocation or schedule arrangement.

In the case of a significant difference between actual and planed project parameters, the manager should take a decision concerning the response to the change. The response can regard a support status quo, a correction of differences, a change of the norms, and also it may be connected with continuing the actual project. This approach is usually considered in the research works. The change of project scope can be another type of reaction regarding the performed variations. In this case, it seems important to build the approach that will generate a set of alternative projects and support the decision-maker. The alternative project is considered as a modification of the primary project, that can be made in different stages of the project life cycle, e.g. by the definition of the project or its implementation.

Rapidly changing expectations related to supporting strategic decisions, as well as aiming to reduce cost and investment risk, result in the need to make a task-oriented decision support system. Most of the publications have considered separately the fields of enterprise and project management. This results in a separate knowledge base respectively for an enterprise and project management. Consequently, it implies the difficulty of implementation of these fields within a single tool that is used for decision support. Hence, there is a need to build a single model that combines the fields of enterprise and project management, and that provides a base for making a task-oriented decision support system.

The paper is organized as follows. A reference model concerning an enterprise and project is presented in section 2. Scheduling in a form of the so-called constraint satisfaction problem is described in section 3. An example of the approach, which presents a possibility of decision problem specification in the straight and in the reverse way, is illustrated in section 4. Finally conclusions and future research are presented in section 5.

2. REFERENCE MODEL

The proposed approach combines the fields of an enterprise and project in single platform – the reference model. This type of approach seems to be natural in the case of an enterprise that executes projects and solves standard decision-making problems. In this way, a knowledge base is created that in addition to the inference strategies allows more efficient implementation of decision support system.

It is assumed that the reference model has the structure of constraints satisfaction problem (*CSP*), and it may be described as follows:

$$CSP = ((V, D), C) \tag{1}$$

where:

$V = \{v_1, v_2, ..., v_n\}$ – finite set of n variables,

$D = \{D_1, D_2, ..., D_n\}$ – finite and discrete domains D of variables, where $D_i = \{d_{i1}, d_{i2}, ..., d_{ir}\}$,

$C = \{c_1, c_2, ..., c_m\}$ – finite set of m constraints binding decision variables.

Each constraint treated as a predicate can be seen as an n-ary relation defined by a Cartesian product $D_1 \times D_2 \times ... \times D_n$. The solution to the *CSP* is a vector ($d_{1i}, d_{2k}, ..., d_{nj}$) such that the entry assignments satisfy all the constraints C. So, the task is to find the values of variables satisfying all the constraints, i.e., a feasible valuation. Generally, the constraints can be expressed by arbitrary analytical and/or logical formulas as well as bind variables with different non-numerical events.

Thus, a constraint can be treated as a logical relation among several variables, each one taking a value in a given (usually discrete) domain. To solve such a problem stated by the set of requirements (constraints) that specify a problem at hand, the concept of constraint programming (*CP*) is employed. *CP* is an emergent software technology for declarative description *CSP* and can be considered as a pertinent framework for development of decision support system software aims. The main idea behind the *CP* concept is based on subsequent phases of constraint propagation and variable distribution [14].

Construction of the reference model requires certain assumptions concerning the structure of the modelled object and the tasks performed in it. It is assumed that the client orders may be taken and commenced at any time (possibly adding the new projects to a set of projects already in progress). The expenses regarding an order are paid from the enterprise's own means or from a bank loan. The budget of the project is set with cash flow budget in the investment period. The client order is chosen by the profitability analysis and technical realizability. The enterprise receives the order specification with the client requirements, regarding among others the scope, price and time completion of project.

The enterprise model can be described by its resources. The project model is created from the requirements of the client. In the model, some parameters are determined, among which a set of constraints and decision variables may be distinguished (Fig. 1). The constraints connect the variables that describe the capacity of the enterprise, as well as the variables that concern the conditions of project completion. For instance, the number of the enterprise's employees limits the duration of the project.

