The Third International Conference on Design Creativity (3rd ICDC)
Bangalore, India, 12th-14th January 2015

A NEW APPROACH TO CONCEPTUAL DESIGN SYNTHESIS OF SENSORS

B. Sarkar[1], A. Chakrabarti[1] and G. K. Ananthasuresh[2]

[1]Centre for Product Design and Manufacturing, Indian Institute of Science, Bangalore, India
[2]Department of Mechanical Engineering, Indian Institute of Science, Bangalore, India

Abstract: Time to market, demand for variety and cost of exploration of the solution space, drive designers to use computer aided synthesis tools. Such tools, generally, generate a large number of solutions but rarely help evaluate and select the promising ones. Designers have to wait for embodiment of these designs before they can quantitatively evaluate a solution. What is required, is an initial rough estimate of the worth of a design before committing to its embodiment, so that the designer can decide if the solution is worthy of further exploration. The paper introduces a new approach to conceptual design synthesis using conceptual structures and signal-flow graph algebra. It is based on a special form of the SAPPhIRE model of causality called SAPPhIRE-lite. It uses conceptual structures for early embodiment and comes up with quantitative system equations. It generates solution principles with all associated feedbacks. This paper focuses on the synthesis aspects of the research.

Keywords: *conceptual design synthesis, flow graph, sapphire, causality*

1. Introduction

Conceptual design is a phase in the process of designing, when solution principles are developed to meet the desired functions (Pahl & Beitz, 1996). Synthesis of conceptual designs help in coming up with a variety of solutions. Various researchers have come up with multiple ways of concept generation. An overview of the computer based design synthesis research is available in (Chakrabarti, Shea, Stone, Cagan, Campbell, Hernandez & Wood, 2011). One such approach is conceptual design synthesis using physical laws and effects as building blocks (Chakrabarti & Bligh, 1996; Zavbi & Duhovnik, 2000; Srinivasan & Chakrabarti, 2009). SAPPhIRE model (Chakrabarti, Sarkar, Leelavathamma & Nataraju, 2005) which uses physical laws and effects is the focus of this paper. It has been shown in earlier literature that SAPPhIRE model of causality can be used for analysis and synthesis of conceptual designs (Srinivasan & Chakrabarti, 2009). Further, it was shown in (Chakrabarti et al., 2013) that SAPPhIRE model can be used to capture different views of function within a generic model of designing. Function exists at the various levels of abstraction of the SAPPhIRE model. Sensors are a special class of devices; they can be represented as a function in the effect level of the SAPPhIRE model.

This paper presents *SAPPhIRE-lite* -- a special form of SAPPhIRE model -- useful for *functions in the effect level*. SAPPhIRE-lite instances are used as building blocks to synthesize sensor designs. A software support has been developed for the same. The software also predicts possible *side-effects*. Quantification in the form of algebraic equations and component descriptions in the form of attributes are also proposed by the support. The support can also predict if a component needs to be

decomposed into sub-components. The paper first introduces the proposed model, and then illustrates the use of the model in synthesizing solutions. It touches upon various aspects of the synthesis process and the algorithms used by the support tool. The other aspects of the model, quantification and side-effect detection, are not discussed in this paper.

2. Literature survey

Computer aided conceptual design literature propose two broad approaches; on the one extreme is the case based approach (Prabhakar & Goel, 1998) and on the other extreme is the synthesis based approach. In case based approach, modification of existing solutions to meet new functionalities is practiced. This leads to biasing towards familiar solution principles (Schmitt, 1993) and as such lacks variety. Synthesis based approach falls into two categories: function-based synthesis and grammar-based synthesis. The function-based synthesis approach first comes up with a functional model then uses this model to come up with solutions. In grammar-based synthesis, the focus is to come up with a formal grammar which has a set of rules that acts on a design vocabulary and transforms the initial design into a wide variety of new designs.

