Imagining Knowledge, a Formal Account of Design

Lex Hendriks and Akin Kazakci

ILLC University of Amsterdam, CGS Mines ParisTech
a.hendriks@uva.nl, osmanakin@gmail.com

Abstract

Design, as in designing artifacts like cars or computer programs, is one of those aspects of rational agency hardly even mentioned in traditional logical theory. As an engineering discipline, design obviously involves reasoning but seems to depend much more on a mix of factual knowledge, experimenting and imagination.

We will present a formal framework for the dynamic interplay between knowledge and imagination inspired by C-K theory Hatchuel and Weil (2003a) and discuss the possible directions for further development of a 'logic of design'.

1 Introduction

While there are extensive studies on modeling and theorizing about design[1], most engineers still perceive design as practice or even art. Considering the large variety of design practices, it is arguably difficult to determine a fixed universal set of core characteristics and phenomena defining design science.

[1] E.g. Braha and Reich (2003), Maher and Gero (1990), Maimon and Braha (1996), Marples (1960), Shai and Reich (2004a;b), Suh (1990), Takeada et al. (1990), Yoshikawa (1981), Zeng (2002)

Indeed, what is scientific about design? One possible answer is the specific mode of reasoning design requires. Design is a reasoning process by which new descriptions for new classes of objects are generated. Thus, the properties of this creative process constitute a core phenomenon for design research.

Yet, traditional formal theories and models of design usually treat creativity as external phenomena, for example, due to imagination, as e.g. in Simon (1969) and Simon (1995). A first category of models aim at structuring the process, G.Pahl and Beitz (1984)), and hence, they provide models for the organization of design activities.

A second category describe design as the progressive refinement of the object being designed, Gero (1990) or Braha and Reich (2003), and hence, propose a model of the evolution of design artifacts. In both cases, the creative mechanisms of design reasoning are not deemed as central (i.e. the models do not seek to explain where do refinements come from or what sort of organizing allows better creativity).

One notable exception to these works is the concept-knowledge (C-K) theory of design Hatchuel and Weil (2002; 2003a) and Hatchuel and Weil (2009). C-K theory places the creative generation of new definitions and objects at the heart of design through a notion of *conceptual expansion*. It describes design as a reasoning process where an object's identity is revised by changing its definition through the introduction of new properties. If successful, a new class of objects has been designed.

Inspired by C-K theory and building on our previous work Kazakci (2009), Hendriks and Kazakci (2010), the current work presents a framework demonstrating how C-K type design reasoning can be formalized within logic.

Our contribution in the current paper is twofold. First, we introduce formal *Design Operators* replacing the underspecified operators of C-K theory, and next we sketch how *Design Scenario's* can be build with such operators to fully explain the phenomena described in C-K theory, such as the *conceptual expansion* mentioned before.

This opens up perspectives of automating, at least partially, C-K type design reasoning, along with many research questions. Computer aided design support has been one of the main inspirations for the application of formal methods to design theory since Simon (1969). How precisely the use of constructive reasoning in design, e.g. intuitionistic logic, can yield a recipe for an artifact based on a proof of its possible existence, is one such a research question. The use of Design Scenarios on the other hand points toward another possible connection with logic, via dynamic logic.

The plan of the paper is as follows. Section 2 gives a summary of C-K theory

and previous work on its formal foundations. Section 3 introduces the basic logical framework we use. Design stages and design concepts and body are defined and some of their properties are summarized. Section 4 introduces design operators and then extends these to Design Scenarios. Some concluding remarks can be found in section 5.

2 A dynamic perspective on Design Reasoning

C-K theory describes design reasoning as the dynamic interplay of knowledge and concepts. Concepts are (partial) descriptions of a new object which existence (or the impossibility of such existence) cannot be decided based on the current knowledge.

Each stage in the design process is defined by its current knowledge and its current concept. The design space can be considered as the product of the knowledge space and the concept space.

- Knowledge Space The elements in the *Knowledge Space* are sets of knowledge, representing all the knowledge available to a designer (or to a group of designers) at a given time.
- Concept Space The elements of the *Concept Space* are (partial) descriptions of unknown objects that may or may not be possible to exist.