It means that fulfilment of specified constraints enables project completion according to client requirements. The enterprise and project model containing examples of decision variables and constraints is shown in Fig. 1.

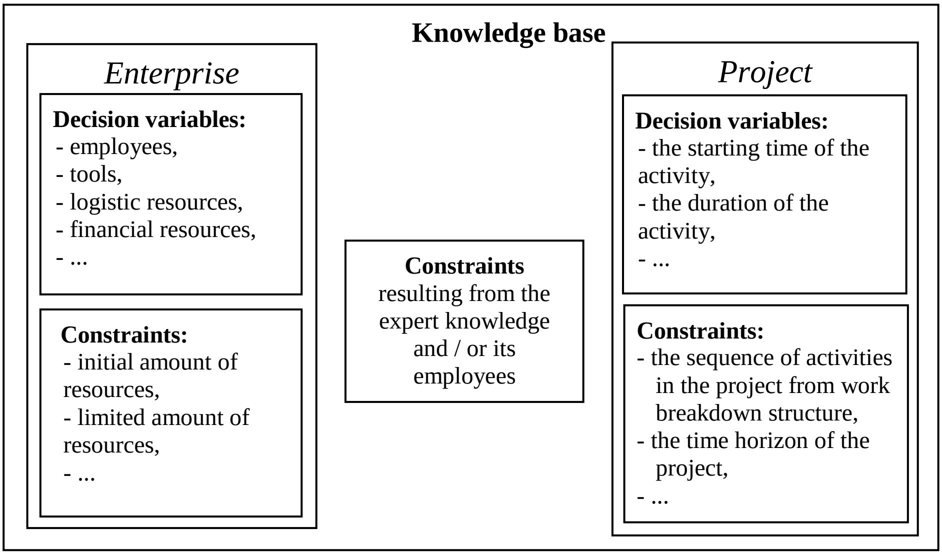
Knowledge base
Enterprise
Decision variables:
- employees,
- tools,
- logistic resources,
- financial resources,
- ...
Constraints:
- initial amount of resources,
- limited amount of resources,
- ...
Constraints resulting from the expert knowledge and / or its employees
Project
Decision variables:
- the starting time of the activity,
- the duration of the activity,
- ...
Constraints:
- the sequence of activities in the project from work breakdown structure,
- the time horizon of the project,
- ...

Fig. 1. Enterprise-project model as a common knowledge base

The assumed model enables descriptive approach to the problem statement, encompasses constraint satisfaction problem structure and then allows implementation of the problem considered in the constraint programming environment. The idea behind the proposed approach assumes the system considered can be represented in terms of a knowledge base (*KB*). *KB* comprises of facts and rules determining the system's properties and relations linking them respectively. Taking into account the concept of constraints propagation and variables distribution following from the constraint programming languages it is easy to note that any *KB* can be represented in a standard form of the *CSP* [18].

KB can be specified in terms of a system [5]. At the input of the system are the variables regarding the fundamental attributes of the object that are known and given by the user. In the considered *KB* for the enterprise-project model, there are, for example, variables concerning the amount of an enterprise's resources or the project structure. The output of the system is described by the attributes of the object that are unknown or are only partially known. In the considered case, there can be included variables regarding e.g. the cost or time of activity, use of resources or the level of investment performance indicators.

Classification of the decision variables in *KB* as input-output variables is arbitrarily made and allows the formulation of two classes of standard queries, in a straight and in a reverse way, as follows [1, 4]:

- a straight way (i.e. corresponding to the question: what results from premises?), e.g. Does a given resources allocation guarantee the schedule does not exceed the given deadline?

- a reverse way (i.e. corresponding to the question: what implies conclusion?), e.g. What activity duration times and resources amount guarantee the given schedule does not exceed the deadline?

