Design Automation: the Right Track

Vladimir M. Sedenkov*

* *Belarusian State University, v.sedenkov@gmail.com*

Abstract: While applied expectations related to design automation are partly met by virtue of CAD, the fundamental ones – structure synthesis problem tackling, continuous design visualization, design process complexity – are still waiting for real design automation, outside of which they are not attainable. Meanwhile, even a correct definition of design automation is hardly possible within the current human-centered and analysis-rooted design technology. The latter serves as a strong barrier to synthesis-targeted computer diffusion across the design event. This paper paves the way to real design automation – Computer Urged Design (CUD) – through exact treatment of basic design notions, problems, processes and design technology change.

Key words: Process, process automation, design process quantum, meta design process

1. Introduction

Among the problems still waiting for sufficient solutions, design automation is of primary importance: its tackling is a prerequisite for efficient coping with the key challenges – product structure synthesis, continuous design visualization and design process (*DPR*) complexity. However, the mainly semi-intuitive design terminology together with unclear *DPR* as automation object postpones the progress in real design automation – *Computer Urged Design* (CUD). Though task-by-task design computerization (resulted in CAD) is the inherent component of CUD, it cannot substitute the latter.

Indeed, the object of any automation is a process. Designing can eventually be also reduced to a certain process (not *DPR*, in exact terms). Hence, when coupling the concept of any process automation with a process associated with a design event, we should come to automated design. However, this simple concept is not a dish for simple realization. No wonder it is the infinite design computerization that is mostly posed for design automation though the latter could not be even defined within the current purely human-centered design technology.

So, after the first step in design automation strategy when exact definitions of a simple process and its automation have been obtained, the key process of designing should be detected. Prior to its automation, this process needs to be adequately systematized. This may require design technology modification or change, which should be first formally identified to clarify its adequacy criteria. Then the process associated with a complete design event may be automated. Overtaking the events which shall follow, we unveil the main results: the process in focus is a process of real time physical design process synthesis or *meta design process*; the adequate technology is a processor (*P*) or *P-independent design technology*, i. e. equally understandable to both types of processors engaged in its realization – informal (human being, *H*) and formal (a computer, *C*); meta design process automation consists in assignment to the process subject a *diprocessor* of C^H type where the master processor is *C*, and reference one is *H*.

The reminder of this paper is organized as follows. Sections 2 and 3 present formal tools used throughout of this research. Section 4 reflects on available and desired levels of *DPR* synthesis systematization. In Section 5, we remove from a shadow the notion of design technology, set off it from design methodology, and introduce a mechanism of technology identification. Section 6 describes *P*-technology. Design automation as *P*-independent determination of the meta design process is discussed in Section 7. Conclusions are drawn in Section 8.

2. Process Automation in General Case

2.1 Process Scheme as a Simple Process Formalization

Due to conventional definition, a process "is a change, procedure, or course of events taking place over a period of time, in which an object transforms, or is transformed, from one state to a preferably more desirable different state" [4]. Such intuitive definition prevents a process to be an object of formal representation, designing or conversions. On the contrary, continuous process theory [8] gives their constructive definition: a process *PR* is a pair formed of a target delivering procedure *D* (describes transformation of objects, energy, raw materials or information) and its implementer – a processor *P* (in general case, *H* or *C*). The record

$$PR=\langle D,\ P\rangle, \tag{1}$$

where *D* is a process *object* and *P* – this process *subject* is called *process scheme*. The scheme is characterized by its *uncertainty levels*: *conditional* one (*UL2*) if its *D* or/and *P* is completely unknown, *virtual* (*UL1*), if scheme's object and subject are determined mentally, and *logical* (*UL0*) when there is a physical *P*, and *D* has a description that can be realized by *P*. Another name for a logical process is *process design*. We come to the latter through reducing process scheme *UL*, called *process determination*.

2.2 Automated process

While $P\in PR$ is a physical processor, $D\in PR$ is a logical processor that processes control data. This causes two inputs of *PR*: the input of *P* (I^P) and input of *D* (I^D). As opposed to a given I^P, I^D could be a priori unavailable and generated in the course of *D* realization by another, different from *P*, processor. Processor engaged in I^P handling is considered as a *master processor* (P_m), while the I^D supplier is called *reference processor* (P_r).

PR subject, the part of which is performed by such a pair of processors ($P_m^{P_r}$) is called *diprocessor* (d*P*). As we have two types of processors – *C* and *H*, there are available four types of diprocessors for the part of *PR* subject: H^H, H^C, C^H and C^C. Accordingly, we have four types of processes: "manual" ($PR^M=\langle D,\ H^H\rangle$), computerized ($PR^C=\langle D, H^C\rangle$), automated ($PR^A=\langle D, C^H\rangle$) and automatic ($PR^T=\langle D, C^C\rangle$).