Whereas knowledge consists of true propositions, concepts are propositions whose status is unknown (based on the given knowledge). Such propositions can neither be stated as true, nor false by the designer at the moment of their creation (e.g., *'some tires are made of dust'*).

According to C-K theory, creative design begins by adding a new and unusual property to an existing concept C to form a new concept C' (e.g. *'tires for life'*). The elaboration of concept can then be continued either by further *expansions* (tires for life are made of silicon) or by *restrictions* (that is by adding usual properties of the initial concept, e.g. tires for life are round). Such conceptual expansions or restrictions are called *partitioning* in C-K theory.

Note that, according to C-K theory, new concepts are formed by combinations of concepts occurring in the propositions of existing knowledge. The designer will use his or her body of knowledge K either to partition further the concepts, or to attempt a validation of a given concept. This last type of operation (K-validation) corresponds to the evaluation of the feasibility of a design description (e.g. could it exist).

Often the validation of a concept will not be readily possible. In order to validate concept *C*, new knowledge warranting the existence conditions of such an object should be acquired. In terms of C-K theory, knowledge should be expanded (*K*-expansion). Such new knowledge may bring new concepts into the game, allowing for new expansions and restrictions of the design concept *C*.

The central proposition of C-K theory is thus "design is the interaction and dual expansions concepts and knowledge" Hatchuel and Weil (2003a; 2009).

2.1 Previous work on formal aspects of C-K theory

Despite the mathematical references and metaphors used in the presentation of the theory, a full mathematical presentation of the theory has not been provided to date.

Nevertheless, some steps for formalizing C-K theory have been taken in some recent work. Hatchuel and Weil (2003b) argues that there are significant similarities between the type of reasoning described by C-K theory and Forcing, a technique used in Set Theory for constructing alternative set theoretic models with desired properties. It is claimed that the parallel between Forcing and C-K theory is an important step for design theory in general but this issue needs more formal investigation.

In a complementary approach, Kazakci et al. (2008)shows that C-K type reasoning can be implemented with much more simple formalisms. They use propositional term logic to model the basic ideas of C-K theory. They suggest a notion of "models of K space" to emphasize that different structures (or formalism) used to model knowledge will yield different conceptive power and degrees of flexibility in reasoning.

In Kazakci (2009) a first-order logical formal account of C-K theory's core notions is presented. To emphasize the constructive aspects of a design process, intuitionistic logic is used to study the interaction and expansion of concepts and knowledge, based on the definitions of the basic notions. Building on this work, Hendriks and Kazakci (2010) complements this approach in that we consider the core proposition of the theory, the dual expansion of concepts and knowledge, and investigates the logical implications of such a principle.

2.2 Producing concepts from knowledge

One of the intriguing idea from C-K theory is the emerging of new *concepts* from (new) knowledge. Can we generate concepts using logic? Recall that in

C-K theory for C to be a concept it has to be 'unknown', in as far as our body of knowledge allows us, whether there are instances of the concept or not. From the examples in C-K theory literature we can reconstruct a simple mechanisms at work here, using the language $\mathcal{L}_K$ in which our body of knowledge K is represented. If we assume $\mathcal{L}_K$ is a first order logic language enriched with a set of constants for specific individuals and predicates (the signature of $\mathcal{L}_K$), it is only natural that extending K may also extend the signature. The new part of the body of knowledge may introduce new constants (Planck's h, the star Vega-β, President Obama) and predicates (being married to, prime, being the president of).

Concepts can now be generated from knowledge by recombination of expressions used in the body of knowledge. If *'Beatrix is the queen of The Netherlands'* then 'x is the queen of y' is a phrase in $\mathcal{L}_K$. Which allows us to form an expression like: *'Planck is the queen of Vega-β'*.

If we model phrases as formulas with one free variable x (a restriction we may lift later on) this amounts to forming conjunction of existing phrases, like in $Boat(x) \wedge Flies(x)$.

