Technical Systems and Their Purposes

Hans-Gert Gräbe, Leipzig

Version of July 20, 2020

The whole is more than the sum of its parts.

1 Introduction

The conceptual foundation of its own theory in the "TRIZ Body of Knowledge" [5] is operated only half-heartedly. Especially the question, what is a *technical system*, is left to everybody's imagination – you know what we're talking about. In this text I try to develop theoretical context for this notion and ask how far it takes, in order to describe the *world of technical systems* in more detail. The not surprising answer is *not very far*, because the whole is also here more than the sum of its parts.

After a short recapitulation of the concepts, developed in [3] in more detail, I show by an example, to what extent also TRIZ benefits from precise notions and what is the impact on modelling for the scope of (abstract) solution models, from which a real-world solution is to be developed.

In the second part, a series of links between the evolution of technical systems and socio-cultural processes are discussed, which in the current TRIZ theory of the "evolution of engineering systems", as [6], are severely underexposed: Cooperation and competition, normalization and standardization and the role of purposes of first and second order.

We show that relational notions in the *world of technical systems* become rather visible in the concept of *technical principles* than of *technical systems*. Hence, the approach in [9] is much better suited to describe the evolution in the world of technical systems as that in [6].

2 Technical Systems – a Conceptual Approach

Operation and use of technical systems is a central element of today's world changing human activities. This requires planned and coordinated activities, because using a system requires its operation. Conversely, it makes little sense to operate a system that is not being used. In computer science this relation between definition and call of a function is well known – calling a function that has not yet been defined causes a runtime error; the definition of a function that is never called points to a design problem.

In computer science this is closely linked to the distinction between design time and runtime. In the real world use of technical systems such a distinction is even more important – during design time the principal cooperative interaction is *planned*, during runtime the *plan is executed*. So for technical systems, their *description form* interpersonally communicated as *founded expectations* and the *enforcement form* resulting in *experienced results* have to be distinguished.

In addition to the description and enforcement dimensions, for technical systems the *aspect of reuse* also plays an important role. This applies, at least on the artifact level, *not* to larger technical systems – these are *unique specimens*, even though assembled using standardized components. Also the majority of computer specialists is concerned with the creation of such unique specimens, because the IT systems that control such a facility are also unique.

The special features of a technical system result therefore mainly from the *interplay of components*, where one also has to distinguish between description form (the modelling) and enforcement form (the delivery, installation and operation of real-world specimens). While in the planning and modelling phase there is still much freedom for changes, the enforcement form is characterized by significantly higher inflexibility. Although the world is more complicated than reflected in a dichotomy like this – who wants to change a plan which has already been approved by the administration – in the following we are working with this conceptual "reduction".

This brings together essential elements for a first approximation to the *concept of a technical system.* In a planning and real-world context this notion is fourfold overloaded

1. as a real-world unique specimen (e.g. as a product, even if it is a service),
2. as a description of this real-world unique specimen (e.g. in the form a special product configuration),

and for components produced in larger quantities also

3. as description of the design of the system template (product design) and
4. as description and operation of the delivery and operating structures of the real-world system producing unique specimens according to this template (as production, quality assurance, delivery, operational and maintenance plans).

Point 4 in particular hardly plays a role in the TRIZ context, although it can be assumed that neither in the private nor in the business environment there is a sustained demand for technical products with foreseeable inadequate service.

In such a context, *technical systems* are systems whose design is influenced by cooperatively acting people. *Existing* technical systems are normatively characterized at description level by a *specification* of their interfaces and at enforcement level by their *guaranteed specification-compliant operation.* This is clearly within the range of the standard TRIZ terminology of a *system of systems* – a technical system consists of components, which in turn are technical systems, whose *functioning* (both in functional and operational sense) is assumed for the currently considered system perspective. In such an understanding, the use of the concept of a technical system is characterized as an epistemic function of (functional) “reduction to the essential”.

In the TRIZ literature such conceptual foundations hardly play a role. Relevant textbooks such as [4] consider the term as given by intuition derived from “industrial practice” [4, p. 2], while other terms as “process”, “product”, “service”, “resources” and “effects” [4, p. 6–10] deserve detailed explanation. How the concept of a *technical system* can be sharpened? In our seminar [2] we identified “the concept of system as descriptional operation to *reduce* real phenomena *to the essential* thus making them feasible for systematic action.” The reduction has the following three dimensions [2, p. 18]:

(1) It provides an external demarcation of the system against an *environment*, reducing these relationships to input-output relationships and guaranteed throughput.