A way to come up with a functional model is to use physical laws and effects as basic building blocks (Chakrabarti & Bligh, 1996; Zavbi & Duhovnik, 2000; Chakrabarti, 2001; Rihtarsic et al., 2012). This often generates solution principles with a lot more variety and uniqueness; but occasionally, one might end up with a number vague solution principle. One reason for this is improper representation of physical quantities (Chen et al., 2013). There has been some work in this direction to associate a flexible set of attributes to physical quantities (Chen et al., 2013). Physical laws and effects have been used in TRIZ (Yan, Zanni-Merk, Cavallucci & Collet, 2014) methodology in the form of a catalogue, to support innovative thinking, but they are not directly used to generate concepts. Tools like Modelica (Mattsson, Elmqvist & Otter, 1998), supports qualitative analysis of conceptual designs, but it can not synthesis designs. Hence, there is no such tool which will synthesize solution principles and then analyze them quantitatively.

Side-effects are defined as the unintended effects that affect the intended working of the system (Chakrabarti et al., 1997). Prediction of side-effects has been studied by researchers e.g. (Chakrabarti et al., 2011). The work presented in this paper addresses the issues experienced with the synthesis approach and enhances it with context information, side-effects and quantification.

3. SAPPhIRE-lite

SAPPhIRE model of causality (Chakrabarti et al., 2005) uses physical laws and effects as the basis for explaining how systems work to achieve high level functions. It has a set of inputs and organs which activates the physical effects to produce a set of outputs. It assumes that organs should not change during the time interval when the physical effect takes place. However, in reality, due to feedbacks, the organs also tend to change. So *SAPPhIRE-lite* – a special form of SAPPhIRE – has been tailored to capture this scenario. The model focuses into the input, organ and effect layer of the SAPPhIRE model and generalizes them to describe complex scenarios involving feedbacks. It has changed the nomenclature and scope of some of the entities of the SAPPhIRE model to support quantification and capture complex scenarios.

The effect level of the SAPPhIRE model is similar to input-output network as in network theory. So SAPPhIRE-lite model uses network theory concepts; it captures causality along with input-output relations and supports quantification. It uses a variant of signal flow graph called *switching flow graph (SFG)* (Smedley & Cuk, 1994) to quantitatively relate the input quantities to the output quantities, depending upon favourable input conditions. SFG is an input-output relation that is controlled by logic; only when the desired logical conditions are met, the output is produced as a function of the inputs. With this change, a computer algorithm can understand the scenario when a relation is valid. It also uses *Kirchhoff's current law (KCL)* (Kuo & Golnaraghi, 2003) for algebraic addition of similar quantities, thus allowing feedbacks to be incorporated. The model does not incorporate *Kirchhoff's Voltage law*; thus it uses a *weaker* form of signal flow graph algebra; hence it can be applied to graphs without considering flow and effort variables. Signal flow graph is used to capture relationships among attributes without considering as to how the relationship is *achieved or*

maintained. Signal flow have been used by other researchers too. SAPPhIRE-lite uses a subset of Nagel's (Nagel, Vucovich, Stone & Daniel, 2007) signal flow grammar.

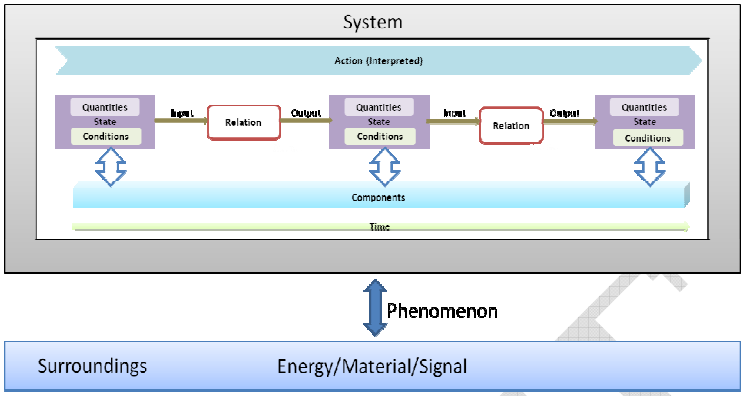

Figure 1: SAPPhIRE-lite model instances are used to capture the working of a system.

