TRIZ*fest*-2022

ON SOME ASPECTS OF TRIZ FLOW ANALYSIS

Hans-Gert Gräbe*

*Institute for Applied Informatics (InfAI), Leipzig, Germany

Abstract

In this paper, a number of concepts on which Lyubomirsky, Logvinov and Lebedyev [1, 2, 3] base their methodology of a flow analysis are critically analysed. It is suggested that the distinction between system models of the first and second kind in the sense of Shchedrovitsky should be taken more into account in TRIZ flow analysis. Such a distinction between *structural* and *processual* organisation is an integral part also of organisational informatics. The similarity of functional and flow analysis, observed by Lebedyev and Logvinov, is related to the restriction of their considerations to a system model of the first kind. In the transition to a system model of the second kind, the functional properties of the components appear as *bundles of functions*. These functions as a *potential* relationship between tool and workpiece (object) find their real-world application in the interaction with the three flows of energy, substance and information. The clear separation of the terms *component* as a stateless functional unit and the flow of *objects* as state-carrying units of instantiation according to [4] is suitable for further terminological clarity here. It is proposed to qualify the previous considerations on flow analysis as *Flow Functional Analysis* and to consider it as part of a more complex field of *Flow Analysis*, which in turn is a part of *Process Analysis* within a system model of the second kind.

Keywords: *flow analysis, functional analysis, component and object, systems of first and second kind, development laws of technical systems, state changes, control flows*

1 Functional Analysis and Flow Analysis. An Inventory

TRIZ flow analysis has not yet been systematically elaborated. [1, 2, 5] see strong parallels between the well-developed TRIZ concepts of functional analysis [6: ch. 4.4] and flow analysis. However, this may be related with the specific view on flow analysis built on Lybomirsky's paper [1]. A closer look at the 60 examples briefly touched upon in that paper to illustrate different patterns of flow development shows the close proximity to a *functional* development of the technical system under consideration. Souchkov argues similarly in the four pages devoted to process analysis in [6: ch. 4.5] (the concept "flow analysis" is not touched in that book). In [1] is furthermore shown that flow analysis has historically arisen from the development law of *minimum energy throughput*, which Altshuller [7] formulated as his second law and counted it among the "static" laws, although a highly dynamic process is involved here. Altshuller formulates in detail "A necessary condition for the viability of a technical system is the flow of energy through all its parts." [7: p. 73].

He thus implicitly postulates a technical system to be an *Open System* that only builds up its internal structure under (in modern terms) the required throughput of energy, substance and information. The stability of these external throughput conditions is, as context, the condition for the stability of the system's performance, concerning its MPV. This also applies beyond a

"minimal" energy throughput – minimality, which is essential and discussed as a matter of course in [1], does nevertheless *not* occur in the original [7: p. 73].

2 Systems of First and Second Kind

At this point, one of the fundamental contradictions of any systemic approach is already present – the decomposition of the indecomposable. For a functional analysis, the decomposition of the system into its components is indispensable. However, the system can only be operated in its assembled state. V. Petrov [8: ex. 1.6] emphasises this with the example of an aeroplane – no part alone can fly, not even the sum of all parts. Only the assembled aeroplane can fly.

Shchedrovitsky devotes a large part of his explanations in [9] to precisely this contradiction, which every engineer understands practically quite well but can rarely put it into words in a reasonably comprehensive way. Shchedrovitsky distinguishes two kinds of system models, which he develops one after the other and that build on each other.

The system concept of the first kind [9: p. 89 ff.] examines the structural design of the system, i.e. its decomposability. Shchedrovitsky emphasises (p. 98) that this is a centuries-old concept and that even such an understanding of systems dominated until well into the 1960s. The system concept of a structural *reduction to essentials* developed in more detail in [10] also follows this approach, but already emphasises its self-similarity.

On such a self-similarity also the TRIZ system operator is based, which addresses the co-evolution of systems. In [11] a major misconstruction of that system operator in its classical reading was already criticised in more detail – the terms supersystem and subsystem suggest an immersive relationship between the rows of the 9-field schema and thus make it difficult to think of subsystems and supersystems in plural as well as to appreciate the specific reduction achievements at each of the modelling levels of system and subsystem (aka component). However, such specifics are an essential part of modern composition concepts, for example in computer science, where API programming and the distinction between specification and implementation are among the fundamental complexity-reducing concepts.