(2) It provides an internal demarcation of the system by combining subareas as *components*, whose functioning is operated based on a “behavioral control” and reduced to input-output relationships.

(3) It provides a reduction of the relationships in the system itself to “causal essential” ones.

In [2] it is further stated that such a reductive descriptional operation rests (explicitly or implicitly) on previously existing descriptive services, in particular

(1) an at least vague idea about the (working) input-output services of the environment,

(2) a clear idea of the processes within the components (beyond the pure specification), and

(3) an at least vague idea about causalities in the system itself, that is already present and precedes the detailed modelling.

3 Does working on notions matter? An example.

Does it pay to work on notions and is that still TRIZ? One of the reviewers of this article categorically opposed at least to the second part of the question and asks for an example. In this section such an application is discussed in more detail.

We examine the following slightly modified problem from [12]: The municipal library has been given a new building in which both the books as well as the lending office should move. Unfortunately the budget is not enough to assign the transportation of the books completely to a service agency. What to do?

For the TRIZ modelling we must first decide which system-supersystem relationship should be considered more closely. Our conceptual system, which starts from a *world of technical systems* as causally connected black boxes, criticizes the term *supersystem* and instead speaks of *neighboring systems*. The *purpose* of this first step in TRIZ modelling is to choose one system in the world of technical systems as black boxes whose more precise structural-functional analysis (hopefully) leads to the solution of the problem. This selection (Part 1 in ARIZ-85C) is largely heuristic, where besides the “undesirable effect” mainly the MUF (main useful function) is guiding the action. In our conceptual approach, this coincides with the “purpose” of the system selected for detailed analysis, that is, the *reason*, why at all this system exists in the world of systems. This reason is essentially derived from the importance of the system for one or – usually – several neighboring systems. So the MUF is more important than the exact identification of the supersystem.

I explained this in detail, since our library example here already presents the first hurdle. It is largely unquestioned (or we set this as heuristic assumption) that the library is the socio-technical system to be analyzed in more detail. Also its technicality (both with regard to the artifactual equipment, e.g. the IT systems, as well as with regard to the technicality of the internal operational organization) hardly anyone will doubt. For an SF modelling[1] it is problematic that the transportation of the books is a temporary additional function of the "system library" that has little to do with the (not yet determined) MUF. On the other hand, the determination of the MUF is important since it allows to identify available resources and relationships with particular clarity. For our conceptual framework this is even *unavoidable*, since the system specific term of *essentiality* is bound to the MUF.

MUF. So, instead of dealing directly with the transportation of the books, we must first take the detour about the MUF, which we postulate as follows: "The library lends books to readers".

Next (in the methodology of [12]) the question "How does the machine (i.e. the library) work?" has to be studied. [12] offers a template for this purpose, which asks for the energy source, the engine etc. up to the processed object and the useful product.

Energy source. The question about the library's energy source seems idle, but here is a first essential difference between our conceptual approach and the usual understanding of TRIZ. In our approach in addition to the MUF as a form of description of the *purpose* also the *throughput* of the system plays a central role, which is constitutive for the reproduction of the internal structure of the system. Altshuller's 1984 list of eight laws of technical evolution yet contains such a concept of structure by throughput as "energy conductivity" [6, p. 2], but such a concept is missing in current versions of the "trends" [6, p. 6]. On the contrary, it counts as great achievement, when a system has evolved to the point where it integrated the energy source from the supersystem as component into its own system [6, p. 40]. How I have to imagine such an "energy self-sufficiency"? Isn't energy "used up" at some point? Has the car to be refueled at some time to refill its energy source? So is the energy source not an energy source, but only an energy storage, and the whole achievement consists in the fact that the energy throughput changed from a continuous operation (TRIZ principle 20) to pulse operation (TRIZ principle 19)? But what is the progress if the step back (to TRIZ principle 20) can be sold as further progress? Of course it is a good idea to consider this energy storage as energy source, but only if the process of "filling up" the energy storage is faded out at the systemic reduction to the essential.