Such a new phrase (if K does not imply that $\forall x.Boat(x) \wedge Flies(x)$ and neither that $\neg\forall x.Boat(x) \wedge Flies(x)$, could be used as a concept C and starting point for design.

Combined with knowledge extension this simple mechanism will supply the design process with a wealth of new concepts without any appeal to imagination or hidden creative powers.

The mechanisms above can easily be extended further, e.g. by lifting the restriction on the type of phrases used. Like in the example *'Planck is the queen of Vega-β'*, where we could start the design turning this into the question *'Wouldn't it be nice if Planck is the queen of Vega-β?'* This could 'branch' (in a series of steps) into 'absorbing very bright light to produce both energy and comfortable back ground illumination'.

2.3 Generating knowledge from concepts

Knowledge may guide the designer in avoiding branches that are known to be undesirable (like in case C contains some constraints on the amount of money to spend on producing a $Car(x)$, combined with the knowledge that $Gold(x)$ will raise the cost of production considerably).

One simple mechanism of knowledge expansion occurs when the designer (or the design team) is aware of the body of knowledge K, say there knowledge

is a part of K called K_0. Extending this K_0 could be done be Googling the web, searching Wikipedia, asking experts etc.

A second mechanism could involve further research in the field K. The experts in the field might be unable to answer the questions of the design team. Further research and experimentation could be necessary. For example we might want to use carbon-epi-hexa-fluor-plexitude for the heat shield of our Vega-β-surveyor, but it is unknown what will happen if carbon-epi-hexa-fluor-plexitude is heated above 5,000 degrees Celsius.

A third mechanism may occur when we try to combine parts of theories, say the nanotechnology with the neurobiology of humain brain, say, in order to use nanotechnological devices to record firing patterns of neurons. Whether such thing is possible may be a complete new subject for scientific research.

3 A logical framework

In this section, we will describe the design process as generating design stages $\langle K;\ C\rangle$, where K is some body of knowledge and C is a concept. In C-K theory one is especially interested in design stages where C is totally new for K. Here we will allow degenerated stages that can be discarded once they are seen as inconsistent or in fact not new at all (hence C turns out to be feasible already based on K).

The design process may extend a design stage in principle in infinitely many ways. One could try to imagine all 'existing' possible 'bodies of knowledge' or all 'possible concepts' and try to describe these as (a special sort of) sets, such that the operations in the design process which transform a stage $\langle K;\ C\rangle$ into a new $\langle K';\ C'\rangle$ can be defined as a special kind of (extended) 'search operations', not unlike known search algorithms (e.g. on databases or in linear programming).

Not only Ockham's Razor makes the 'existence' of such 'Knowledge Spaces' or 'Concept Spaces' suspect, simply from a pragmatic point of view, some sort of constructive reasoning in the design process seems to be attractive. Operations in the design process defined using 'mental images' of infinite collections containing 'lawless' sequences are certainly not constructive Kazakci (2009). Note that according to C-K theory there is no 'algorithm' or construction that will determine the next step in a design path. Such a path can be seen to behave 'lawless' in Intuitionist sense Kazakci (2009). Therefore, at least from a pragmatic point of view, it seems reasonable to assume that at each point in the design process the body of knowledge has a finite representation.

We introduce a basic logical framework that will allow us to represent in logical terms some of the ideas in C-K theory. We will be a little vague on the precise logical language $\mathcal{L}$ and the rules of the logic used. This is because we want to extend the framework later on. This could be done in different directions and we don't want to assume too much about either the expressivity of the language or the strength of the logical rules.

Without any problem the reader may assume $\mathcal{L}$ is the language of predicate logic and the logic is the usual classical logic (although all our results will be valid in intuitionist logic as well: we will simply not use the axiom $A \vee A$ or the rule $A \vdash A$ of classical logic). Note that, in a constructivist type of logic, like intuitionist logic, a proof of C from K is a construction, one that under a certain conditions can be used as a recipe to construct an instance of $C(x)$ based on the constructions that exist according to K - which is a suitable characteristic for modeling design endeavor.