3. Design automation

3.1 Formal representation of a problem

As we noted above, the schemes of conditional processes (1) need to be determined till the virtual level, and virtual process schemes are waiting for logical determination to become a process design. To determine some conditional *PR*, there should work off two real processes: *SD* process (search for *D*) has to find out or synthesize $D\in PR$, while process *SP* (search for *P*) has to do the same with respect to $P\in PR$. In resulted triple of processes, its members are linked with relation of determination or **d**-*relation* ($\xrightarrow{\text{d}}$) and constitutes a structure out of process schemes (Fig. 1*a*). Processes *SD* and *SP* delivers results provided the input of *PR* is known. Otherwise, one more process is necessary – PR_1 that shapes for *PR* its input and on this account is linked with *PR* by the new relation (Fig. 1*b*) – providing or **p**-*relation* ($\xrightarrow{p}$).

$$SP \xrightarrow{d} PR \xleftarrow{d} SD \qquad\qquad PR_1 \xrightarrow{p} PR$$

a) *b)*

Figure 1. The first order structures of processes

The two relations on a set of processes and/or their schemes enable to construct structures out of process schemes, which may serve as continuous representation of available processes or describe determination (synthesis) of a conditional process as reduction of its uncertainty level (*UL2→UL0*) [8]. Let us use the structure shaped by processes in Fig. 1*a* to formalize the notion of *problem* (*Pr*).

Following G. Polya [7], we shall distinguish between *problem solving* and *problem answering*. A problem is solved if there have been found both an answer providing procedure *D* and processor *P* suitable for *D* realization, that is the processes *SP* and *SD* were completed successfully; the answer to the problem is supplied by *PR*. Keeping up the structure of processes declared in Fig. 1*a*, rewrite it in a linear form, which has been called *problem scheme* or problem formal representation (2):

$$Pr=<<SP><SD,\ PR>> \tag{2}$$

Problem solving and problem answering together are considered as *problem realization*. Problem scheme provided with *PR* inputs (I^P и I^D), claimed result O^P (*PR* output), and a type of $P\in PR$ is named *problem statement* (Fig. 2).

$$\begin{array}{c} \qquad\qquad\quad \Rightarrow O^P \\ Pr=<<SP><SD,\ PR>> \\ \qquad\qquad I^D \Uparrow\ \Uparrow I^P \end{array}$$

Figure 2. Ordinary problem statement

If at least one of the *PR* components (*D* or *P*) is undeterminable, the corresponding problem is unsolvable. Insolubility of the *first kind* means that *PR* object and subject are impossible in principle, while insolubility of the *second kind* manifests that the problem has no solution but an answer to it is possible. In the last case, the required answer will be obtained through implicit realization of another problem instead of original one. Problems with the second kind insolubility are not uncommon; they will be referred to as rhetorical or *pseudoproblems*. The problem, which is realized instead of a pseudoproblem is termed *goal problem*.

3.2 Problem and process associated with designing

Designing is a process. Even though it is a set of processes, it can be reduced to a single one – a goal process. Any artificial process serves as *PR* of some problem (Fig. 2). Then the search for a process that affords a design is equivalent to the search of a goal problem with a goal process within its scheme. Let us test for the part of goal problem the design one (*DPr*). Writing down its scheme (3),

$$DPr=\ <<SD><SP,\ DPR>> \tag{3}$$

we get evidence that this problem in formal sense is unsolvable (pseudoproblem): a priori determination of its goal process (design process, *DPR*) on object is impossible, while a posteriori determination is practically useless due to uniqueness of each *DPR*. But if a design is possible, its *DPR* also exists. The only eligible way of such existence is physical *DPR* synthesis in real time (concurrently with product design generation). But where do pseudoproblems come from?

There is a class of problems, realization of which yields along with the target result some side results. Side result is impossible without generation of the main one. This fact is sometimes overlooked, and just upon a side result, the need of which had arisen, a separate problem is stated. Naturally, such a “problem” cannot have an adequate solution – pseudoproblem owes its result to solution of the goal problem.

A similar case has turned out in designing as well. The required product design is provided by *DPR* and outside this process product design synthesis is impossible. Hence, the goal problem (*GPr*) in designing should be, and really is, the problem of *DPR* synthesis. The main result of the latter is physical *DPR*, side result is a product design. Design practice confirms this: it is easy to see that just *DPR* synthesis problem (4) is realized there by default with establishing for this an appropriate process: call it *meta DPR* (*mDPR*). The content of implicit to date *mDPR* has always been *DPR* synthesis through its model interpretation – implicit previously and explicit today.

$$GPr= <<SD><SP, mDPR>> \tag{4}$$

Thus, the part of *GPr*, realization of which yields a product design, is performed by the problem of real time physical *DPR* synthesis (4). *DPr* resultant process – *mDPR* – is the base process of any design event. *mDPR* automation is thus rendered design automation. Yet prior to cast for the part of *mDPR* subject the diprocessor C^H (to automate *mDPR*), *DPR* synthesis should be efficiently systematized.