The model is pictorially described in the [Figure 1]. It has multiple levels of abstraction. It captures the state of the system over a span of time. The state of the system is described by a set of measurable quantities and conditions belonging to the underlying level of components and parts. Physical phenomena are the underlying exchanges of material, energy or signal between the system and its surroundings that cause physical effects to take place. Physical effects in turn lead to state-changes. These state-changes are interpreted as actions. Physical effects get activated when favourable input conditions are met and required inputs are available. We can describe a system's causal behaviour using this model as a pictorial graph. We can also carry out mathematical analysis of its state variables. The details of the various elements of the model are given below.

3.1. Quantity

Quantity is at the very core of the model. More precisely, it is called a measurable quantity. In order to describe the state of a system we need a set of measurable quantities which are associated with material properties or with physical attributes. All physical laws deal with quantities. Some quantities have multiple attributes; for example: wave has amplitude, frequency and phase. Some quantities need special association with other quantities to describe themselves; for example: frequency of oscillation, frequency of an electro-magnetic wave. Some of the laws of physics are applicable to a quantity only in its specialized form; whereas, some laws are applicable in general; for example: Planck's relation is only applicable to frequency of electro-magnetic waves, but the relation between frequency and its time-period is applicable for any wave. So quantities exhibit a hierarchical inheritance property. The model uses an inheritance tree to keep track of specialized quantities and their associated relations. Quantities are expressed in the model with the following set of attributes: name, description, parent-quantity, dimension, domain, order-of-tensor, and magnitude-direction tensor. SAPPhIRE-lite refers to inputs and quantifiable organs of the SAPPhIRE model as quantities.

3.2. Condition

A physical effect takes place when its desired inputs are met and favourable conditions are in place. A condition is expressed as a set of logical predicates. It is used to describe the context. The set of attributes that a condition can have is unlimited; but in general, a condition describes a situation, an event, functionality of a component, presence or absence of something, or a quantifiable or an abstract property. For example: Solid-Substance has mass, volume, density and form. When we say Hard Solid-Substance, it has mass, volume, density, form and hardness. So we see conditions also have some inheritance hierarchy. One can use an inheritance tree to form complex conditions and aggregate a set of properties with minimal effort. A condition should describe the under lying conceptual structure effectively using its set of attributes. SAPPhIRE-lite refers to logical type of organs of the SAPPhIRE model as conditions.

3.3. Relation

Quantities and conditions describe the state of a system. Physical effect causes change of state. Each physical effect has an associated conceptual structure (Chakrabarti A., 2004; see also Rihtarsic et al., 2012). Quantities belong to that of the conceptual structure. Physical effect captures the mapping function that relates the output quantities to the input quantities. It would be more appropriate to refer physical effect as a *relation*; thus generalizing it to capture any type of functional relationships across domains. SAPPhIRE-lite refers to physical effects of the SAPPhIRE model as relations. Each output quantity can be mathematically related to the set of input quantities. This allows SFG to be used for modelling relations. 1)The condition associated with a relation will capture the details about the conceptual structure required for the relation. There has to be a mapping of these conceptual structures to the underlying physical components. Conceptual structures are used as a layer of abstraction of the underlying physical components.

3.4. Component

SAPPhIRE-lite refers to the parts of the SAPPhIRE model as components. Components interact with their attributes; attributes form quantities and conditions for the state of the system. A component may embody one or more conceptual structures. A system may be decomposed into a hierarchy of components.

3.5. Action

Action is an interpretation of the state change exhibited by the system. Generally, the action level consists of a tree structure, where the top level action is decomposed into smaller actions (sub-actions). During the process of designing using top-down approach, often designers use the action tree to decompose the problem into manageable sub problems. Once the actions are decided, the corresponding state changes, physical effects, inputs, outputs, conditions, phenomena and parts gets selected. Like the action tree, the system and its surroundings can be also decomposed into a components-tree during analysis. The action tree and the components tree are closely related; a component might have multiple actions; and multiple components might be required to meet a single action.

3.6. Phenomenon

A phenomenon is the abstraction of the exchange of material, energy and signal between a system and its surroundings; e.g. heat exchange. It causes physical effects to take place. Since SAPPhIRE-lite is based on signal flow and it doesn't *explicitly* capture the flow of material or energy, it indirectly captures a phenomenon. The exchange of material and energy is captured in the form of inputs and outputs.