In our notation $T \vdash A$ means a formula A is provable from the set of formulas T. T^* will be the set of all formulas (in $\mathcal{L}$) derivable from T ($T^* = \{A \in \mathcal{L} \mid T \vdash A\}$). So $T^* \subseteq S^*$ and $T \vdash A$ implies $S \vdash A$.

3.1 Design stages and design space

Let us model the body of knowledge (as a first approximation) as a finite set of formula, in the language of first order logic (predicate logic). Such a finite theory K will always be 'partial knowledge' and hence extendable. Again in a first approximation, we could model the 'concept' C as a formula (in predicate logic).

Our definition of design reasoning is based on the one in C-K theory, where design is defined as a *reasoning* activity, starting with a proposition of the design concept and proceeding by *operators* adding knowledge or expanding the design concept. The definition below is slightly more abstract, which would seem to allow more design stages and design steps than described in C-K theory itself.

Definition 3.1. A *design stage* is a pair $s = \langle K; C \rangle$, where K a finite set of sentences, called the *body-of-knowledge* of s and C a sentence, called the *design concept* of s.

A design stage $\langle K;\ C \rangle$ is called *consistent* if $K \nvdash \neg C$.

A design stage $\langle K;\ C \rangle$ is called *open* if $K \nvdash C \vee \neg C$.

A *design step* is a pair (s_0, s_1) where s_0 and s_1 are design stages. We will often use the notation $s_0 \Rightarrow s_1$ for the design step (s_0, s_1). Usually we will also assume that $s_0 = \langle K_0;\ C_0 \rangle, s_1 = \langle K_1;\ C_1 \rangle$ etc.

A design step $s_0 \Rightarrow s_1$ is called *sound*, written as $s_0 \overset{s}{\Rightarrow} s_1$, if s_1 is consistent and for all $A \in K_0$ it is true that $K_1 \vdash A$ (i.e. $K_1 \vdash \bigwedge K_0$) and $K_1, C_1 \vdash C_0$.

Design step s_0 *implies* s_1 if $\bigwedge K_0 \to C_0 \vdash \bigwedge K_1 \to C_1$.

Design step s_0 is *equivalent* with s_1 if s_0 implies s_1 and s_1 implies s_0.

We will use the notation $s_0 \vdash s_1$ if s_0 implies s_1 and $s_0 \equiv s_1$ when they are equivalent.

Note that in the definition above there are no constraints on the type of sentence used as the design concept. Our C does not necessarily have the form $\exists x.P_0(x) \wedge \ldots \wedge P_n(x)$, where the P_i are the desired properties for the object that we wish to come out of the design process. It is even not required that C is an existential formula. One can imagine for example that the 'thing' we try to design is a way of transforming all x with property A into some y with relationship R between the x and the y. So $C = \forall x(A(x) \to \exists y R(x, y)$ would be a conceivable design concept.

That we assume the body-of-knowledge K to be finite is not a real restriction in practice (at any moment of time each finite group of people could only be aware of a finite number of facts). We could allow for infinite K in principle, but this would slightly complicate our formulas like in the definition of $s_0 \vdash s_1$, where for an infinite K the conjunction $\bigwedge K$ formally is not defined.

The following facts follow directly from our definitions.

Observation 1. Let $s_0 = \langle K_0;\ C_0 \rangle$, $s_1 = \langle K_1;\ C_1 \rangle$ and $s_2 = \langle K_2;\ C_2 \rangle$ be design stages.

1. If s_0 is consistent then K_0 is consistent (i.e. $K_0 \nvdash \bot$).
2. If $s_0 \Rightarrow s_1$ is sound and $K_0 \subseteq K_1$ then s_0 is consistent.
3. If $s_0 \equiv s_1$ and s_1 (or s_0) is consistent then $s_0 \Rightarrow s_1$ is sound.
4. If $s_0 \Rightarrow s_1$ and $s_1 \Rightarrow s_2$ are sound, so is $s_0 \Rightarrow s_2$.