The above-mentioned categories encompass the different reasoning perspectives, i.e. forward and backward reasoning. The corresponding queries can be stated in the same model that can be treated as composition of variables and constraints, i.e. assumed sets of variables and constraints limiting their values. In that context, the problem statement of scheduling, which is specified in terms of CSP, is presented in next section.

3. CONSTRAINTS SATISFACTION PROBLEM FOR SCHEDULING

Given amount z of discrete resources r_k specified by (e.g. workforce, tools, money): $R = (r_1, r_2, ..., r_z)$. Given amounts $q_{k,h}$ of available resources at the moment of H: $H = \{0, 1, ..., h\}$. Given a project P_i is specified by the set composed of l activities: $P_i = \{A_{i,1}, A_{i,2}, ..., A_{i,l}\}$. The activity $A_{i,j}$ is specified as follows:

$$A_{i,j} = (s_{i,j}, t_{i,j}, Tp_{i,j}, Tz_{i,j}, Dp_{i,j}) \tag{2}$$

where:

$s_{i,j}$ – the starting time of the activity $A_{i,j}$, i.e., the time counted from the beginning of the time horizon H.

$t_{i,j}$ – the duration of the activity $A_{i,j}$.

$Tp_{i,j} = (tp_{i,j,1}, tp_{i,j,2}, ..., tp_{i,j,z})$ – the sequence of moments the activity $A_{i,j}$ requires new amounts of resources: $tp_{i,j,k}$ – the time counted since the moment $s_{i,j}$ of the $dp_{i,j,k}$ amount of the k-th resource allocation to the activity $A_{i,j}$. That means a resource is allotted to an activity during its execution period: $0 \le tp_{i,j,k} < t_{i,j}$; $k = 1, 2, ..., z$.

$Tz_{i,j} = (tz_{i,j,1}, tz_{i,j,2}, ..., tz_{i,j,z})$ – the sequence of moments the activity $A_{i,j}$ releases the subsequent resources: $tz_{i,j,k}$ – the time counted since the moment $s_{i,j}$ of the $dp_{i,j,k}$ amount of the k-th resource was released by the activity $A_{i,j}$. That is assumed a resource is released by activity during its execution period: $0 < tz_{i,j,k} \le t_{i,j}$; $tp_{i,j,k} < tz_{i,j,k}$; $k = 1, 2, ..., z$.

$Dp_{i,j} = (dp_{i,j,1}, dp_{i,j,2}, ..., dp_{i,j,z})$ – the sequence of the k-th resource amounts $dp_{i,j,k}$ are allocated to the activity $A_{i,j}$: $dp_{i,j,k}$ – the amount of the k-th resource allocation to the activity $A_{i,j}$. That assumes: $0 \le dp_{i,j,k} < q_k$; $k = 1, 2, ..., z$.

The constraints regarding the enterprise include the initial and available amounts of the resources. Moreover, the project portfolio should be completed within the given time horizon $H = \{0, 1, ... , h\}$. It is assumed the activities cannot be suspended during their execution, and also:

- each activity can request any kind and quantity (not exceeding the resource's limited amount) of any resource,
- each resource can be uniquely used by an activity,
- the quantity of resource used by an activity cannot be changed or allotted to other activity,
- an activity can start its execution only if required amounts of resources are available at the moments given by $Tp_{i,j}$.

The following activities order constraints are considered:

- the k-th activity follows the i-th one:

$$s_{i,j} + t_{i,j} \le s_{i,k} \tag{3}$$

- the k-th activity follows other activities:

$$\begin{gathered} s_{i,j} + t_{i,j} \le s_{i,k} \\ s_{i,j+1} + t_{i,j+1} \le s_{i,k} \\ \dots \\ s_{i,j+n} + t_{i,j+n} \le s_{i,k} \end{gathered} \tag{4}$$

– the *k*-th activity is followed by other activities:

$$\begin{gathered} s_{i,k} + t_{i,k} \leq s_{i,j} \\ s_{i,k} + t_{i,k} \leq s_{i,j+1} \\ \ldots \\ s_{i,k} + t_{i,k} \leq s_{i,j+n} \end{gathered} \tag{5}$$

