Chat of the 17. Interdisciplinary discussion

With comments by Hans-Gert Gräbe (HGG)

Presentation of Nikolay Shpakovsky

The notion of a system was introduced as modeling a problem ("mental image") of different complexity

- as assembled device
- as tool processing an object (SAO model, funtional model)
- as potentially workable system
- as system for a certain goal
- as product producing system
- as harmful system of Lenyashin

[10:47] Kleemann : how can the approach of the x-factor grasp notions of development of a system? (relates to slide 36)

HGG: The system has to be completed (law of completeness of the components of a TS), so development is completion of an incomplete system.

[10:49] Igor Zadesenets : The question is what is directly?

[10:50] Kirill Domkin : What is the difference between Engine and Energy Source?

HGG: Energy source provides energy, engine transforms energy into useful energy.

[10:52] Alexander Solodkin : let me ask verbally please. I have some doubt about introducing the product term.

HGG: This remained unclear, but relates to my presentation. If the tool (or component) is considered completely functional and (only) the object carries state, then the "product" is the transformed object, i.e. the object now carrying a special state. The idea here is as in market terminology – the product is an object with special use value, we forget about the production of the use value (the way of production is abstracted in the exchange value, at least in an economic labour value theory).

[11:00] Immanuel Thoke : @Kirill as I see it the Energy Source is the Input for the Engine to produce the output as the input for the transmission.

[11:18] Andrey Trostyanskiy : Can the harmful system make a useful effect?

HGG: As far as I understand TRIZ has no notion of life cycle of a technical system. This relates to the phase of maintenance and repair of a technical system already in use, the task is to transform it from a state with harm to a state without (that) harm.

[11:19] Immanuel Thoke : states define a property of a system itself, whereas a product describes the outer view between given input and specific output, of course this state is achieved due to a certain input, but it doesn't describe how requirements have been matched by the system.

[11:22] Stelian Brad : I would like to add something

HGG: Another question was about the IFR. Alexander explained that this idea is similar to the idea of absolute truth and hence only useful in a very special mental setting. In a theory

of adaptive systems this makes no sense. Stelian explained that this concept is also present in a slightly different meaning as "desirable state" in the systems notion of Russell Ackoff.

[11:25] Immanuel Thoke : shouldn't we distinguish between ideal and realistic goals and absolute and relative goals?

[11:40] Immanuel Thoke : ok what Alexander Solodkin said is very interesting. Yesterday I thought about following: Oneself cannot control itself, but it can control itselfs in perspective of different contradictory components with different ideal goals towards optimization between the parametric contradictions.

[11:41] Igor Zadesenets :

1. Different IFR from different perspectives can be set.
2. IFR is only the tool for finding achievable results.

Presentation of Nadine Schumann

The main point was a distinction between different notions of law:

- in the strict sense (pure causality as in Newton's mechanics)
- scientific laws (statements based on repeated experiments or observations)
- laws directly of incirectly based on empirical evidence

Laws of development are of the third kind, based on heuristical principles as the foundation of interpretation.

This was related to Wundt's account on psychical causality.

Then the question of Trends of Development and Evolution of Technical Systems was discussed. The notion of evolution has a restricted meaning according to its historically biological origin and may be not appropriate in the context of technical systems.

[12:17] Siegfried Weigert (ibw) : What is Law (In this context here): An "Item" that "always has been there" like mathematical/physical "laws" OR "A set of rules that has been defined by"lawyers.

HGG: That's an interesting question not discussed by Nadine. As far as I understood for her "law" has "always been there" but even "been there" has a multitude of interpretations (the notion of law differs even between mathematics and physics). The "lawyers' laws" as written rules are only a specific form of processual institutionalisations, many of the more genarl ones are informal but nevertheless binding, e.g. the "state of the art", in the sense that no one expects that you do *not* follow these rules. If you do not follow the rules you have at least inconvenience in cooperation and may be socially punished in the worst case. This institutionalisation transforms practically proven to proven practices that get social binding quality.

[12:22] Stelian Brad : I would like to have a comment

Stelian just drawed on that distinction. The notion of law applies to two different settings:

1. describe how something evolves in an organic manner (descriptive)
2. laws as regulations to fight complexity (activity centered)

[12:22] Kleemann : does that mean you have to start with a "product" and the circumstances?

[12:24] Alexander Solodkin : I'd like to comment as well :) After Stelian :)

Alexander emphasises that for the notion of law predictability is important and distinguishes laws from phantoms.

[12:28] graebe : Some more questions: @Nadine: What is a lifeline of a system? @Stelian: What is a system that meets decisions?