Our definition of sound design steps is one way of formalizing the intuitive notion of one design stage 'implying' another, found in most descriptions of design (i.e. in Hatchuel and Weil (2009) and Braha and Reich (2003)). The informal notion is sometimes used in a loose sense of 'having some reason to go from s_0 to s_1'. On other occasions the 'implication chain' is a series of stages where apparently more logic is involved. Our notion of soundness is a weak form of such a logical connection, whereas the defined implication ($s_0 \vdash s_1$) is rather strong[2]

Two special cases of sound design steps may clarify the often observed difference in direction of the 'implication' between 'refining' the specifications and 'refining' the (structural) knowledge. Note that if $K_0 = K_1$ and $s_0 \Rightarrow s_1$ is sound, then $K_0, C_1 \vdash C_0$ and hence $s_1 \vdash s_0$. On the other hand, if $C_0 = C_1$ and $s_0 \Rightarrow s_1$ is sound, $K_1 \vdash \bigwedge K_0$ and hence $s_0 \vdash s_1$.

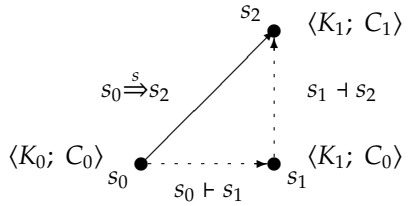

Figure 1: Splitting design steps in a K- and a C-component

As long as we confine ourselves to sound design steps that only change the design concept or only the body-of-knowledge (which theoretically we could obviously always do by splitting up steps if necessary, see figure 3.1), we could use the implication of one stage by another as a basis[3] for describing the design process (taking care each time using the right direction of the 'implication' between the states). In a more realistic model of design steps, where both parts of the state can change in one single step, the implication between states (e.g. in the way defined above) becomes awkward to deal with.

Simple examples of sound steps are:

- (Adding knowledge) $\langle K;\ C\rangle \Rightarrow \langle K, A;\ C\rangle$

[2]Another notion of (strict) implication would be $s_0 \vdash\!\!\sim s_1$ defined as $K_0 \vdash C_0 \ \Rightarrow\ K_1 \vdash C_1$. Implication of states implies strict implication, but not the other way around.

[3]This would result in a more restrictive relationship than soundness.

- (Adding properties) $\langle K;\ \exists x C(x)\rangle \Rightarrow \langle K;\ \exists x(C(x) \wedge P(x)\rangle$
- (Introducing a definition) $\langle K;\ C \wedge D\rangle \Rightarrow \langle K, P \leftrightarrow C \wedge D;\ P\rangle$

An example of $s_0 \vdash s_1$ where $s_0 \Rightarrow s_1$ is not sound would be $s_0 = \langle K;\ C\rangle$ and $s_1 = \langle K;\ C \vee D\rangle$. If $K \nvdash D \rightarrow C$ then K together with $C \vee D$ does not imply C.

As can be concluded from the examples above, our formalism does not require the design concept to be of the form $\exists x C(x)$ as perhaps expected from the informal description of the design concept as a description of something we seek to exist. Allowing the full generality of a first-order sentence provides us with a lot more flexibility as we will show in the sequel.

As promised in the introduction we would prove the conjecture that using sound design steps leading to a closed design stage will provide a proof for the original design concept.

Theorem 1. *Let $s_n = \langle K_n;\ C_n\rangle$ be a closed design stage reached after n sound design steps from $s_0 = \langle K_0;\ C_0\rangle$ (so $K_n \vdash C_n$) then $K_n \vdash C_0$.*

Proof. From the fact 1.3 we can conclude that $s_0 \overset{s}{\Rightarrow} s_n$. Hence, by definition, $K_n, C_n \vdash C_0$. As $K_n \vdash C_n$ by the Cut-rule it follows that $K_n \vdash C_0$. ⊣

4 Design scenarios

In C-K theory design operations are made within or between 'K-space' and 'C-space'. Translated into our formalism we get:

Create	$K \longrightarrow \langle K;\ C\rangle$
	Forming the first concept from properties of K.
Refine	$\langle K;\ C\rangle \longrightarrow \langle K;\ C'\rangle$
	Adding a property from K to C.
Enhance	$\langle K;\ C\rangle \longrightarrow \langle K';\ C\rangle$
	Using properties from C to find additional knowledge.
Expand	$\langle K;\ C\rangle \longrightarrow \langle K';\ C'\rangle$
	Combining Refine and Enhance.
Validate	$\langle K';\ C\rangle \longrightarrow \langle K;\ C\rangle$
	Adding knowledge about the existence of C to K
	This may or may not add C or $\neg C$ to K.