4. Design process synthesis systematization

4.1 Estimated figures of the systematization

To compare different approaches to *DPR* synthesis systematization and estimate gained effect, we distinguish three appraisal adjectives: 1) the *level of systematization* (in ascending order: trial and error method – zero level, *DPR* model interpretation – the first level, *DPR* quasi-design realization – the second level); 2) *design goal volume* the synthesized *DPR* should produce (for instance, in Pahl and Beitz model [6] – those are conceptual, embodiment, or/and detailed designs, while in axiomatic design theory [11] – functional design and product construction design); 3) direction within design stages, in which trial and error method is pushed-out from *DPR* synthesis (usually it takes place from the late to early stages, i.e. from right to left or RL direction; there are no LR samples yet). Today’s design practice is featured by coexistence of the first and second levels of systematization.

4.2 The first systematization level

This level – *DPR* synthesis by its model interpretation – retains in full conventions of manual design, except that *DPR* models became explicit (“how little they can explain and how much we have to add by our imagination to the models”, [1]). Human is a sole interpreter of the model, and the number of its interpretations may exceed the number of interpreters. Low abstraction of *DPR* model blocks designer’s creativity; as the abstraction grew, *DPR* synthesis systematization goes down to the trial and error method. Canonic example of the first level systematization is the Pahl and Beitz *DPR* model [6] where product design structuring, *DPR* structuring, and design progress concept (stepwise reduction of abstraction grades in design description) are inherited from analysis. New *DPR* models (C-K theory [5], for instance) do not change the essence of its synthesis systematization – interpretation of these models by a designer and inefficient use of computer in the role of a tool on demand.

4.3 The second level of *DPR* synthesis

This level of *DPR* synthesis – by realization of *DPR* quasidesign – is represented by axiomatic design theory [11]. Prefix “quasi” here is motivated by the fact that *DPR* design, as design problem solution, is impossible for any volume of design goal; suffix “design” denotes the distinctive feature of the level – explicit realization of *DPR* fragment description. This fragment is intended for design of product construction and uses design of product structure (functions), which should be already available to the moment. Splitting a design into functions and construction is more appropriate for *DPR* synthesis at this level than analysis-rooted levels of its abstraction in the Pahl and Beitz model [6].

However, it is hardly possible to extend this method of partial *DPR* synthesis from designing a construction to functions (structure) designing (to shift the trial and error method by one stage to the left). At the same time, the most efficient solutions for construction design may turn unrealizable because of impossibility to make changes into already in existence but in given design event inactive design of functionality.

Thus, both levels of *DPR* design systematization use RL as working direction, about which there is no clarity where and how effectively it would be over. For both levels, the volume of design goal is the final stage of designing. Design representation is purely human-oriented. The field of such systematization use is redesign of already existent products (the first level) or optimization already available decisions in product construction (the second level). Finally, in the absence of a prototype or analogue for the product under design, both schemes of *DPR* synthesis are unworkable. There is an obvious need for a more efficient systematization.

4.4 The third level of systematization

We confirm the two embedded from the left to right *aspects* of product design: *dynamics* design (functions and structure) and *statics* design (construction). Due to evolutionary shaping of each aspect, the aspect design has initial, final and intermediate states, which are called *maturity levels* (*ML*). For short, structure design is denoted as *sD*, and construction design – as *cD*.

Set now the third level characteristics: 1) systematization vector – LR; 2) design goal volume for which *DPR* is synthesized – complete product design; 3) synthesis technique – iterations of *DPR* quantum generation. At this level, we combine basic concepts characteristic to both previous levels: the concept of fragment-by-fragment synthesis of complete *DPR* (the first level), and concept of *DPR* fragment synthesis by realization of its quasidesign (the second level of systematization). Fragment synthesis must be explicit (which is possible when a fragment is close to *DPR* quantum) and construction of complete *DPR* should be regular and continuous.

The necessary conditions for the third level shaping include the choice of adequate design progress concept and consistent with it a regular and in-depth product design structuring (till minimal maturity levels) suitable for *DPR* synthesis out of its quantum. But conceiving of any *DPR* quasidesign is attended with the search for design problem solution. So we begin the third level construction with attempts to get such a solution through sequential elimination of *DPr's* insolubility factors, on the one hand, and decreasing of design volume goal, on the other.

4.5 Design technology inevitable change

Having laid down the criteria of efficient *DPR* synthesis systematization (but actually – design systematization), apply now to one of the design science divisions – theory, technology or methodology – for assets. The mission of the theory is to generate, justify and upgrade designs of adequate (at the point in time) design technology; methodology comes to available technology to reveal the points of its efficiency increase and accomplish its rationalization and optimization; hence, our addressee is design technology.