So what is the "energy source" of the library, i.e. the external throughput that supports the reproduction of the internal structure? This is obviously the *budget*, from which the library finances the organization of its operations. If one distributes this according to [12] more

[1]In view of the close connection between Functional Analysis [4, ch. 4.4], Substance-Field Models [4, ch. 4.9] and the application of the 76 TRIZ standards [4, ch. 4.10] I use this abbreviation deliberately ambiguous – on the one hand as substance-field modelling and on the other hand as systemic-functional modelling, see also [10].

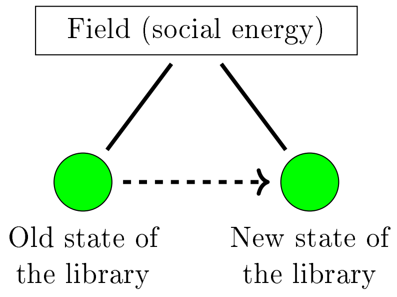
exactly on energy source, engine and transmission, we see that the "energy source" is money, the engine "the library budget" and "transmission", as the process of transformation of one type of energy into another, more useful one, the conversion of "money energy" into in "social energy."

Already at this point the analysis of the problem circumstances yields a solution proposal, which, however, requires a transition to the supersystem:

- *Cause of the conflict:* The problem is caused by an inadequate energetic throughput; with the available energy, the system cannot establish the required system structure, we need more money.
- *Refined problem model:* How can the library increase its budget?
- *Brainstorming solution:* Crowdfunding.

For reasons of space, a more detailed analysis of this problem model cannot be given here, which leads to significantly more substantial solutions proposing appropriate actions in the field of urban politics.

However, we are looking for a solution that can be implemented within the given global budget constraints of the library. For this we have better to understand, "how the machine works". Before delving into the details of the MUF, let's first try to understand what is happening with the library at large. In the best case we can avoid detailed modelling. The transportation of the books is obviously part of a transition of the library from an old to a new state.



This can be displayed in an SF diagram, where the problem is the insufficient action. The SF diagram (without "field") is already the complete functional model according to [4, ch. 4.4]. The inventive standards [4, ch. 4.10] can be applied to strenghten the insufficient action. For this purpose, according to [4, ch. 4.9] the functional model has to be transformed into an SF model adding a "field". The field is the source of the "energy" for the action [4, p. 185]. However, our diagram is not a substance field diagram, but a system function diagram. The "substance nodes" are two different states of the system, the "mediating field" represents the energy, as "social energy" already in "useful" form, that is required to *execute* the transition of the system from the old to the new state. The intensity of that field is partly determined by the conversion of "monetary energy" into "social energy" triggered by a simple contract with the transport agency.

To find ways to amplify this action we can still – as with "ordinary" SF models – apply TRIZ standards. This cannot be elaborated here in detail due to space constraints.Three conflict resolution hypotheses emerge:

1) If we had fewer books, the budget would be sufficient.
2) If we had more money (or other sources of "social energy"), we could transport all the books.
3) If the transport of a single book would cost less, then we could transport all books with the available budget.

All three hypotheses are worth a closer analysis and lead to very different solution proposals. We're just looking at option 1 to discuss another problematic point.

So how can the number of books be reduced that have to be transported? To analyze that question we now have to turn to the MUF and must understand more precisely "how the machine works". First of all, I briefly recapitulate how you become a reader: You sign up and get a reader's pass, which you use to identify yourself to the library's IT system as a person entitled to lend books. The TRIZ modelling according to the template "tool → action → processed object → useful product" ([12]) is clear: The *tool* is the IT system of the library (which is *controlled* by a library staff member as operator), the *action* is the registration as user, the *processed object* is the database entry of the user and the *useful product* the user registration.

But how does lending a book work? In this case there are *three* entities involved – the IT system of the library, the reader and the book. How distribute these three items to the *two* places in the template "tool → lends → object"? A closer look shows that here the tool IT system operates on a *relation* between reader and book. The whole story is also clear from a database point of view – in a (normalized) database scheme, there is a reference table "lended", in which a new record is created with the user ID and the book ID. We realize that objects can also be *relationships between objects*, and such a relation can easily be transformed into a separate object assigning an ID to the data record itself.