In the system concept of second kind [9: p. 89 ff.], Shchedrovitsky discusses ways composing systems of the first kind into more complex systems and thus forms of movement of that self-similarity of the system concept of the first kind towards new systems and thus towards increasingly complex structures of an "artificial world" in which material structurings are charged with forms of description and in this way become creatively accessible to the multi-optionality of cooperative human action, including thought action, at least in a limited way. This is not only about *system improvement*, but also about *system genesis*, if one wants to name this way the transition from a "chaotic" [12] to a more structured perception of "natural" processes. The development pattern *Mono-Bi-Poly-System* [13], that pursues precisely such a genesis process, can also be classified here.

Shchedrovitsky describes the starting point of such a self-similar system genesis as follows (ibid, p. 89, my emphasis).

> A complex object is represented as a system in the first place, when we have distinguished it from its surroundings by either completely breaking all of its connections or by preserving them in the form of *functional properties*; in the second place, when we have divided it into parts (mechanically or according to its inner structure) and thus obtained a *totality of parts*; in the third place, we have *connected the parts* and turned them into elements; in the fourth place, when we have *organised the connections into a unified structure*; and when, in the fifth place, we

> have *put this structure back in its previous place*, thus delineating this system as a unity.

What next? Shchedrovitsky continues

> What is insufficient in this first concept of a system? To ask what is insufficient about it is not to say that we cannot work with it. On the contrary, we cannot work without it. But that is not enough. It is only the first moment in systems analysis. The point is that there are only *connections* and *no processes* here. It is easy for us to say [...] that there are processes standing behind the connections. But there are no processes here in patent form. *This systemic-structural approach does not capture process as such*. The second defect is that systems always turn out to be subsystems. There are no criteria for pinpointing the unity of a system.

It are precisely these questions of a complex descriptive structure that is invoked in flow analysis. The word "potok", which is used in a largely uniform way in Russian texts, appears in English translations in several forms as flow, flux, stream ... and thus points to a semantic overloading of the terminology that hardly played a role in the analyses in [1, 2, 3, 5] of the development of the concept of a flow analysis. This is also due to the fact that the system concept of TRIZ essentially remains on the level of a system concept of the first kind.

Shchedrovitsky continues

> When this point was understood, the second concept of a system was born. It takes on the first concept in full, but refers it to the structural plane. From the viewpoint of the second concept, to represent something as a simple system is to describe it in four planes as follows:
>
> 1. process
> 2. functional structure
> 3. organisation of material
> 4. material alone.
>
> In other words, if we have an object, then to represent this object as a simple system or monosystem is to describe this object
>
> - first as a process,
> - then as a functional structure,
> - then as an organisation of material,
> - and in fourth place as mere material.
>
> And these four descriptions should refer to the same object and be interconnected.

These two concepts of a system correspond to the levels of *structural and processual organisation*, which are widely used in organisational informatics to describe the functioning of organisational structures with e.g. the means of Business Process Modelling, cf. for example the organisational manual of a German ministry [14: sect. 1.1.1].

3 Technical systems and Bundles of Technical Functions

In such a systemic context of second kind the (abstract) concept of a minimal technical system *tool – processes – object* as functionality of a technical *component* is many times instantiated in a heterogeneous bundle of *technical functions*, which – in the simplest case – transform the *object* (workpiece) in a well-defined sequence of operations several times until it becomes the *useful product*. For example, the bodywork department of a car manufacturer consists (structurally) of two sub-departments (components) – the press subdepartment and the paint subde-

partment – which are involved with different functions in the transformation of the raw material into the bodywork as useful product.

The flow of workpieces through this chain of functions leads to a sequence of targeted state changes of the workpiece by – in the best case – memoryless technical components (TC) that provide the function required for the state change in each case. For this to happen, however, the workpiece must fit a well-defined specification at the input of each TC as well as at its output, because the output of a TC on the path of the flow of workpieces is followed by the input of the next TC.

4 Flow Functions and Flows of Workpieces

This structure is typical for a simple assembly line system, showing that the *flow* of workpieces, as the prototype of any "useful" flow, is *orthogonal* to the functional processing stages and itself encapsulates multiple functions that can be the subject of a process-oriented functional analysis [6: ch. 4.5].