Technology is rarely made explicit in discussions, today it is in the shadow of methodology – a Flying Dutchman of design science. Broad interpretation of methodology, attribution of extra functions to it and groundless hopes for their realization have made methodology a central part of design science. As a result, methodology has proven to run the complete design science household and tries on the functions of both design technology and theory.

Coming back to *DPR* synthesis that needs for quantization of design representation, regular synthesis of *DPR* quantum and such synthesis iterations, we can observe that current design technology does not have the appropriate support tools. Inherently it remains to be a technology of implicit manual design, whereas *DPR* synthesis of the third level assumes substantial increase of work for computer. Methodology unable neither to change technology nor offer a trend of its changes; optimizing some current technology, it by definition serves as apologist and preserving agent of this technology.

To remove technology from a shadow and make it a subject-matter of discussions and conversion, there is a need for its identification mechanism. The latter will provide the ability to analyze this or that design technology and reveal the factors of its inadequacy. Therefore our next step is adequacy criteria formation and appropriate technology construction.

5. Design technology identification

5.1 Asynchronous and synchronous identification

Entitative design technology in no way makes oneself known before its *asynchronous* identification – design support system construction. However such late identification does not permit to assess technology prior to its implementation. There is a necessity for a mechanism of synchronous (at any time) technology identification. In its absence, monitoring and analysis of technology changes or its purposeful adjustment are impossible. A proposed mechanism of synchronous technology identification consists in the following.

In Fig. 3, a scheme of design event goal problem together with elements of its statement is presented.

$$\begin{array}{r} \Rightarrow O^P \\ \boldsymbol{GPr} = <<\boldsymbol{SP}><\boldsymbol{SD}, \boldsymbol{mDPR}>> \\ I^D \Uparrow \quad \Uparrow I^P \end{array}$$

Figure 3. Design event goal problem scheme

Here I^D stands for a product design representation, the part of I^P is performed by a *DPR* model or *DPR* quasidesign for some design goal volume. Values of I^D, I^P and O^P specify particular statement of *GPr*. In turn, the way of design representation is predetermined by a selected design progress concept (*DPC*). Some examples of available to date *DPC*: successive abstract reduction in design description [6], evolution of population [12], random search [5], evolution of individual [10]. *DPC*, I^D and I^P are named *key triad* of a design event or its datums. Datums values identify design technology in a synchronous mode.

5.2 Current design technology analysis

Thus, we have the central problem of design event – *DPR* synthesis – and two current design technologies, reflecting the levels of *DPR* synthesis systematization. Let us undertake a brief analysis of both with the help of the just introduced identification mechanism. To this end, we determine their datums values.

Design progress concept used at the first level goes back to experience of [6] – successive reduction of abstract level in a prototype description associated with the required product. Design representation uses here the schemes of hierarchical decomposition of a product structure adopted in analysis and didactics (system, subsystems, assemblies, organs, parts). *mDPR* subject input (I^P) is this or that *DPR* model; *mDPR* object input (I^D) – designer's knowledge and experience. The inputs of each synthesized *DPR* fragment are the same in both technologies – current state of design and a need description with the list of requirements. Let us map this into the goal problem statement – *GPr1*:

needs & requirements=I^D *I^P=current design state (in abstract or actual)*

O^P=physical DPR (and product design)

$\Uparrow$

GPr1=<<SP><SD, mDPR>>

designer's knowledge=I^D *I^P=DPR model*

Figure 4. Canonical goal problem statement

Such datums values of the first level technology bring to the following summary. This (*DPR* synthesis) technology generalizes the experience of manual design. Its definitions, representations, methods, procedures, models, data structures and, in addition, design language are oriented mostly to informal processor (*H*). Designate such design technology as *H*-technology. The latter evidently has exhausted the sources of its development and less and less meets the required purposes.

The main distinction of the second level technology [11] is the value of *mDPR* subject input (I^P): it is not now a *DPR* model but *DPR* design intended for product construction generation (availability of functionality design is presupposed). Meanwhile, the values of two other datums do not support the outlined trend of technology renovation: instead of design progress concept, there is an intricate route of its formation, drawn up of axioms; two-body product design representation – by functionality and construction designs – is not enough synthesis-oriented and closes, in our view, most advanced ways of technology development, remaining it as optimization technology of available product construction.

Now the general conclusion on technologies of the two levels can be drawn up: in fact, those are *H*-technologies not considerably adapted to their computer-aided realization. *H*-technologies have too little of "synthesis genes" but in plenty of "analysis genes". The loss of *H*-technology adequacy is testified by impossibility of structure synthesis problem solution and disability to meet new challenges – critical increase of *DPR* complexity, needs for its continuous visualization and proper management.