A clean concept formation thus leads to an object concept, which is commonplace in computer science, but difficult to think in TRIZ with its strongly artifactually backed object concept.

Shortly the rest of the story: We see that there are lended and not lended books, and only the books that are not lended must be transported by the service agency. So let's figure out how we can ensure that as many books as possible are lended.

4 The VDI's concept of technology

We explained in the last section the great importance of relational conditions among objects already in simple contexts of application. Szyperski [11] takes this as starting point for a clear separation of the concepts *component* and *object*. In the classical TRIZ it is rather difficult to distinguish between the notions "technical system", "component", "object", "element" or "product".

For example, the TRIZ glossary in the VDI standard 4521 [14] defines the term *technical system* as a "man-made totality of several interacting elements that serve a purpose" (ibid, p.8), but what does this mean? A glossary entry "element" is missing, just like the terms "component", "object" or "product".

The term "technical" appears in that standard in a second place in a context, where it is about the claim that TRIZ theory is able to identify "general evolution trends of technical, i.e. purpose-oriented, systems". Instead of purpose orientation Boris Zlotin points in his preface to [6] to a completely different central driver of evolution: "Innovation, which can be broadly defined as the development of new systems ..., is generally *driven by the advancement of scientific knowledge*" (my emphasis).

The concepts of the VDI standard 4521, based on purpose-driven "totalities of interacting elements", do not even offer language instruments to express the important to every engineer difference between model and reality of a technical system. The authors of the VDI standard 4521 must also be faced with the question about consistency of the VDI standards as a whole, because the (in [14] not even mentioned) VDI standard 3780 "technology assessment" [13] develops a much richer concept of technology as

- a set of benefit-oriented, artificial, representational entities (artifacts or physical systems),
- human actions and facilities where physical systems arise and
- human actions in which physical systems are used.

The definition avoids the terms "technical system" and "element" in favour of "benefit-oriented, artificial, representational entities". Similarly [8] with the concept of *technical object*, preceding the concept of a *technical system*. The VDI standard 3780, in addition to the artifactual dimension includes "physical systems", i.e. not only the machine, but also the machinery[2] and thus the unique specimens of large technical systems.

5 Cooperation, Competition and the World of Technical Systems

In the following sections we discuss in more detail three aspects of embedding our concept of a *technical system* into more complex socio-cultural processes. In doing so, we deepen and further develop our conceptualization. In a first step, we investigate the relationship between invention and acquisition of technical systems.

The description of planning, design and improvement of technical systems is based on the efficiency of *already existing* technical systems, which are present both as components and – from the view of a system in the supersystem – as neighboring systems. So, not only *inventing* but also *purchasing* appropriate components – or even just purchasing their *service* – as well as *cooperation with independent third parties* is on the agenda.

Engineering practices thus reside in a complex *world of technical systems*. From the descriptive perspective of a certain system other systems occur as components or neighboring systems with their *specification* only. Such a reduction to the essential practically appears as a shortened way of speaking about a social normality, what I call a *fiction*. This fiction can and is maintained in daily language use as long as the social circumstances are in operation that guarantee the maintenance of this social normality, i.e., as long as the *operation of the corresponding infrastructure* is guaranteed. Thus technical systems are, at least in their enforcement dimension, *always* socio-technical systems.

[6] nevertheless limits the concept of a *technical system* ("engineering system" in their notation), abstracting from social moments, in order to understand more closely their "evolution" from a "marketability" perspective. Such an access is quite plausible, because on the one hand, every concrete good is a technical system in the sense developed above, since it is a set of "useful" features given by their specification and, on the other hand, these features are to be

[2]Marx goes even further in this point: Only the "*automatic system of machinery* [...] turns the machinery into a system" (MEW 42, p. 584).

considered as *purposes* which are bound under modern economic conditions to their proof in the exchange of goods.

But this set of useful features also determines possibilities and limits to the substitutability of goods in the overall technological process. Those borders lead to a stratification of "the market" into specific markets for specific product groups. This real-world structure of the *technology markets* precedes the S-curve analysis in [6] and is implicitly taken for granted there. Each such technology market is characterised by a specific set of technical functionalities, where [6] assumes with the postulate of an MPV (main parameter of value), that such a market is grouped around a special technical parameter, which is of particular importance for the creation of value.