HGG: Lifelines I found first in (Altschuller 1979) but for me it remains mysterious until now what are the constitution principles that are used to line up technical systems in such lines. (TESE 2018) draws on "pragmatic S-curves" and it seems obviuos for me that they are drawn from commercial product catalogues, as explained in (Gräbe 2020a). Biological evolution is usually drwan in "evolution trees" and such a concept is presented also in (Shpakovsky 2010) for "technology evolution". But "technology evolution" and evolution of technical systems seem to be quite different topics.

HGG: Stelian focused on the point that (management) systems are related to people and decision meeting processes. This aspect is, in my opinion, underestimated even in modern Business TRIZ concepts. In (Gräbe 2020b) I proposed to make a difference between systems for decision preparation and systems for decision meeting. They have very different characteristics as convergent an d divergent thinking (according to Guildord) have.

[12:35] graebe : @Alexander: What is "human"? The individual or the mankind? Or something between (we call it cooperative action)?

[12:35] Immanuel Thoke : what is meant by "exact"? What do mathematicians mean if they say "nearly ever"?

[12:39] graebe : @Nadine: So this notion of "human" is as by Vernadsky – Scientific thought as planetary phenomenon?

HGG: Nadine in her answer did not address the question about "lifelines" but the question "what is human".

[12:40] Kleemann : it seems to me, that the problem that Nadine touched is the construction of dynamics and not of structures. Do you mean that this view of development has to be revised?

[12:53] Ulrich Fritsche : = feature, property

[12:56] Kleemann : then the question arises: can you model a pragmatic s-curve, which is non-linear. Is there more than one pragmatic s-curve possible?

[12:57] graebe : I proposed a branching of the "s-curve" at its phase 3 in (Gräbe 2020a).

[12:59] Immanuel Thoke : a classic approach would be geometric or physical linearization and representation of error rates in probabilites of these uncertainties.

[13:01] Kleemann : @thoke: then you have the problem of linearity again. Maybe other mathematical or logical approaches, as fuzzy logic or patterns in chaotic processes.

[13:09] Immanuel Thoke : yeah triz approach is again separation in time and space .. relying on quasi-stable phases where minor instabilities can be controlled by a dominating protocol.

[13:10] Alexander Solodkin : may I add something?

Alexander emphasises that development is not so easy in a biased world where you have to manage also dissent. He several times emphasises the management concepts realised within

Toyota[1] as a concept centered system of processes. Moreover he emphasises that the managerial style strongly depends on the context, in particular is different in stable and unstable times. For this point see also (Mann 2019).

[13:13] Immanuel Thoke : that refers to the term of exacteness and scalability.

[13:14] graebe : There is a good rule in open source: Raw consensus and running code.

[13:21] Immanuel Thoke : (polemic question) so we handle the black spot i.e. in realization of climate change not knowing how to fix the system in time because it does not meet the actual requirements of the states, companies and people and hoping for better technologies in future that will fix this in time or rely on adaption of changing environment?

HGG: As I understand it this is – roughly – the core of the program developed in (Shchedrovitsky 1981).

Stelian: This is a difficult question since the particular circumstances and decisions influence many other particular circumstances. This requires to work and concentrate efforts on the non-technical conceptional frontier to stabilise the macrosystem.

[13:28] Immanuel Thoke : yeah but we have also corona .. have you also experienced the quick change between cold and warm phases?

[13:31] Immanuel Thoke : thanks for the vivid discussion so far

Presentation by Hans-Gert Gräbe

The presentation focused on developments in the area of software engineering that deeply changed during the last 15 years from a "programming from the scratch" to a component oriented production style. If 15 years ago the knowledge of a programming language was enough to produce valuable code nowadays additionally the knowledge of a framework (consisting of both a "glue code concept" and a wealth of available components) is required to be productive in coding. Within that development also new professions came up – the profession of a software architect (engineering platforms), a component developer (providing coding within a platform) and a component assembler (assembling larger systems from components). Such "larger systems" accompany "larger technical systems", and both are unique specimen as explained in (Gräbe 2020a) and – differently to components – not designed for direct reuse. Technological development thus moves from direct reuse of components to higher forms of abstraction and modeling. Such questions are not at all addressed in the traditional TRIZ theory about laws of evolution of technical systems.

Stelian: Software systems are the highest complex systems and the most invisible systems.

HGG: There are many concepts within software engineering to address this complexity, e.g. ITIL[2] or SAM[3].

[14:14] Kleemann : can such a design be used for a non-linear s-curve approach.

[14:22] Kleemann : structure to graph. interesting

[14:35] Nadine : TRIZ is open enough to model that. this is the big advantage of the TRIZ method, or? it guarantees the flexibility of the Supersystem.