This functional overlay of flows is recorded in [15: table 6 and 7] in a separate column with heading *function type*, which can take the values *transport function*, *correction function* and *production function*. However, this subdivision based on an earlier work of M. Bauer does not seem to be very helpful in that modelling, since all three function types are always present in the depicted parts of the tables. Nevertheless, such a *Flow Function Analysis* based on a clear classification of flow function types seems to be quite useful in order better to understand the difference and interplay between a flow as an active tool and a flow as a simple sequence of passive workpieces.

The transport function seems to be one of the basic functions of flows. Here, however, a distinction must first be made as proposed e.g. in [3] between the carrier and the carried. In [3] Lebedyev tries to unite both under the term *flow*. But is that appropriate on its own? The conveyor belt in an assembly line production as a carrier of the workpieces has clear independent functions as a TC – in addition to the actual transport function, the speed of the conveyor belt determines the work rhythms and also the reaction possibilities in problematic situations – triggering the "red button" stops the conveyor belt and thus also the flow of workpieces in order to concentrate on eliminating the problem (as provided for in the Toyota Production Model, for example).

5 An Example

Such a simple operational mode with a "red button" is not always given, as is obvious from the example of the dishwasher, which is analysed in more detail in [15]. The water flow fulfils its transport function by transporting the "workpiece" dirt from the plates to the collection sieves. In Schaumann's flow analysis, the water flow is thereby modelled as a carrier of three other flows (coarse, fine and micro dirt). We are dealing with a situation similar to that of the conveyor belt: the carrier as flow is meaningless ("parasitic") without the carried material, but on the other hand, as a mixture of carrier and carried material, it may change the properties and thus the behaviour of the flow. Hence the properties of the transport function of the flow do not result solely from those of the carrier. At the same time, we come across a third extremely important set of instruments for the analysis of certain types of flows – the *Material Flow Analysis* (MFA) or *Substance Flow Analysis* (SFA). On the quantitative side mainly tools from mathematics, physics and chemistry are used to describe the spatio-temporal flow behaviour of (largely) homogeneous mixtures of fluid substances. This aspect, which is massively underexposed in TRIZ flow analysis, cannot be discussed further here.

Let us return to [15] and take a closer look at the flows of dirt. We apply another TRIZ tool for their analysis, the *Smart Little People* (SLM) or *Miniature Dwarf's* method, in order to emphasise the workpiece character of the dirt particles more clearly. At the beginning the dwarfs cling to the plates everywhere. The water flow – initially in a productive function – detaches the dwarfs from the plates and then transports them – in its transport function – to the collection point, the filter system. The focus of the analysis and innovation in [15] is precisely on the functioning of this filter system. In this case, however, the operational zone includes not only the filter system itself, but also the feed mechanisms to it, since the dwarfs have different sizes and thus resist to be "captured" by the filter system to different degrees. The subdivision into three flows (streams?) (coarse, fine and micro dirt) therefore has less to do with the subdivision into flows than with different technico-functional properties of the objects (the dwarfs) to be processed with respect to the tool (the filter system) in the cleaning process, which are brought to the collection point several times via the carrier flow of water. When enough dwarfs are "arrested", they are "released" into the wastewater via a functional inversion in the filter system.

Caveat: I share the uncomfortable feeling you may have had about this application of the SLM model. It reveals a fundamental problem with this kind of anthropomorphisation of technical processes in which the object itself is to be taken out of circulation. Further thought should definitely be given to similarities with real processes in the socio-cultural systems of this world.

6 State Changes and Contol Flows

Systems of the second kind are thus characterised by the fact that its components provide a bundle of technical functionalities that, in the best case, changes repeatedly the state of objects that are moving in a complex network of flows between these components. In computer science, such flows play an important role as *control flows*. Already in the concept of the von Neumann computer, elements of *code structuring* can be found, such as the description of repetitive processes in loops or the outsourcing of frequently used code parts in function definitions, to simplify the *structural organisation*. This simplification of the structural organisation in the form of a *system of the first kind* is nevertheless accompanied by significantly more complicated control flow structures and thus by a higher complexity of the description form of the *processual organisation* of such a programme considered as a *system of the second kind*. To master this complexity, hierarchical flow concepts with precisely defined entry and exit points (breakpoints) are used, which simplify the debugging of code, for example. The essential conceptual tool of such a flow analysis is thus the (dual) *structuring* of the process dimension of such a system of the second kind in the form of another system of the first kind or even several such systems, e.g. in modern framework architectures as Spring Boot or Dot-Net etc.