This, however, is connected with the requirement of a further abstraction, because concrete goods, in the above sense as technical systems, as *concrete* bundle of technical functionalities, are in principle suitable to be traded at *several* such technology markets and practically do so. Such a technology market is also less likely defined by the goods traded on it, as rather by the companies producing these goods. But this shifts the need for abstraction from a MPV as an independent characteristic to the *ability of companies to produce* technical artifacts with this MPV within appropriate price-performance ratios. This also makes it clear, that the conflicting relationship of goods on the market reflects the contradictions and tensions between the producers, rooted in the difference between the founded expectations and the experienced results in *former* exchange of their work products.

However, this competitive relationship on markets is embedded into the larger cooperative relationship of the capability of components, *principally* to work together. The question "What is a component?" at the very beginning of [11] has a simple answer: "Components are for composition". In the rest of the book the concepts and modes of such a "composition" are developed. The concrete composition procedures are manifested in the corresponding *component frameworks* such as Spring Boot, that serves as a common bracket for the provider of a component and the customer as user of this component. It is the common context for their actions to provide the component and insert it in the complex system structure of the customer. The component framework describes not only the principal interaction of the components based on the requirements of normalization and standardization on a higher level of abstraction (description dimension), but is also offered from different suppliers as runtime environment for components (enforcement dimension).

Such runtime systems – certainly also technical systems – are special: they are *jointly* operated by the provider and the customer, which requires a coordination of the socio-technical accompanying processes at a high level. In such a system, a division of labor is present – the provider is responsible for the quality of the functionality, the customer for the quality of the data. The customer is also responsible for the functions and malfunctions of the whole system to the general public and is accounted for damages caused by its operation within the scope of a prima facie evidence in legal suits. Hence the operational (technical) quality of the real world system in this socio-technical relationship is equally influenced by *both* parties.

6 Normalization and Standardization

This software industry approach is also present in many engineering and technical applications. "Modular systems" are widely used and make it possible to create unique technical real-world systems in a simple way, combining the *logic of the specialized application* as "core concern" of the component with the *logic of networking* within the infrastructure as "cross cutting concerns". Both logics are orthogonal to each other, which means that the trends 4.2 "of increasing system completeness" and 4.4 "of transition to the supersystem" in [6] practically act in different directions.

For modular systems and "mature" technologies an increasing system completeness plays clearly a less important role compared to combinability of components, as a visit to a DIY store immediately shows – the machine systems of well-known manufacturers concentrate on the provision of energy, suitable tools can be used combining them via appropriate APIs (such as velcro, screw or click fasteners on the mechanical level) with the energy machine[3], whereby, depending on the business strategy of the well-known manufacturers, the respective technology sub-market of "suitable equipment" is monopolized or open also for less well-known manufacturers of suitable work equipment. In both cases, *normalization and standardization* in this "world of technical systems", i.e. inherently socio-technical processes, play a much greater role than the further technical refinement of artifacts.

At the same time, such a standardization process opens a perspective for an economy of scale for standard components, i.e. for implementations of "mature" concepts established in the direction of "ideal final results". The economy of scale induces an costs-*reducing* effect per individual item and thus shifts the main development focus from the competition for the better *technical* solution to the competition for the more cost-effective *economic* production. So the S-curve does not necessarily end – and probably rarely does – with the decommissioning in stage 4 [6, p. 38], but reaches in stage 3 of mature *technical* quality a turning point towards a phase of *general availability*, in which the *ever lower* economic expenditures for the availability of this "state of the art" take over the leading function of the further development.

Hence the trend 4.1 *of increasing (technical) value* turns into a trend *of decreasing economic value*, or – to express it in economic terms – the market previously driven by demand changes to a supply-driven market: The same (mature) use value has ever lower exchange value. This corresponds to TRIZ principle 17 of *transition to another dimension.* Thus in stage 3 of the S-curve development the leading function (MPV) in the *system of production of common tools and standard components* moves from technical driving forces to economic ones. This process of "commodification" was described in sufficient details by F. Naetar [7], so there is no need to embark into details here.

This *phase boundary* in the world of technical systems, identified based on the TRIZ principle 17, separates two development phases from each other – "young" and "mature" technologies. At the same time, we see at this point that the concept of technical systems as *bundles of functionalities* from *different* technical areas has to be opened and rebundled in the concept of *technical principles* according to *similar functionalities* properly to understand evolutionary processes in that world.