3.7. Model example

[Figure 2] shows a screenshot of the SAPPhIRE-lite model of a thermistor as captured by the tool developed for this purpose. In this example the component tree, formed by surroundings, system and thermistor, is shown in prominence; within each component the other elements of the model are captured; actions (A), quantities (Q), relations (R) are shown. The conceptual structure associated each relation is shown in the figure; it is also used as a name for the set of conditions associated with the relation. These set of attributes are not shown in the graph but are presented to the user as a lookup table by the tool. Similarly, quantities have attributes that are not shown in the figure. Quantities and relations are connected by the edges (QR-edge), thus forming the signal flow graph. Each relation's output edge has a mathematical function relating the output quantity to the input quantities. The QR-edges are all labelled, thus signal flow graph algebra can be used to form mathematical equations for the entire system. Phenomenon is not explicitly captured by the tool, even though the model has it.

Relations are mathematically represented using a-causal equations. The exact causal order is not important for solving the mathematical equations. But for human understanding, causality is important. So the model supports multiple relations within a set of quantities with the same

mathematical relation. For example: Ohm's law has the mathematical equation V - IR = 0. This relation is valid if we apply I and R as inputs and get V as the output or apply V and R as input and get I as output. Only the direction of edges changes if we see the signal flow graph for these relations. The model supports such type of cyclic relations. This leads to redundant system equations. In this example there are two cyclic relations ("Heat-Temp Def" and "Heat-Temp Def – Temp") which lead to a pair of redundant equations. The redundancy can be easily taken care of by an equation solver. The model in the figure looks significantly complex because of the feedback paths and some trivial conversion relations. One might hide some of them from the user, but they are required for mathematical consistency.

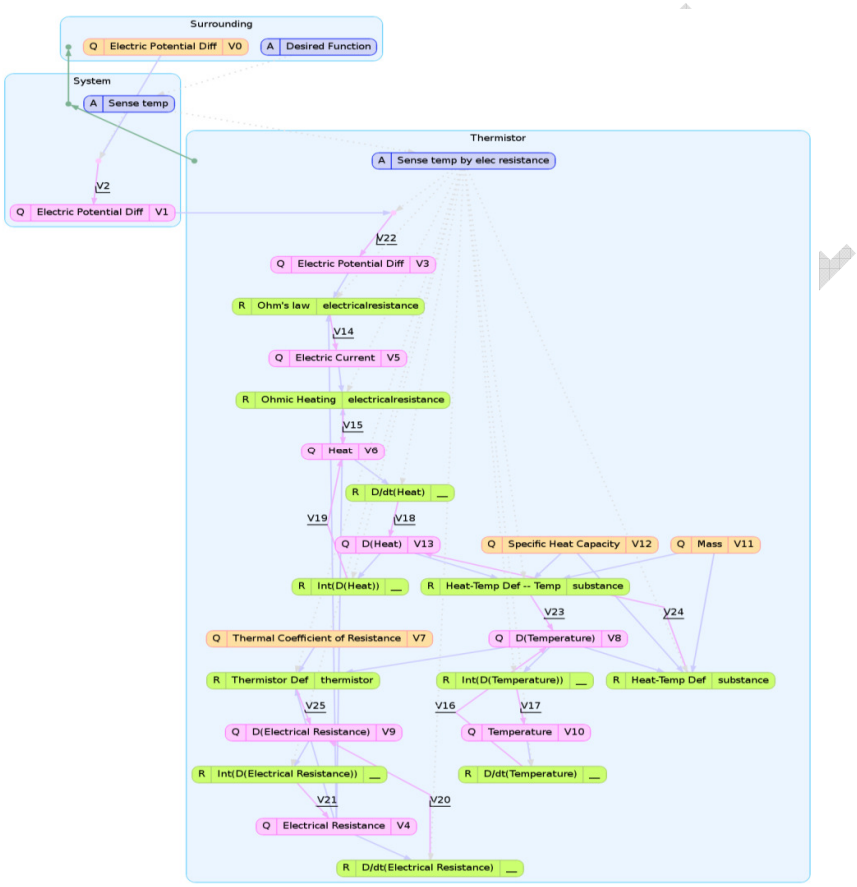

Figure 2: SAPPhIRE-lite model of Thermistor (Temperature dependent resistor). Quantities are indicated by the label 'Q' and relations by 'R'. The quantities and relations form a network.