According to (1) the reference model for scheduling can be described as follows:
A set of decision variables *V*:
- the starting time of the activity $s_{i,j}$
- the duration of the activity $t_{i,j}$
- resources z, $Tp_{i,j}$, $Tz_{i,j}$, $Dp_{i,j}$

$$V = (s_{i,j}, t_{i,j}, z, Tp_{i,j}, Tz_{i,j}, Dp_{i,j}) \tag{6}$$

The values sets of variables *V* is specified by the set of domains:

$$D = (D_{si}, D_{ti}, D_z, D_{Tpi}, D_{Tzi}, D_{Dpi}) \tag{7}$$

Note that for the known values of decision variables (e.g. for a variable concerning available amounts of *z* resources), the domain is a set with single element.

A set of constraints *C* includes the constraints regarding an enterprise and a project, for instance, the constraints concerning the sequence of activities, the cost or available amounts of the resources. Some of the constraints link the field of enterprise with project, e.g. the number of available employees.

$C = \{C_1, C_2\}$, where:

C_1: $H = \{1, ..., h\}$ – the constraint of the project horizon *H*,

C_2: $s_{i,j} + t_{i,j} \leq s_{i,k}$ - the constraint of the activities sequence in the project.

An answer to the following question is sought: does a given resources allocation guarantees the project completion by assumed constraints, and if so, what are its parameters?

This question can be expanded to the next, for instance, does a given resources allocation not exceed the given deadline *H* and the given financial resources *q* in time unit *h*? It allows a class of multicriteria problems to be taken into consideration.

The examples regarding the above-described problem are presented in next section.

4. ILLUSTRATIVE EXAMPLES

The example aims to illustrate a possibility of *CSP* specification for decision problem of project planning in the straight and in the reverse way. It assumes, the activities compete with the access to the discrete resources. In the example, single project with nine activities $P = \{A_1, ..., A_9\}$ is considered that network is presented in Fig. 2. Bold lines represent the critical path.

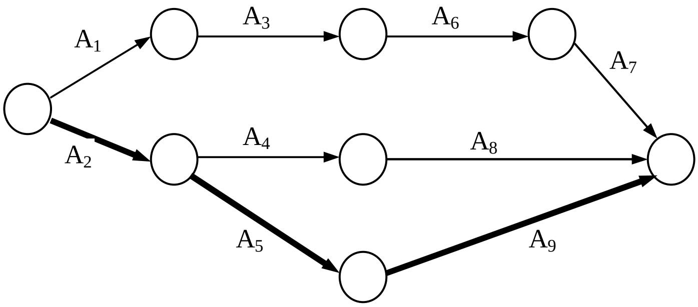

Fig. 2. Activity network of project

4.1. Routine queries formulated in the straight way

Example 1

Operation times for the project by the following sequence are determined: $T = (3, 4, 2, 2, 3, 3, 1, 4, 5)$. Moreover, given the time horizon $H = \{0, 1, ..., 15\}$, and resource r that is limited by 26 units. Number of resource is constant in whole time horizon H. It assumes, an amount of resource is allocated to an activity at the moment of its beginning and can be released only by this activity at the moment of its completion. The required number of resource from the database of past projects, which belong to the same class as considered project, is determined. The resource according to linear function is calculated as follows: $dp_j = 2 + 2 \cdot t_j$. Thus, the sequence of the resource amounts allocated to the activity j is following: $Dp = (8, 10, 6, 6, 8, 8, 4, 10, 12)$.

The order constraints according to the activity network of the project and formulas (3), (4), and (5) are following:

C_1: $s_3 \geq s_1 + t_1$, C_2: $s_4 \geq s_2 + t_2$, C_3: $s_5 \geq s_2 + t_2$, C_4: $s_6 \geq s_3 + t_3$, C_5: $s_7 \geq s_6 + t_6$, C_6: $s_8 \geq s_4 + t_4$, C_7: $s_9 \geq s_5 + t_5$.