Such a phase transition serves as general development law of technical systems, but plays no role in [6] obviously due to structural reasons – the background of the experience analyzed in [6] are, as in large parts of TRIZ, the inventive practices *before* this phase transition, which focuses on patents and the further development of the "state of the art". The second phase, however, the wide operation of a generally available technology, is also full of contradictions

[3]The progress of material sciences, in particular with hook and loop fasteners, initiated a massive return to *mechanical* coupling principles despite of the TRIZ principle 28 of *the replacement of mechanical schemes.*

and the target of the activities of a new generation of TRIZ practicioners, which are much more closely related to the immediate needs of the technical requirements of *production*.

7 Purposes

We identified, in addition to the classical TRIZ world of problem solving, related to "inventing" and exploration of new territory, a second world of technical systems, which presents itself, through normalization and standardization, as a world of general availability, in which it seems, there are no more problems since all works fine – at least as long as the societal efforts to operate the infrastructural basis of this *fiction* are successful.

From both directions we end up in a *world of technical systems* or perhaps only of technical artifacts, or rather "technical objects" that N. Shpakovsky in [8] takes as starting point for the question, whether every collection of technical objects is already a technical system or which additional requirements must be met for this.

Shpakovsky's answer is that technical systems are characterized by a *well defined purpose*, which is given *from outside* and has to be fulfilled by the system. The system notion developed here is well suited for such an approach, because, with the *specification* of a component, purposes can be well and accurately expressed. Less clear is for now, *where* these purposes come from. The TRIZ answer is clear: "From the supersystem". However, in [3] it is shown that there may be *several* supersystems to a system and the term of a *neighboring* system is more appropriate to the situation. A world of technical systems without hierarchies is a world of *relations* between technical systems, out of which *purposes* can be explained: *That* system has been developed because *this* requires its useful function for its own operation. So the purpose of *that* system is to serve *this* one. Technical objects *bundle* this way functions and services of different components, to create and offer own services. Let's apply at this point a substance-field swap and consider the *relational*, the individual function, the individual service *as substance* and the technical objects bundling the functions as relations, as *mediator* between these functions. *Purposes* are in such an understanding *requirements* arising from human practices, according to which such functional bundles are compiled. *Evolutionary lines* of the development of technical systems originating in such functional bundles follow evolution principles of fields and start with simple compositional principles – to a bolt find a suitable nut –, range over established procedures – what to consider when painting a window? Which colours choose? Which brushes? How to prepare the surface? How to paint? – to higher forms of abstraction such as the form that is needed to form a larger quantity of bricks from clay, to burn them and then build from them an entire house.

However, abstractions that structure these purposes are not arbitrary, but follow in their turn *purposes of second order*. Szyperski [11, p. 139 ff.] identified under the heading "Aspects of scaling and granularity" a long list of such "purpose patterns", according to which *functions* are bundled into units, i.e. as components, for example as a unit of abstraction ("design expertise embodied ready for use"), as a unit of accounting (unit of cost monitoring), as a unit of analysis (unit of troubleshooting), etc. These *purposes of second order* are not independent of each other, as [11, p. 145] notes – a unit of analysis can not usefully broken down into *several* units of expansion.

These manifold *practices of component formation* are in turn not independent from each other, but constitute their own worlds of practical interactions and experiences. In the *component* *framework* these experiences are normalized and standardized. The omnipresence and convenience of the use of technical objects is thus reproduced at the level of engineering and technical activities, which in turn is not primarily rooted in the principles themselves, but in the *capability to fit* with other relevant principles. This ability to fit does not fall from heaven either, but is in turn a result of *reasonable cooperative human practices.*

The world of technical systems is thus embedded into a world of relationships of technical systems, which reflects complex socio-technical relationships that are driven by specific purposes. These purposes are in turn numerously related to each other, and it is this relationship structure, which is subject to structuring by these "purposes of second order". This is not the end of the story, since component frameworks themselves are structured through comprehensive design pattern [1] and process pattern as "dependency injection" or "inversion of control" etc. This cannot be discussed here in detail.