4. Database

In order to synthesize sensors, a database comprising of SAPPhIRE-lite instances is maintained. For synthesis, information is not required from all the levels of the SAPPhIRE-lite model; only the inputs, outputs, conditions and relations are sufficient. SAPPhIRE-lite instances of known physical effects are populated into the database. Each physical effect has a specific conceptual structure associated with it. The information about the structure is also added to the database in the form of conditions. Physical effects which are just the function of time are incorporated into the database. Hence the database has only lumped parameter models of relations which can be expressed using ordinary differential equations.

The component tree has three types of nodes: root, leaf and intermediate nodes. Each node type exhibits a different view point. The view from a leaf node is just the component and its

surroundings; so it sees the system as a *2-tier* block [Figure 3]. Similarly, for an intermediate node, it has a set of components, an interconnection layer and surroundings; it is a *3-tier* block [Figure 3]. From the root node's view point, the system has a set of components and an interconnection layer; it is a *2.5-tier* block [Figure 3]. With these views in mind the relations in the database obeys some simple rules: (a) all outputs are in the same entity where the relation resides, (b) only the input quantities can come from across entity boundaries, (c) no inputs can come from peer component entities. These rules are intended to give a simple hierarchical view of the models created by the relations.

The database also stores the equations associated to the output quantities. As shown in [Figure 4], the quantities and relations form a *bi-partite* graph in the database if we consider the QR-edges between them.

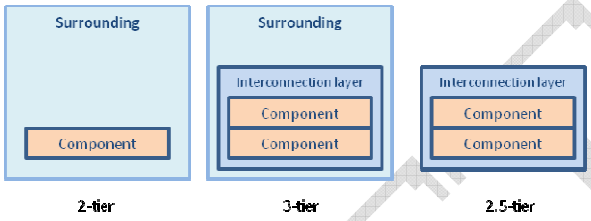

Figure 3: View points

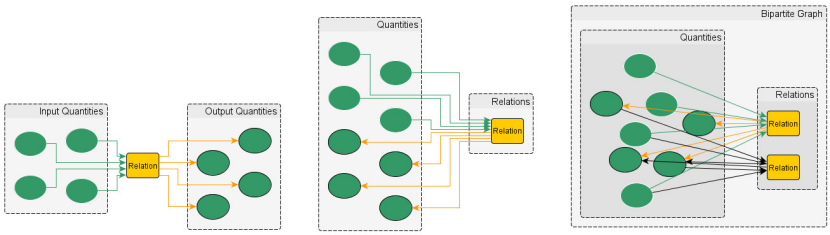

Figure 4: Relations map input quantities to output quantities. Quantities are only connected to relations. This forms a Bi-partite graph in the Database.

5. Synthesis

The database is used to synthesize new sensors. The user specifies the quantity that needs to be sensed; and also the quantity which will be the output. The synthesis algorithm uses a copy of the *bi-partite* graph of the database to generate all paths connecting the input quantity to the output quantity. The search algorithm uses *k-shortest path* algorithm (Eppstein, 1998) for this purpose. The search algorithm gives a set of paths connecting the input quantity to the output quantity through a series of relations and quantities. Since we have a bi-partite graph, each path will have alternating nodes of quantities and relations; each relation binds two quantities together, one as an input and the other as an output. The algorithm also keeps track of the length of each path. It produces a set of *bare solution principles* ranked with respect to the path length. It is termed as *bare solution principle* because the associated *feeding graphs* are not yet explored. The feeding graphs are required to produce the desired inputs to activate all the relations in the bare solution principle. This might give rise to feedback paths and might contribute as side-effects. For the generated solution graph ordinary differential equations (ODE) can be generated using the SFG algebra. Since the generated ODE might not have sufficient number of constraints to be solved by any equation solver, designer's interactions would be required.