The considered problem belongs to the class of “straight” ones where for a given parameters describing the enterprise-project system the activities schedule is sought. It reduces to the following question: is there, and if so, what form does a schedule have that completion time does not exceed the deadline H, and that fulfils the resource constraints? Note the answer to above-mentioned question is connected with determination of the starting time of the activity s_j, where $0 \leq s_j < 15$; $j = 1, 2, ..., 9$.

The obtained solution follows from model implementation in the *CSP*-based reference model and programmed in Oz Mozart. The first admissible solution has the following form: $S = (0, 0, 3, 4, 4, 5, 8, 6, 8)$. The project schedule fulfilled all constraints imposed by an enterprise capability and project requirements, is presented in Fig. 3.

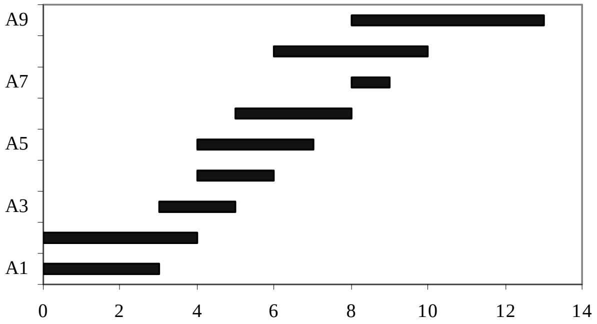

Fig. 3. Gantt's chart of project

The level of resource usage containing the assumed resource's limit in the time horizon is illustrated in Fig. 4.

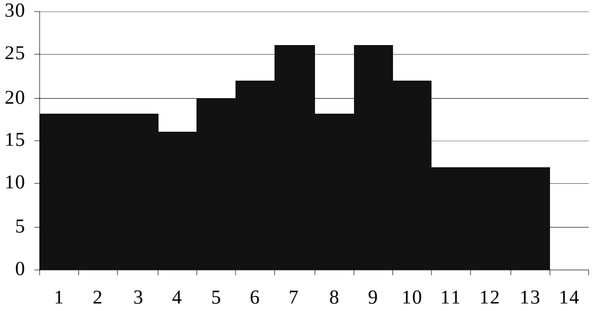

Fig. 4. Gantt's-like chart of the resource usage

Example 2

Given the project *P* specified by the same activity network, time horizon, durations of the activities, and amount of the resource allocated to the activity as in Example 1. However, the new limit of resource ($r \leq 20$) is considered.

The considered problem also belongs to the class of "straight" ones, and it can be reduced to the following question: is there, and if so, what form does a schedule have that completion time does not exceed the deadline *H*, and that fulfils the resource constraints?

Similarly to the previous case, the solution results in a determination of the beginning moments of the activities s_j, however regards smaller amount of the resource. By this constraint, the set of admissible solutions is empty. This means there is no schedule. Thus, there is a possibility to reformulate the considered problem by stating it in a reverse way, i.e. the way aims to search for decision variables (e.g. amount of resource for the activity) guaranteeing that the completion time of the project does not exceed the assumed deadline *H*. This way is considered in next subsection.

4.2 Routine queries formulated in the reverse way

Given the project P specified by the same activity network, time horizon, durations of the activities and limit of the resource ($r \leq 20$) as in Example 2. Amounts of the resource allocated to the activities are not known, however the constraint determining the amounts is given. According to the database of past project, the relationship between an amount of the resource and a duration of the j-th activity has been determined as follows: $dp_j = a + b\cdot t_j$, where $a = \{1, 2\}$ and $b = \{2, 3\}$.

Taking into account above-mentioned assumptions, the problem reduces to the question: what amounts of the resource allocated to the activities dp_j guarantee that completion time of a schedule does not exceed the deadline H, and resource limit r?