5.3 From inadequacy factors to adequacy criterion

In the light of increasing complexity of new products, the main factor of the current design technology inadequacy may be articulated as a contradiction between *H*-monotechnology and two different types of processors taking part in its realization – *H* and *C*. Designing without computer is unreal now, but *C* is forced on completely alien to it design technology.

Transition to the third level of *DPR* synthesis systematization assumes a partnership between *H* and *C*, i.e. rejection of a losing computer status "a tool on call". Such relations are feasible if to take opposite courses: concurrently upgrade computer intelligence and adapt for it a design technology. As *C*-oriented technology is impossible, the sole decision is a *parity* or $\mathcal{P}$-technology of designing equally "understandable" to both types of processors (*P*) – *H* and *C*. Technology parity becomes the criterion of its adequacy and precondition of design automation. First and foremost, $\mathcal{P}$-technology signifies processor-parity values of its datums: design progress concept, design representation and *mDPR* inputs. Let us find them.

6. *P*-independent design technology

6.1 Common *P*-independent design progress concept

A design does not emerge in one go, and this or that *DPC* (Section 5.1) is inevitably realized in every design event. *DPC* presets, explicitly or implicitly, a sequence of design states, which are denoted as *maturity levels* (*ML*), and the way of transition from ML_i to ML_{i-1}, $i=1,2,\ldots$ Complete design event produces four outcomes: product dynamics design (structure design or metadesign), statics design (construction or product design), product operation environment design, and *DPR* design (if recorded). A common *DPC* for all these designs is necessary.

The ordered set of ML_i is called *diachronic* or "historical" design structure (*dh*-structure). In *H*-technology, the number of design maturity levels has to preserve their visibility and usually is not large that makes *dh*-structure trivial (conceptual, embodiment and detailed designs [13]). *P*-technology allows a sufficient number of design states to ensure its incremental synthesis. Instantiation of each member of *dh*-structure, i.e. representation of the current design state, is called its *synchronous* or *sh*-structure.

For the designs obtained, we take for the part of common *DPC* the evolution of individual or *evolutionary synthesis* treated as adaptation of operation environment design to the current state of the structure (construction) design, and adaptation of the current structure (construction) design state to a new state of operation environment design.

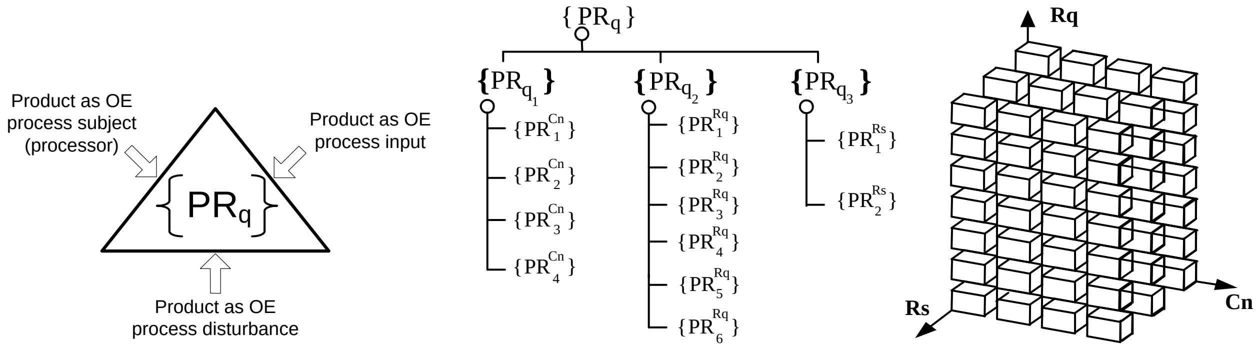
6.2 *P*-independent representation of product operation environment design

Product operation environment (*OE*) being designed concurrently with the product is presented by a family of processes ($\{PR_q\}$) the product will be merged in after physical implementation. Only three type of relations are possible between a product and some process from the family: a product can be (Fig. 5*a*) a process *input*, process *subject* (processor) or process *disturbance*. This gives grounds for dividing the family $\{PR_q\}$ into three sets: $\{PR_{q1}\}$, $\{PR_{q2}\}$ and $\{PR_{q3}\}$ correspondingly (Fig. 5*b*). Processes from $\{PR_{q1}\}$ specify for their subject operating *conditions* (Cn): processes from $\{PR_{q2}\}$ introduce *requirements* (Rq) to their inputs; processes from $\{PR_{q3}\}$ impose *constraints* (Cs) on their disturbance.