We see that for the study of evolutionary aspects it is more important to understand the world of *relationships* between technical systems than the world of technical systems itself. In our system theoretical approach, such a relationship is considered as a relationship between components of a system primarily under a *functional* perspective as specification in the description dimension and as promise of guaranteed specification-compliant performance in the enforcement dimension. Subject of the TRIZ *methodology* is the transformation of the one into the other. TRIZ methodology itself is only a form of description, that helps domain experts to perform this transformation *practically.* Functions appear in this transformation process in three modes: *before* the transformation as *purpose*, as something you would like to have in the world, *within* the transformation as *implementation* and *after* the transformation as *service*, as a *realized promise.*

Technical systems thus appear as functional bundles and their individual element – the function – in three different modes: as purpose, as implementation and as service.

8 Summary

In this paper, an attempt was made to expand a theoretical context for a concise notion of a *technical system.* It turns out that when looking at special partial questions a reduction to the technical dimension alone is useful, but in a more general context, the position and significance of technical systems cannot be understood – neither in the modelling perspective nor in the perspective of realization – without consideration of the (socio-culturally determined) *purposes* of its existence. Moreover these purposes reflect rather *relational structures between* technical systems, which only become apparent in the world of technical systems as a whole and not already from the analysis of single technical systems.

We further showed that there is a big difference between "young" and "mature" technical systems. In the case of the former there is indeed the technical dimension of the development of concrete *technical principles* by *inventing* in the foreground, but with the latter the socio-economic dimension of the *involvement* of external technical know-how at the concrete problem solving level becomes more important and thus the socio-cultural networking structures of a *world of technical systems* that require *private procedural skills* of a completely different kind.

Hence also here the whole is *more* than the sum of its parts.

References

[1] Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides (1995). Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley. ISBN 978-0-201-63361-0.

[2] Hans-Gert Gräbe (2020). Reader for the 16th Interdisciplinary Seminar *The concept of resilience as an emergent characteristic in open systems* on February 7, 2020 in Leipzig. `http://mint-leipzig.de/2020-02-07/Reader.pdf`.

[3] Hans-Gert Gräbe (2020). Die Menschen und ihre Technischen Systeme (Men and their technical systems). LIFIS Online 05/19/2020. DOI: `10.14625/graebe_20200519`.

[4] Karl Koltze, Valeri Souchkov (2017). Systematische Innovation (Systematic Innovation). Hanser, Munich. Second edition. ISBN 978-3-446-45127-8.

[5] Simon Litvin, Vladimir Petrov, Michail Rubin (2007). TRIZ Body of Knowledge. `https://triz-summit.ru/en/203941`.

[6] Alexander Lyubomirskiy, Simon Litvin et al. (2018). Trends of Engineering System Evolution. ISBN 978-3-00-059846-3.

[7] Franz Naetar (2005). Commodification, Wertgesetz und immaterielle Arbeit (Commodification, law of values and immaterial labour). Grundrisse 14, p. 6–19.

[8] Nikolay Shpakovsky (2003). Человек и Техническая Система (Man and the technical system). `https://wumm-project.github.io/Texts/Shpakovsky-mts-ru.pdf`

[9] Nikolay Shpakovsky (2010). Tree of Technology Evolution. Forum, Moscow.

[10] Yevgeni E. Smirnov (2016). Элементно-функциональное моделирование конфликтов: ЭФМ.К (Element-function modelling of conflicts: EFM.K). In: Три поколения ТРИЗ. Материалы ежегодной научно-практической конференции, посвященной 90-летию Г.С. Альтшуллера (Three generations of TRIZ. Materials of the annual scientific-practical conference, dedicated to the 90th birthday of G.S. Altschuller), St. Petersburg 2016.

[11] Clemens Szyperski (2002). Component Software: Beyond Object-Oriented Programming. ISBN: 978-0-321-75302-1.

[12] Target Invention (2020). TRIZ Trainer. `https://triztrainer.ru`.

[13] VDI-Norm 3780. Technikbewertung – Begriffe und Grundlagen (Technology assessment – Terminology and basics). September 2000.

[14] VDI-Norm 4521 Blatt 1. Erfinderisches Problemlösen mit TRIZ – Grundlagen und Begriffe (Inventive problem solving with TRIZ – Basics and terminology). April 2016.