[14:39] Igor Zadesenets : Question for Nikolay. What is the fundamental difference between

[1] https://global.toyota/en/company/vision-and-philosophy/production-system/
[2] https://en.wikipedia.org/wiki/ITIL
[3] https://en.wikipedia.org/wiki/Strategic_alignment

Engine, transmission and so on and Control? May be they all are Control?

HGG: In my opinion this chain is an expression of the "law of energy conductivity through the system", that can be rooted in the theory of dissipative systems. This theory claims that energy throughput through a system strongly relates to the structures within the system (as, e.g. for the Bénard cell). Hence energy dissipation within a system is required for both the supply of the required energy throughput for the components and the energy required to regenerate the functional structure of the system itself.

[14:41] Immanuel Thoke : so chaos also arises from not knowing at which abstraction level causes a problem as an engineer with no transparent insight and sufficient debugging infrastructure have to deconstruct each level of abstraction to determine the error.

HGG: If you take the Theory of Dynamical Systems and the notion of attactor as a generalisation of the notion of equilibrium, you have the notion of "deterministic chaos", i.e. attractors of systems described by relatively simple deterministic equations that have themselves complicated structure, even up to fractal buildings (strange attractors). So this is not necessarily a problem of not enough insight.

[14:44] Igor Zadesenets : OK. And what is the fundamental difference between process and component?

HGG: Whats about such an explanation? A component is a black box view on a process.

[14:48] Immanuel Thoke : but it's hard to determine sideeffects in python due to unstrict typing especially if you use libraries you do not know.

[14:50] Kleemann : in this cocept means modelling not directly using a model?

[14:53] Nadine : Thats the concept of pragmatic information, or? in contrast to syntactic and semantic concepts of information.

Presentation of Stelian Brad

From the presentations:

Innovation is more than an invention, it is not only a technical phenomenon but has to address the needs of a target group:

Innovation ... a nonlinear, multiple-looped and agile process through which a novel idea is generated and then embedded into an elaborated viable solution that addresses a need of a given target group in a way that fits the group's culture; thus, being wanted, affordable, valued-for-money and adopted.

Systems have to be maintained, this may be very challenging due to

- complexity – the behavior at the layer between deterministic and chaos, with high nonlinearity, where small changes in the value of some parameters lead to radical unexpected evolutions.
- complicatedness – a system with intricately combined and involved parts such that it is very difficult to understand or analyze
- dynamicity – a state of being in perpetual change, which makes dataset difficult to keep accurate.
- abundance – a state of oversufficient quantity, thanks to technology (e.g. abundance in communications)

Three concepts are important to conceptualize maintenance:

- entropy → a state of disorder, uncertainty, randomness in the system, the level of possible combinations of the parts in the system.
- equipotentiality → apparent capacity of any intact part of the system to carry out functions which are lost by the destructed parts.
- equilibrium → state of balance relative to the forces acting in the system.

These points were discussed on the current state of software development.

[15:10] graebe : In Rubin's work I have found the concept of system capture several times. What role does this play in an environment with many systems?

[15:14] graebe : Interactions (one-to-one) increase only quadratically, exponentially increases the number of relations = subsets of the set.

HGG: This relates to slide 4. Exponential growth is a pure theoretical phenomenon, describing "natural" growth ($y'(t) = c \cdot y(t)$) without limiting factor. Adding a limiting factor you get e.g. a logistic curve. For growth of networks the phenomenon of scale free networks[4] was observed that develop not exponentially but following a power law.

[15:15] Immanuel Thoke : does complexity increases unboundedly if we swap paradigms or does complexity patterns change?

[15:18] graebe : In the theory of Dynamical Systems there is a notion of Steady state Equilibrium (that can move in space). How it relates to the 3 e on the slide?

[15:23] graebe : Machine tool building vs. industrial plant engineering.

HGG: It is one of the main observations from a practical point of view: the main task for recent software engineering is to organise digital support for already existing business processes. Digitization requires to "make things explicit" that existed so far in a less explicit form, so the hardware and software modeling has to be accompanied by an "orgware modeling" (a concept introduced by K. Fuchs-Kittowski, see (KFK 2020), in German). This has much to do with modeling and not so much with innovation, but within modeling formerly unnoticed contradictions (and unformalized implicit workarounds, the "social soft") pop up. This is the today business of semantic modeling, data analysts, business process modeling, organisational modeling, software architects and component assembler.

As the design of large technical systems (plants) is unique since the Production management systems differ significantly even within the same company (we observed that for BMW), IT systems are unique specimen, too. From such a point of view the challenges to software engineering look quite different and it is just that view that is addressed in Szyperski's book (see my presentation). May the "software paradoxes" (slide 7) result from such a difference in viewing the topic?