Next, nominate in each set from $\{PR_q\}$ a number of subsets (hierarchy in Fig. 5*b*). Thus, for example, requirements from $\{PR_{q2}\}$ are grouping into subsets, the number of which accords with numbers of product life-cycle stages; conditions from $\{PR_{q1}\}$ are divided in four groups reflecting the sequence of design complexity levels – conceptual, functional, technological and disturbance compensation level; the constraints generated by processes from $\{PR_{q3}\}$ shape two lists – minimizing intrinsic and extrinsic (ecological, for instance) product resource consumptions accordingly.

To construct diachronic representation for *OE* design, we consider three set vectors in Fig. 5*b* as space generator, construct out of those 3D *dh*-structure of *OE* design and call it &-*cube* (Fig. 6*c*). Coordinates of &-cube cells signify maturity level of this design; specific or synchronous *OE* design state is manifested by the value of $\{Cn\}\cup\{Rq\}\cup\{Cs\}$, i.e. by the suite of conditions, requirements and constraints accumulated in &-cube cell.

The contents of adjacent &-cube cells along scan trajectory are nesting related. The trajectory is selected by a designer and links initial and final cells.



a) *b)* *c)*

Figure 5. Environment design *dh*-structure construction

6.3 *P*-independent design representation

Simulating the product design evolution, let us construct a quasi-hierarchy of attainable maturity levels. (Within a quasi-hierarchy or *q*-hierarchy, descendants of the same node are linked by some relation as well.) The first level of *q*-hierarchy consists of four *MLs*, which are called *design goals*: *prototype* (PRT), *market version* of the product (MKV), *manufactured version* (MFV), and *released version* (ART) or artifact (Fig. 6).

Each goal attainment falls into four stages, shaping four *design subgoals*: *quasisystem* (qSYS), *system* (SYS), *quasiconstruction* (qCON) and *construction* (CON) accordingly. *q*SYS is a minimal set of design constituents capable to realize a declared aim-achievement concept; the stage of SYS deriving extends *q*SYS with control functions; qCON is resulted in spatial composition of product structure components; at the CON stage, there takes place the refinement of dimensions, materials, interfaces, tolerances, grades of finish, etc.

The nodes of *q*-hierarchy are linked by nesting relation ($\prec$) depicted in Figure 6 by arrow: PRT $\prec$ MKV $\prec$ MFV $\prec$ ART и qSYS $\prec$ SYS $\prec$ qCON $\prec$ CON. Coordinate index helps to distinguish between the listed macro*ML*s: ML_{km}, where k is design goal ($k=\overline{1,4}$), and m – subgoal ($m=\overline{1,4}$). Product design synthesis is associated with continuous determination of product operation process presented by its scheme (Section 2.1). Therefore, synchronous representation of each design maturity level uses *S*-tree [8].

6.4 The platform of designs

It is time to integrate *dh*-structures of product design and *OE* design into a single representation – call it *unified design platform*. The latter is obtained by substitution *OE* design *dh*-structure (&-cube} instead of each terminal node in product design *dh*-structure (Fig. 6). Thus, &-cube cells will accumulate corresponding maturity levels of *OE* design and product designs (dynamics and statics).

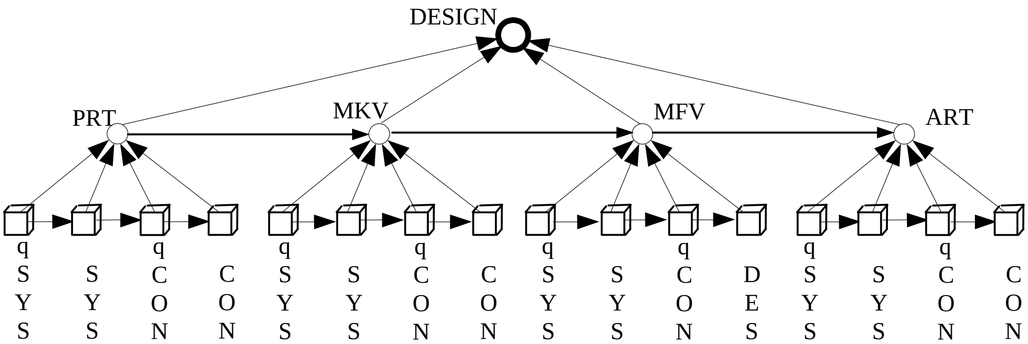


Figure 6. Design platform as integration of product and environment design *dh*-structures

7. Meta design process determination in *P*-independent design technology

7.1 *mDPR* input building

The way of complete physical *DPR* synthesis outlined for the third level of systematization (Section 4.4) consists in regular and explicit generation of physical *DPR* quantum in every scanned &-cube cell along the marked trajectory. Such generation is based on realization of quasi *DPR* (or *qDPR*) quantum design. In this case, *qDPR* is a design process for some volume of design goal presented by its synthesis part without analysis (feedback) part. We shall search for the *qDPR* quantum design through attempts to elicit formal solution to the virtual design problem (*DPr*).