6. Components

The algorithm does not explicitly come up with the components required for the implementation of the solution principle. But it makes a prescription for the components in terms of some conceptual structures. It suggests a set of boolean attributes in the form of *conditions* that would be required to be satisfied by the component that would exhibit the desired functionality. When there are contradictory requirements, it is an indication to split the component into sub-components. Often, splitting of

components is required to reduce the side-effects. Optimal component splitting is computationally a hard problem to solve (Buluc, 2013), but there can be heuristic algorithms to give good solutions. The current version of the software doesn't support splitting of components.

7. Software Design

A web based interactive software tool has been developed using the algorithm. It has a pictorial representation of the hyper-graph that models the sensor along with its quantities and relations. The database currently has about fifty relations and thirty quantities.

8. Synthesis Example

Consider the example of an accelerometer. The user specifies "acceleration" as input and "electric current" as the output. The algorithm generates a set of *bare solution principles* (BSP) and allows the user to select the interesting ones. The user can interactively select a few and later on get the detailed look at each of the potential candidates for further analysis. The tool groups the generated BSPs into some categories based on component structure, domain, length of chain etc. For a detailed look the user is presented with the solution principle along with the set of system equations. The tool also shows the user the set of relations which are vulnerable to side-effects. Since side-effect is beyond the scope of this paper it will not be discussed further. An example of a solution principle generated by the tool is given in [Figure]. The chained relations and quantities are expressed as the rows of the table. In this example acceleration is sensed by a body (Solid Substance) having mass and converted to body force, than to pressure which causes strain to a fibre optic cable with Bragg gratings. This has caused a shift to the Bragg wavelength of the fibre optic cable and this shift is sensed by electromagnetic wave (light wave) which has wavelength matching to that of the Bragg wavelength. The light wave is sensed by a semiconductor device having a PN junction and thus produces electric potential and hence electric current. The condition name is referring to the conceptual structure and the context which is required for the relation to be active.

Bare Solution Principle

Type	Name	Component	Condition Name
Quantity	Acceleration	comp3	
Relation	**Newton's 2nd Law of motion**	**comp3**	**Solid Substance**
Quantity	Force (Body)	comp3	
Relation	**Pressure Def**	**comp3**	**Solid Substance**
Quantity	Pressure	comp3	
Relation	**Reactive Force**	**comp3**	**Solid Substance**
Quantity	Stress - Normal	comp3	
Quantity	Stress	comp3	
Relation	Law of Elasticity	comp3	Solid Substance
Quantity	Strain	comp3	
Relation	Bragg wavelength shift	comp3	Fiber Bragg Grating
Quantity	Length - wave - bragg	comp3	
Relation	Notch Filter	comp3	Fiber Bragg Grating
Quantity	Frequency - EM wave	comp3	
Relation	EM Wave Energy	comp3	EM Wave
Quantity	Energy - Radiant	comp3	
Relation	Photo Voltaic Effect	comp3	PN junction
Quantity	Energy - Electric	comp3	
Relation	Elec Potential Def_1	comp3	Charge in Elec Field
Quantity	Electric Potential Diff	comp3	
Relation	Ohm's law	comp3	Electrical Resistance
Quantity	Electric Current	comp3	

Conditions

Condition Name	Attributes
Solid Substance	mass = true
	solid = true
Fiber Bragg Grating	bragg grating = true
	brag wavelength = true
	optical fibre = true
EM Wave	electromagnetic wave = true
PN junction	energy band gap = true
Charge in Elec Field	electric field = true
	charge moves = true
Electrical Resistance	finite electrical resistance = true

Figure 5: Synthesized accelerometer solution principle

9. Summary

A model – SAPPhIRE-lite -- has been proposed. It incorporates signal flow graph algebra to support quantification. The model has been used to capture physical effects and populate a database. A synthesis algorithm has also been proposed which uses the database to come up with new type of sensor designs. A software implementation is also available. The salient features of this approach are: (a) it can generate solution principles with feedbacks; (b) it captures all possible effects that are associated with the solution principle, thus it also captures all possible side-effects; (c) it supports quantitative analysis of the solution principle embodied using conceptual structures (which is not discussed in this paper). The limitations are: (a) it supports only lumped-parameter models; (b) it is difficult to come up with an embodiment of a component with the given set of conceptual structures. (c) Database population is often a difficult task.