"See the growing popularity of Python or ROS (Robot Operating System) – huge libraries" (slide 9) – Python or ROS are very different things. If you look at the .NET framework then ROS is similar to the CLR, the "virtual machine" supplied by the Common Language Runtime required to integrate all components, written in different languages (e.g. in Python). But for different languages to work together the CLI (Common Language Infrastructure) is required.

[4] `https://en.wikipedia.org/wiki/Scale-free_network`

[15:31] Immanuel Thoke : in TRIZ there is a non-invasive alternative to the invasive intonation of breakthrough but this might be subjective as I see the examples – turning something inside out in order to move to abstraction level of the supersystem and integrate it in the system.

[15:33] graebe : What's about "coopetition" (in German *Kooperenz*[5])? This notion was coined in discussions already 15 years ago. "Raw consensus and running code" is just about that.

HGG: In the discussion the notion of "software ecosystem" played a significant role "defined as a set of businesses functioning as a unit and interacting with a shared market for software and services, together with relationships among them. These relationships are frequently underpinned by a common technological platform and operate through the exchange of information, resources, and artifacts."[6]

Such a "technical platform" is not a static thing but has to develop, serve the needs of the users of the platform, and has to be operated. It was just that topic explained in my presentation on the example of CORBA. Open Source movement developed such a platform with git, github, gitlab, CI (continuous integration), git development models etc. `https://symbolicdata.github.io/Using.Git`

[15:37] graebe : But the Microsoft position depends on its internal development from the "Halloween Papers" 1998 to Open Sourcing .NET and the "Dotnet Foundation".

[15:46] Immanuel Thoke : supply chain management often relys on coopetition – cooperation on infrastructure, competition on market prices

[15:48] Alexander Solodkin : may I comment?

HGG: Alexander emphasised that complexity requires cooperative actions and mentioned the work of Rapoport – I found (Rapoport 1977) – on that topic. Cooperative action played also an important role in our lecture for the students during the semester, but we emphasised on the importance of the interrelation of cooperative acton and common conceptual world.

[15:57] Immanuel Thoke : referring back to climate change .. a breakthrough would be to see the ecosystem as a inner property of infrastructure of economic systems and marketing products not as a companies property but as a product of ecological process this may sounds trivial but household calculations works the other way around.

[15:58] Igor Zadesenets : May I comment?

HGG: Igor mentioned that we much deviated from the original topic and asked what's about the TRIZ laws of development of technical systems themselves. What is the state of art, what is their benefit etc. I commented that in two small writings.

- `http://mint-leipzig.de/2021-02-05/Laws.pdf`
- `http://mint-leipzig.de/2021-02-05/Laws-State.pdf`

[5] `https://www.freie-gesellschaft.de/wiki/Kooperenz`
[6] `https://en.wikipedia.org/wiki/Software_ecosystem`

References

- (Altschuller 1979) Genrich S. Altshuller. Creativity as an exact science.
- (KFK 2020) Klaus Fuchs-Kittowski. Informationssystem-, Arbeits- und Organisationsgestaltung in Produktion und Verkehr – das Orgware-Konzept, die Paradoxie der Sicherheit, des Wächters, der Beherrschung großer Datenmengen. https://www.informatik.uni-leipzig.de/~graebe/Texte/Fuchs-20.pdf
- (Gräbe 2020a) Hans-Gert Gräbe. Human and their technical systems. In Proceedings of the TRIZ Future Conference 2020, p. 399-410. https://hg-graebe.de/EigeneTexte/mts-20-en.pdf
- (Gräbe 2020b) Hans-Gert Gräbe. TRIZ and Systemic Transitions. Submitted to TRIZ Reviews. https://hg-graebe.de/EigeneTexte/sys-20-en.pdf
- (TESE 2018) Alexander Lyubomirskiy, Simon Litvin et al. Trends of Engineering System Evolution. ISBN 978-3-00-059846-3
- (Mann 2019) Darrell Mann. Systematic innovation in complex environments. In: Online Proceedings of the TRIZ Summit 2019 Minsk. https://triz-summit.ru/confer/tds-2019/articles/reports/
- (Rapoport 1977) A. Rapoport, O. Frenkel, J. Perner. Experiments with cooperative 2×2 games. Theor Decis 8, 67–92. https://doi.org/10.1007/BF00133087
- (Shchedrovitsky 1981) Georgi P. Shchedrovitsky. Principles and General Scheme of the Methodological Organization of System-Structural Research and Development. https://wumm-project.github.io/Texts/Principles-1981-en.pdf
- (Shpakovsky 2010) Nikolay Shpakovsky. Tree of Technology Evolution. Forum, Moscow.