DPr has two factors of insolubility: 1) the lack of initial data to produce the required design *ML*; 2) *DPr* is not one but amalgam (inseparable composition) of design problems. Then the experiment with its solving will consist in elimination of its insolubility factors. We shall proceed, on the one hand, with splitting *DPr* into component problems, and on the other hand – with reducing design goal volume. Incompleteness of initial data is surmounted by *DPr* statement in the first cell of design platform. Starting set of initial data here is quite sufficient to obtain the next maturity level of design. When this happens, it becomes clear how to replenish original data to continue generation of the next *ML* in turn, and so on.

As for the second *DPr* insolubility factor, there are three designs being concurrently produced: product structure design (*sD*), construction design (*cD*), and *OE* design (*eD*). So, the matter is that we have not one but composition of DPr_i, $i=\overline{1,3}$, which prevents from describing a solution to all the triple by one procedure. Besides, each DPr_i is again represented by a couple of problems – the synthesis problem and analysis problem (with resultant synthesis process, *SPR*, and analysis process, *APR*, correspondingly). In terms of resultant processes, original *DPr* may be presented by the following triple of pairs: (SPR_1, APR_1), (SPR_2, APR_2), (SPR_3, APR_3). In accord with LR direction of systematization level at hand, we begin to solve (to determine resultant process on object and subject) the very left problem – the problem of product structure synthesis.

As *OE* design problem (DPr_3) is a pseudoproblem with regard to DPr_1 (structure designing), APR^{eD} coincide with APR^{sD}, and SPR^{eD} is a part (supply agent) of SPR^{sD}. This enables to build *mDPR* input – SPR^{sD} design (Fig. 7), i.e. *qDPR* quantum design intended for generation of product structure design (*sD*) increment.

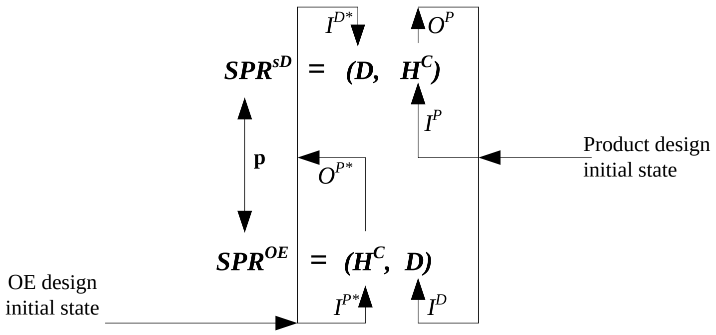


Figure 7. The unified quasi *DPR* quantum design

An essential detail: SPR^{sD} is the resulting process of the structure synthesis problem, which is also insoluble (pseudoproblem of the second kind). But we already have for it a goal problem – the problem of product operation process determination, tackling of which is outlined in [9].

7.2 Meta design process' object and subject

Now there is nothing to prevent us from deriving a solution to the goal problem of design event (real time physical *DPR* synthesis), which consists in determination of its scheme resulting process (*mDPR*) on subject (*P*) and object (*D*). Since our objective is design automation, we assign for the part of $P \in mDPR$ the diprocessor of C^H type (Section 2.2). Thereby, management of *DPR* synthesis is committed to computer.

The main functions of $D \in mDPR$ are design platform cells scanning and synthesis in each cell the physical *DPR* quantum through *qDPR* quantum design realization, combining the output with a design of analyzing part and implementation of the latter. If the *DPR* quantum is resulted in acceptable state of design *ML*, the next cell along the trajectory becomes active. This cell takes over from the previous one a new state of product design and the current state of *OE* design, whereupon the next *DPR* quantum in turn is synthesized here. In terms of *H*-technology, design problem is solved over again in every cell of trajectory but for a new design maturity level. Thus, the side result of *DPR* synthesis is a new state of product design. If this state does not assessed as acceptable, the *DPR* quantum will be resynthesized in the active or some previous cell.

Complete DPR^{sD} is obtained by iterations of DPR^{sD} quantum synthesis, the number of which is equal to the number of cells in scan trajectory of the design platform. In order to obtain a complete product *DPR*, the way of DPR^{sD} quantum synthesis is used for DPR^{cD} quantum synthesis as well; this is possible due to the unified synthesis-oriented *sh*-representation for designs of product dynamics and product statics.

Goal problem statement within the scope of $\mathcal{P}$-technology (*GPr2*) takes on a form as in Figure 8.