10. References

Buluc, A. a. (2013). Recent Advances in Graph Partitioning. CoRR, abs/1311.3144.

Chakrabarti, A. (2004, March). A new approach to structure sharing. ASME JCISE, 4, 11-19.

Chakrabarti, A., & Bligh, T. P. (1996). An approach to functional synthesis of mechanical design concepts: theory, applications, and emerging research issues. AI EDAM, 10(4), 313-331.

Chakrabarti, A., Johnson, A., & Kiriyama, T. (1997). An approach to automated synthesis of solution principles for micro-sensor designs. Schriftenreihe WDK, 125-128.

Chakrabarti, A., Regno, R., Sarkar, B., & Srinivasan, V. (2011). A Tool for Automated Synthesis and Side-Effects Detection in Sensor Designs. Research into Design-Supporting Sustainable Product Development (ICoRD'11), 581-589.

Chakrabarti, A., Sarkar, P., Leelavathamma, B., & Nataraju, B. (2005). A functional representation for aiding biomimetic and artificial inspiration of new ideas. AI-EDAM, 19(2), 113-132.

Chakrabarti, A., Shea, K., Stone, R., Cagan, J., Campbell, M., Hernandez, N. V., & Wood, K. L. (2011). Computer-based design synthesis research: an overview. Journal of Computing and Information Science in Engineering, 11(2), 021003

Chakrabarti, A., Srinivasan, V., Ranjan, B., & Lindemann, U. (2013). A case for multiple views of function in design based on a common definition. AI-EDAM, 27(03), 271-279.

Chen, Y., Liu, Z., Huang, J., Zhang, Z., & others. (2013). A multi-agent based framework for multi-disciplinary conceptual design synthesis. DS 75-6: Proceedings of the 19th International Conference on Engineering Design (ICED13), Design for Harmonies, Vol. 6: Design Information and Knowledge, Seoul, Korea, 19-22.08. 2013.

Eppstein, D. (1998). Finding the k shortest paths. SIAM Journal on computing, 28(2), 652-673.

Kuo, B. C., & Golnaraghi, M. F. (2003). Automatic control systems (Vol. 4). John Wiley \& Sons New York.

Mattsson, S. E., Elmqvist, H., & Otter, M. (1998). Physical system modeling with Modelica. Control Engineering Practice, 6(4), 501-510.

Nagel Robert, L., Vucovich Jayson, P., Stone Robert, B., & McAdams Daniel, A. (2007). Signal Flow Grammar From the Functional Basis. Guidelines for a Decision Support Method Adapted to NPD Processes.

Pahl, G., & Beitz, W. (1996). Engineering Design - a systematic approach. New York: Springer.

Prabhakar, S., & Goel, A. K. (1998). Functional modeling for enabling adaptive design of devices for new environments. Artificial intelligence in Engineering, 12(4), 417-444.

Rihtarsic, J., Zavbi, R., & Duhovnik, J. (2012). Application of wirk elements for the synthesis of alternative conceptual solutions. Research in Engineering Design, 23(3), 219-234.

Schmitt, G. (1993). Case-based design and creativity. Automation in construction, 2(1), 11-19.

Smedley, K., & Cuk, S. (1994). Switching flow-graph nonlinear modeling technique. Power Electronics, IEEE Transactions on, 9(4), 405-413.

Srinivasan, V., & Chakrabarti, A. (2009). Sapphire--an Approach to Analysis and Synthesis. DS 58-2: Proceedings of ICED 09, the 17th International Conference on Engineering Design, Vol. 2, Design Theory and Research Methodology, Palo Alto, CA, USA, 24.-27.08. 2009, (pp. 417-428).

Yan, W., Zanni-Merk, C., Cavallucci, D., & Collet, P. (2014). An ontology-based approach for using physical effects in inventive design. Engineering Applications of Artificial Intelligence, 32, 21-36.

Zavbi, R., & Duhovnik, J. (2000). Conceptual design of technical systems using functions and physical laws. AI EDAM, 14(1), 69-83.