This approach, especially if it is scaled beyond the boundaries of a single programme, requires a considerable amount of standardisation and norming and thus, on a conceptual level, ultimately the transition from verbs to nouns. For the comprehensive reuse of the *processual* solution as a black box, it must itself be brought into the artefactual form of a technical system and thus the process into the form of a *product*. In this product form, and thus descriptively as a noun, the solution can be both integrated into more complex analyses and traded. In computer science, the latter is referred to as COTS – Components off the Shelf. However, the possibility of *structural composition* is limited by the requirement that the composed system is only viable if, at runtime, the process and thus also the control flow unfold again according to the rules. The indecomposability of the system has to be restored.

7 Beyond Object Oriented Programming

The merger of attributes and functionality into such manageable units is pursued in computer science with the approach of object-oriented programming (OOP). Although this merger made it possible to create lightweight objects dynamically and thus to conceive, compose and orchestrate a large variety of information flows from structured objects, it has the disadvantage that such objects can occur both as (functional) *components* and as (stateful) *objects* in the sense of TRIZ. Major terminological confusions in TRIZ about the use of the terms component, object, element, product, etc. have one of their starting points here.

Szyperski therefore suggests in [4] to go beyond OOP and return back to a clear separation of function and state. Such a separation can already be found in the CORBA concept [4: ch. 13], where essential services are provided by *servants* as stateless functional units whose interface specifications are available in machine readable form written in a specific IDL – Interface Definition Language – and stored in specific repositories in the infrastructure. In order to access these services in real information flows, *broker objects* are required to encapsulate state information about the current flow. These broker objects often interact with other objects that are more functional in nature. See, e.g., the example of the licensing service [4: p. 242], in which several information flows intersect – the primary information flow of the service itself, the secondary information flow required to decide whether the service may be licensed under the conditions given in the first information flow, and a tertiary information flow that processes the accounting of the service.

Szyperski's proposal amounts to revoke the merger of function and state in the OOP object concept and to make a clear separation between (functional) *components* that have no externally observable state and *objects* as encapsulation of state. Such a separation is particularly useful in distributed business environments, where the provider of a service changes the state of objects that are in the customer's area of responsibility. In [4] a component is defined by the following three characteristic properties:

- It is a unit of independent deployment.
- It is a unit of third-party composition.
- It has no (externally) observable state.

In [4: p. 36] this concept is clarified:

> In many current approaches, components are heavyweight units with exactly one instance in a system. For example, a database server could be a component. If there is only one database maintained by this class of server, then it is easy to confuse the instance with the concept. An example would be the payroll server of the company. For example, the database server, together with the database, might be seen as a module with an observable state. According to the above definition, this "instance" of the database concept is not a component. Instead the static database server program is, and it supports a single instance – the database "object". In the example, the payroll server program may be a component, while the payroll data is an instance (an object). This separation of the immutable "plan" from the mutable "instances" is essential to avoid massive maintenance problems.

Szyperski continues: "The notion of instantiation, identity, and encapsulation lead to the notion of *object*. In contrast to the properties characterising components, the characteristic properties of an object are:

- It is a unit of instantiation and has a unique identity.
- It may have state and this can be externally observable.
- It encapsulates state and behaviour".

8 Altshuller's Development Laws and Open Systems

Let us return to Lyubomirsky's description [1] of the roots of flow analysis. He states that the starting point was Altshuller's *Law of the minimum energy conductivity* of the system required to supply its components with an energy throughput without which the respective component is not even viable. This is a fundamental constitutive principle of *Open Systems* in general, especially of living and social systems driven by metabolism. A specific internal structuring only gets formed and can be maintained if a defined energetic throughput through the system is guaranteed. The classic example in the (mathematical) Theory of Dynamical Systems are the Bénard convection cells in a heated fluid. According to [1], the understanding of Altshuller's law has evolved from a static view of the requirement of a *minimum* energy throughput to *initiate* the process (of internal structuring) in the first place, to the principle of *optimising* this throughput, which is seen solely in an *increasing conductivity*, i.e. in a *quantitative* optimisation. The structuring effect of the energy throughput on a component depends, however, in most cases also very strongly on *qualitative* parameters of the energy flow. Both the type and composition of the energy and the supply regime play an important role, for example, in the technical exploitation of resonances and dissonances.