Design concepts often are considered as (partial) definitions of an artifact, formalized as formulas $\phi(x)$ with exactly one free variable x. In the publications

on C-K theory the design concept is introduced as a set of properties. The idea behind this (e.g. in Kazakci (2009) is that the concept C is a sentence of the form $\exists x.P_1(x) \wedge \ldots P_n(x)$ and $\{P_1, \ldots, P_n\}$ is the set of properties that make up the concept C.

To treat the design concept as a set of properties within our formal framework forces us to introduce properties, sets of such properties and some operator $\exists$ to bridge the gap between sets of properties and the propositions C introduced before.

Definition 4.1. Let S be a set of properties.
$\exists S$ will be the proposition claiming the existence of an object with all the properties in S.
For the design stage $\langle K;\ \exists S\rangle$ we will also use $(K;\ S)$.

A further analysis of the design operations listed above shows that aparently also other basic extra-logical operations on (sets) of properties and propositions, like Query, Test and operators to extract properties from formulas, will play a role in describing the dynamic interplay between knowledge and imagination.

Definition 4.2. Let A be a formula and P a predicate occurring in A, than both $P(\vec{x})$ and $\neg P(\vec{x})$ are *properties* of A.
Let A be a formula and S a set, then the basic design operations are defined as:

$\neg$	For property P, $\neg P$ is property: the complement of P.
$.Prop$	$A.Prop$ is the set of properties of A.
!	$!S$ is an element of S.
$!_{\subseteq}$	$!_{\subseteq}S$ is a subset of S.
?	If S is a set of properties, $?S$ is a sentence (obtained by a query).
test	If A is a formula $test(A)$ is a formula.

In our extended language we will allow the use of set operations, relations (e.g. $\cup, \cap, \in, \subseteq, =, \neq$) and the contstant $\emptyset$ (for the empty set).

How exactly selection, e.g. of a subset of properties is made, or how the query ? or the *test* are performed we keep for now formally undefined. Several options may be investigated within (and by expanding) our formal framework. Our definition of the complement of a property does not rule out that $\neg\neg P$ is the same property as P, nor does it requires so. Usually one surely would expect $\neg\neg\neg P$ to equal $\neg P$, but for the moment we will not fix the rules of the formalism at this level of detail.

For selection one may thinks of a random choice. The query may be a Google-search with a the English names of the properties, out of the result

of which somehow some piece of knowledge is selected and translated in the language $\mathcal{L}_{K'}$, where K' may be an extension of K containing some new properties.

4.1 Scenarios

We are now ready explain what we consider the core of C-K theory, the *dual expansion of concept and knowledge*, in terms of our basic operations. Consider the following scenario:

$$
\begin{array}{lll}
\langle expand \rangle(K;\ S) & := & \{ \\
 & & P \quad := \quad !S; \\
 & & K_P \quad := \quad ?\{P\}; \\
 & & A \quad := \quad !K_P; \\
 & & Q \quad := \quad !A.Prop; \\
 & & K_P \quad := \quad ?\{Q\}; \\
 & & R \quad := \quad !\{Q, \neg Q\}; \\
 & & \textbf{return} \ \ (K \cup K_Q \cup K_P;\ S \cup \{R\}) \\
 & & \}
\end{array}
$$

This scenario can be read as: Based on a property of S we expand our knowledge to K_P. This new knowledge, resulting from a query, may introduce some new properties, e.g. the property Q. A query fed by Q could lead to new knowledge K_Q. The set of properties for the design concept is now extended with either Q or its complement, whereas the body of knowledge meanwhile is extended with both K_P and K_Q.