In order to response to this question the values of the following sentences are sought: $Dp = (dp_1, \ldots, dp_9)$ and $S = (s_1, \ldots, s_9)$. The reference model encompassing assumption of the considered example was implemented in Oz Mozart programming environment, and the obtained solution is following: $Dp = (7, 9, 5, 5, 7, 7, 3, 9, 11)$ and $S = (0, 0, 3, 4, 4, 5, 8, 7, 9)$. The project schedule fulfilled all constraints is presented in Fig. 5.

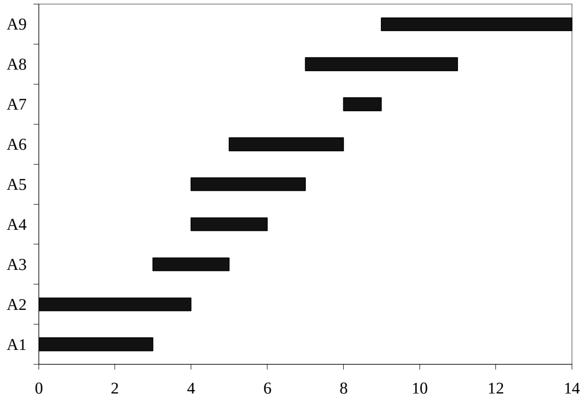

Fig. 5. Gantt’s chart of project

The chart illustrating the changes of resource usages, by assumed resource’s limit and the time horizon, is presented in Fig. 6.

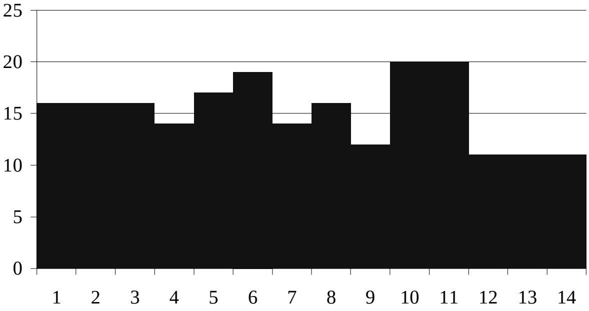

Fig. 6. Gantt’s-like chart of the resource usage

The assumed ranges of decision variables and constraints determine the possible values of sought parameters. The result is a set of feasible solutions in time unit *h*. This set can be empty, or with one or many solutions. Note that the number of generated solutions depends not only on the knowledge base, but also on a user-declared granularity of solutions in constraint programming languages such as, for instance, ILOG or Oz Mozart [15].

5. CONCLUSIONS

In the present, changeable business environment, quickness of response to customer needs or pressure on innovation and effective cost management determine the success or failure in the struggle for market position. This forces more frequent and larger-scale changes in contemporary organizations. The answer to these new challenges is the application of the principles of project management. In the case of projects carried out on a client order, erroneous estimation of expenditures and project deadlines may result penalties being accrued, as agreed upon in the contract or covering the costs with the company's own money. A wrong decision may worsen the liquidity of an enterprise or even lead to its bankruptcy. In this situation, it seems extremely important to support the decision maker.

The proposed approach assumes a kind of reference model encompassing open structure, enabling one to take into account different sorts of variables and constraints as well as to formulate straight and reverse kinds of project planning problems.

Since a constraint can be treated as a logical relation among several variables, each one taking a value in a given (usually discrete) domain, the idea of *CP* is to solve problems by stating the requirements (constraints) that specify a problem at hand, and then finding a solution satisfying all the constraints. Because of its declarative nature, it is particularly useful for applications where it is enough to state *what* has to be solved instead of *how* to solve it [1].

The advantages of the proposed approach include the possibility of the description of enterprise and project management in terms of a knowledge base. Moreover, in the presented approach it is possible to obtain a set of feasible solutions in the different phases of the project life cycle. This is especially attractive in the absence of the possibility of continuing the project in its primary form and can support the decision maker in choosing the alternative project.