This, however, brings a number of other Altshuller's development laws into the focus of consideration – the *Law of Adjusting the Rhythms* (nowadays also called *Trend of Increasing Coordination* [16]), the *Law of Uneven Development of System Components* (caused by the respective specific absorption capacity of energy of a component also in the demarcation and competition with other components) and the *Law of Transition to the Supersystem* (i.e. from the point of view of the components, the shift of control of this energy flow, which is *external* for the component but *internal* for the system, to the system level). We see that the effects of several of Altshuller's laws meet in a reasonably comprehensively understood Flow Analysis.

9 Flows and Transmission

Another connection, which has not been considered much in the previous explanations, is the implicit appearance of flows in the (extended) TRIZ model of a technical system with the components energy source, propulsion, transmission, tool, processed object and control. Indeed, it is about an energy flow with source, transformation into energy useful for the tool (propulsion), flow of energy to this tool (transmission) and transformation of the energy into a state-changing effect on the object (tool). The flow of energy ends here at the tool. Lyubomirsky emphasises in [1] that in the further qualification of the law of minimal energy conductivity, *substance and information flows* are also brought into focus. Of course, these types of flow are also fundamental ingredients to ensure the viability of a technical system.

The same considerations about the interaction of quantitative and qualitative parameters as developed above for the energy flow also apply to the *flow of substance*. However, the target of this flow is not the tool, but the *place* (a central notion in Shchedrovitsky's system concept) that the object occupies as object of transformation. The same applies to the *flow of information*, understood as a stream of data that provides relevant formally structured information as *state of a description* of the state transformation to be performed on the object. The state-changing effect of the tool on the object is thus contingent on these three flows – the *flow of substance* that brings the object (or the tool) into the operative zone, the *flow of information* as the description of the state transformation to be exercised and the *flow of energy* required to carry out this process. Earlier, long before these three flows meet in space and time, the *tool* was put in place as a *pre-structured functional principle* that brings these three factors of production together in operation for the *real-world* change of state. Such a description is close to Szyperski's understanding of component and object.

In the process of operation, which can be encapsulated as a minimal technical system by itself, these *three* flows thus come together. The extension of the term *minimal technical system* in TRIZ to a *complete technical system* (initially without control) adding transmission and energy source [16: p. 40] focuses on flow and functional transformation of energy only. Adding the control to the technical system, the flow and functional transformation of energy is joined with the flow and functional transformation of information. The close interlacing of both flows is expressed in the fact that in the abstract TRIZ model of a complete technical system the control not only affects the tool, but also (potentially) the source, propulsion and transmission. While energy and information flows are considered as *active* agents in the process of transformation of the workpiece by the tool, in the TRIZ concept of that functional transformation the workpiece as an object remains peculiarly *passive*. The corresponding *flow of substance* is limited in its function in that methodological model solely to transport into and out of the operative zone. Of course, such a passivity of the workpiece is by no means appropriate for a description of many technological process, especially in chemical and biological contexts.

10 Flows and SF Modelling

The prominent role of the energy flow compared to the flows of substance and information is also constitutive for generalisations of functional analysis, especially around elements of a substance-field (SF) analysis. At a first glance, such a generalisation seems to have little to do with flow analysis. However, fields mediate effects between two substances, which can often be interpreted as tool and workpiece of a functional model, but due to the more symmetrical structure of an SF model the clear division into active and passive parts that is typical for functional models are avoided. Fields, at least in the classical understanding in which the field concept is not overstretched as in some TRIZ applications, are always introduced descriptively as *potentiality* for action, which generate the energy flows required in the operational mode, for example, via differences of potentials. This is why gradient methods play an important role in Lyubomirsky's patterns of the development of technical systems [1] directly or in a disguised form (for example by shortening the flow length).

11 Conclusion

Smirnov's EFM.K methodology [17, 18] also addresses such a connection and attempts to extend it to flows of substance and information. Due to space restrictions, this cannot be discussed here. However, in the majority of flows created in this way, concepts such as carrier, channel, source and sink which are constitutive for [1, 3] do not play a role here. Flows with these additional properties are already highly technically enclosed flows, which are themselves more or less elaborated technical systems with very specific functional properties. The mathematical methods of MFA or SFA can also be meaningfully applied to such *enclosed flows* only. In the case of the propagation of thermal or acoustic fields in the SF analysis, as well as in the case of the propagation of liquids and gases through diffusion or similar phenomena that are widespread in technical applications, a flow analysis can hardly be carried out in a targeted manner with the conceptual tools developed in [3]. This does not devalue [3] in any way, but raises the question how to frame the object of a flow analysis considered there in an appropriate way.