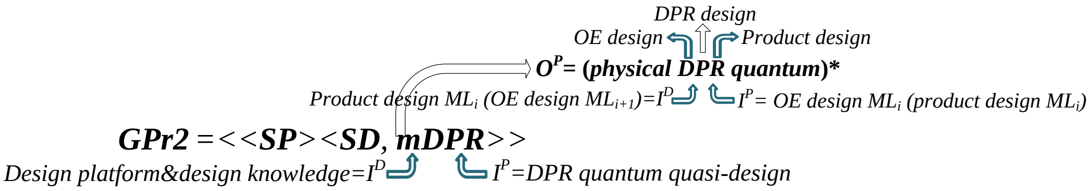


Figure 8. Design event goal problem statement

8. Closure

Two overripe, in our view, contradictions determine not only design automation state but also the state of affairs within design science in whole. Designing is a synthesis-oriented discipline, while the current design technology remains analysis-tailored. While the role of computer in analysis is "a tool on call", in complex synthesis a computer tends to be a manager of the process. But the current design technology prevents such computer's specialization.

Design monotechnology (*H*-technology) and two different types of processors (*H* and *C*) engaged in its realization is the essence of the second contradiction. Within *H*-technology, it is impossible not only to implement design automation, but even to define it. Computer is forced into foreign to its nature technology and cannot digress from the part of "a tool on call" in concept. Design computerization (CAD) is a part of design automation, but it cannot neither substitute nor grow into it.

The issue of design automation is the issue of design technology change. Axiomatic design theory [11] became an attempt to make a breach in the *H*-technology of design; however overwhelming number of results, models and theories remain apologetic with respect to *H*-technology. The latter, in the interim, has essentially hooped design science progress. In the light of rising product complexity, *H*-technology limitations continue to increase, and design community’s inconstancy in trying to relax the hoop (excursus to systems theory [4], protocols [2], discussions on design problem [3], “prescriptive/descriptive” mode of coping with the flow of *DPR* models, new and new mantras – “design for X”, “design for six sigma”, “design as interdiscipline”, etc.) would rather constrict it even more tightly.

To pave the way for a new technology, we at first have changed semi-intuitive definitions of the key design event notions for their exact equivalents. Next, the key process and key problem associated with a design event were distinguished. The problem of real time physical design process synthesis is actually the discipline making problem. (Any mature discipline is an interdiscipline, but there is no interdiscipline prior the discipline.) The search for a solution to it is the space for design theories becoming, while its result deriving – the space for technologies development.

Within $\mathbb{P}$-technology, *DPR* complexity loses its topicality, structure synthesis problem gets the solution, and meta *DPR* comes to the focus of research. In addition, designing here does not need for its automation – it (Computer Urged Design) merely could not be different. Design system mockup realizing this technology has been constructed to validate its underlying principles. Next in turn, development of the new generation design system (instead of numerous design support systems in *H*-technology) should be started.

9. References and Citations

[1] Andreasen, M. (1998) *The role of artifact theories in design*, In Proceedings of the Workshop Universal Design Theory, Karlsruhe, Germany, pp 57-71.

[2] Cross, N. (1994) *Engineering Design Methods* (*Strategies for Product Design*, 2nd edn.), London: Wiley.

[3] Dorst, C.H. (2006) *Design Problems and Design Paradoxes*, Design issues, vol. 22, No. 3, pp. 4-17.

[4] Eder, E. and Hosnedl, S. (2008) *Design Engineering* (*A manual for enhanced creativity*) CRC Press.

[5] Hatchuel, A. and Weil, B. (2003) *A new approach of innovative design: an introduction to C-K theory*, In Proceedings of ICED’03, Stockholm (on CD).

[6] Pahl, G. and Beitz, W. (1983) *Engineering Design*, Springer, Berlin-Heidelberg.

[7] Polya, G. (1965) *Mathematical discovery*, V. II, John Wiley&Sons, New-York-London.

[8] Sedenkov, V. (2010) *Continuous process theory: fundamentals and applications*, In Proceedings of the 1st Int. Conf. on Modeling and Management Engineering Processes, Cambridge, UK, pp. 113-124.

[9] Sedenkov, V. (2010) *New Design Paradigm: Shaping and Employment*, In Handbook of Research on Trends in Product Design and Development, Chapter 2, ed. by Silva, A., Simoes, R., Hershey-New York, pp. 18-38.

[10] Sedenkov, V. (2009) *The main mystifications ingrained in engineering design*, In Proceedings of ICED'09, Vol. 2, Stanford, CA, US, pp. 215-227.

[11] Suh, N. P. (2001) *Axiomatic Design*, New York: Oxford University Press.

[12] Vajna, S., Clement, S., Jordan, A., and Bercsey, T. (2005) *The autogenetic design theory: an evolutionary view on the design process*, Journal of Engineering Design, v. 16(4), pp. 423-440.

[13] VDI2221 (1987) *Systematic Approach to the Design of Technical Systems and Products*, Beuth Verlag GmbH (Ed), Translation of the German edition 11/1986, VDI Verlag GmbH Düsseldorf.