Further research focuses on the presentation of the model reference for the project management problem in a dynamic form, taking into account the subsequent project management functionality and assessing their impact on the set of feasible solutions. It should also define criteria for evaluating project alternatives, and carrying out verification of the knowledge base of described object.

REFERENCES

1. BANASZAK Z., BOCEWICZ G., BACH I.: *CP-driven production process planning in multiproject environment,* Decision Making in Manufacturing and Services, pp. 5-32, Vol. 2, No.1-2, 2008.
2. BELASSI W., TUKEL O.I.: *A new framework for determining critical success/failure factors in projects,* International Journal of Project Management, pp. 141-151, Vol. 14, 1996.
3. BERTALANFFY L.: *General system theory - A Critical Review,* General Systems, pp. 1-20, Vol. 7, 1962.

4. BOCEWICZ G., BACH-DĄBROWSKA I., BANASZAK Z.: Declarative approach to computer aided project planning systems design. Akademicka Oficyna Wydawnicza EXIT, Warsaw 2009. (In Polish).
5. BUBNICKI Z.: *Logic-algebraic method for knowledge-based relation systems,* Systems Analysis Modelling and Simulation, pp. 21-35, Vol. 33, 1998.
6. COOKE-DAVIES T.J., ARZYMANOW A.: *The maturity of project management in different industries: an investigation into variations between project management models,* International Journal of Project Management, pp. 471-478, Vol. 21, 2003.
7. JORGENSEN M., SJOBERG D.I.: *The impact of customer expectation on software development effort estimates,* International Journal of Project Management, pp. 317-325, Vol. 22, 2004.
8. JORGENSEN M., GRUSCHKE T.: *The impact of lessons-learned sessions on effort estimation and uncertainty assessments,* IEEE Transactions on Software Engineering, pp. 368-383, Vol. 35, No. 3, 2009.
9. Meredith J.R., Mantel S.J.: *Project Management – a managerial approach*. Third Edition, John Wiley and Sons, New York 1995.
10. MOLOKKEN-OSTVOLD K., JORGENSEN M.: *A comparison of software project overruns,* IEEE Transactions on Software Engineering, pp. 754-766, Vol. 31, No. 9, 2005.
11. NITITHAMYONG P., SKIBNIEWSKI M.J.: *Success/failure factors and performance measures of web-based construction project management systems: professionals' viewpoint,* Journal of Construction Engineering and Management, pp. 80-87, Vol. 132, 2006.
12. REICHELT K., LYNEIS J.: *The dynamics of project performance: benchmarking the drivers of cost and schedule overrun,* European Management Journal, pp. 135-150, Vol. 17, 1999.
13. ROBERTSON S., WILLIAMS T.: *Understanding project failure: using cognitive mapping in an insurance project,* Project Management Journal, pp. 55-71, Vol. 37, 2006.
14. ROSSI F.: *Constraint (logic) programming: A survey on research and applications,* New Trends in Constraints (Apt K.R. et al.), pp. 40-74. Springer-Verlag, Berlin 2000.
15. SCHUTLE H., SMOLKA G., WURTZ J.: *Finite Domain Constraint Programming in Oz.* German Research Center for Artificial Intelligence, Saarbrucken 1998.
16. SHORE B.: *Systematic biases and culture in project failures,* Project Management Journal, pp. 5-16, Vol. 39, 2008.
17. SINGH R., KEIL M., KASI V.: *Identifying and overcoming the challenges of implementing a project management office,* European Journal of Information Systems, pp. 409-427, Vol. 18, 2009.
18. VAN ROY P., HARIDI S.: *Concepts, techniques and models of computer programming*. Helion, Gliwice 2005.
19. YEO K.T.: *Critical failure factors in information system projects,* International Journal of Project Management, pp. 241-246, Vol. 20, 2002.