It is suggested to take the distinction between system models of the first and second kind in the sense of Shchedrovitsky and thus a distinction between *structural* and *processual* organisation more into account. In the transition to a system model of the second kind, the functional properties of the components that appear as *bundles of functions* are only *one* of the ingredients which turn the relationship between tool and workpiece (object) in a real-world ap-

plication to action. Additionally, the interaction with the flows of energy, substance and information is required. The clear separation of the terms *component* as a stateless functional unit and the flow of *objects* as state-carrying units of instantiation according to [4] is suitable for further terminological clarity here. It is proposed to qualify the previous considerations on flow analysis as *Flow Functional Analysis* and to consider it as part of a more complex field of *Flow Analysis*, which in turn is a part of *Process Analysis* within a system model of the second kind.

References

1. Lyubomirsky A. "The Law of Improving the Efficiency of Matter, Energy and Information Flows". *Proceedings of the TRIZ Developer Summit 2016*. (in Russian).
2. Lebedyev Yu., Logvinov S. "Integration of Flow Analysis with Function Analysis". *Proceedings of the TRIZ Developer Summit 2014*.
3. Lebedev Yu. "Improving flow analysis tools". TRIZ Master Dissertation, 2015. https://matriz.org/yu-lebedev/, last accessed March 4, 2022. (in Russian)
4. Szyperski C. "Component Software: Beyond Object-Oriented Programming". New York, ACM Press, 2002.
5. Lebedyev Yu. "Classification of flows in technical systems", 2011. https://www.metodolog.ru/node/967, last accessed March 4, 2022. (in Russian)
6. Koltze K., Souchkov V. (2017). "Systematic Innovation". Hanser, Munich, 2017. (in German).
7. Altshuller G.S. "Creativity as an exact science". Moscow, Sov. Radio, 1979. (in Russian).
8. Petrov V. "Laws and patterns of systems development". Book in 4 volumes, ISBN 978-5-0051-5728-7. (in Russian).
9. Shchedrovitsky G.P. "Selected Works. A Guide to the Methodology of Organisation, Leadership and Management". In: Khristenko, Reus, Zinchenko et al. *Methodological School of Management*. Bloomsbury Publishing, 2014. ISBN 978-1-4729-1029-5.
10. Gräbe H.-G. "Men and their technical systems". *Proceedings of the TRIZ Future Conference 2020*, pp. 399-410.
11. Gräbe H.-G. "Technical Systems and Their Purposes". In: Oliver Mayer (ed.), *TRIZ-Anwendertag 2020*, Springer Nature, 2021, p. 1-13. http://dx.doi.org/10.1007/978-3-662-63073-0_1
12. Mann D. "(Systematic) Innovation in Complex Environments". *Proceedings of the TRIZ Developers Summit 2019*.
13. Souchkov V. "Business Services Innovation: Transition to Supersystem". *Proceedings of the TRIZ Developers Summit 2019*.
14. Organisational handbook of the Federal German Ministry of interior, for construction and home. https://www.orghandbuch.de/, last accessed March 4, 2022. (in German)
15. Schaumann U. "Flow analysis in TRIZ – a different approach to problem solving". In: Oliver Mayer (ed.), *TRIZ-Anwendertag 2020*. Springer Nature 2021, p. 67-79. https://doi.org/10.1007/978-3-662-63073-0_7
16. Lyubomirskiy A., Litvin S. et al. "Trends of Engineering System Evolution". Sulzbach-Rosenberg, 2018.
17. Smirnov E. "Element-Functional Conflict Modelling: EFM.K". In: *Three generations of TRIZ. Proceedings of the annual scientific-practical conference dedicated to the 90th anniversary of G.S. Altshuller*. St. Petersburg 2016. (in Russian).

18. Smirnov E. “The first level algorithm for finding inventive solutions based on function-oriented modeling of conflicts and analysis“. In: *Acta Technica Napocensis, Series: Applied Mathematics, Mechanics, and Engineering*, Vol. 63, Special Issue, October, 2020

Paper category: Research

Corresponding author: Hans-Gert Gräbe, graebe@infai.org