Assume we started with $K \nvdash \exists S$, it now might be the case that $K, K_P, K_Q \vdash \exists S$, or even $K, K_P, K_Q \vdash \exists(S \cup \{R\})$.

The scenario above kept close to the description of expansion in C-K theory, as in Hatchuel and Weil (2002).

5 Conclusions

Our approach in this paper has been a mix of formal reasoning and informal analysis of engineering design, especially as described by C-K theory. The result is a formal framework with some 'dynamic 'features, without a well defined semantics.

References

D. Braha and Y. Reich. Topological structures for modelling engineering design processes. *Research in Engineering Design*, 14:185–199, 2003.

J. Gero. Design prototypes:a knowledge representation schema for design. *AI Magazine*, 11:26–36, 1990.

G.Pahl and W. Beitz. *Engineering Design: a systematic approach*. The Design Council, London, 1984.

A. Hatchuel and B. Weil. C-K theory: Notions and applications of a unified design theory. In *Proceedings of the International Conference on the Sciences of Design*, page 14, INSA Lyon, France, March 2002.

A. Hatchuel and B. Weil. A new approach of innovative design: an introduction to C-K design theory. In *Proceedings, International Conference on Engineering Design*, 2003a.

A. Hatchuel and B. Weil. Design as forcing: Deepening the foundations of C-K theory. In Marjanovic et al. (2010), page 14.

A. Hatchuel and B. Weil. C-K design theory: an advanced formulation. *Research in Engineering Design*, 19:181–192, 2009.

L. Hendriks and A. Kazakci. A formal account of the dual expansion of concepts and knowledge in C-K theory. In Marjanovic et al. (2010), pages 49–58.

A. Kazakci. A formalization of C-K design theory based on intuitionistic logic. In A. Chakrabarti, editor, *Proceedings of the International Conference on Research into Design ICORD09*, pages 499–507, Bangladore, India, January 2009.

A. Kazakci, A. Hatchuel, and B. Weil. A model of ck design theory based on based on a term logic: a formal background for a class of design assistants. In D. Marjanovic, M. Storga, N. Pavkovic, and N. Bojcetic, editors, *Proceedings of the 10th International Design Conference DESIGN 2008*, pages 43–52, Dubrovnik, Croatia, May 2008.

M. Maher and J. Gero. Theoretical requirements for creative design by analogy. In P. A. Fitzhorn, editor, *Proceedings of the first International Workshop on Formal Methods in Engineering Design, Manufacturing, and Assembly*, pages 19–27, Colorado Springs, Colorado, January 1990.

O. Maimon and D. Braha. A mathematical theory of design. *International Journal of General Systems*, 27:275–318, 1996.

D. Marjanovic, M. Storga, N. Pavkovic, and N. Bojcetic, editors. *Proceedings of the 11th International Design Conference DESIGN 2010*, Dubrovnik, Croatia, May 2010.

D. Marples. The decisions of engineering design. *Journal of the Institute of Engineering Designers*, pages 181–192, December 1960.

O. Shai and Y. Reich. Infused design: I theory. *Research in Engineering Design*, 15:93–107, 2004a.

O. Shai and Y. Reich. Infused design: II practice. *Research in Engineering Design*, 15:108–121, 2004b.

H. Simon. *The Sciences of the Artificial*. MIT Press, Cambridge, Mass., 1969.

H. Simon. Problem forming, problem finding and problem solving in desing. In A. Collen and W. Gasparski, editors, *Design and Systems: General Applications of Methodology.*, pages 245–259. Transactions Publishers, New Jersey, 1995.

N. P. Suh. *The principles of design*. Oxford University Press, New York, 1990.

H. Takeada, P. Veerkamp, T. Tomiyama, and H. Yoshikawa. Modeling design processes. *AI Magazine*, 11:37–48, 1990.

H. Yoshikawa. General design theory and a cad system. In T. Sata and E. Waterman, editors, *Man-Machine Communication in CAD/CAM*, pages 35–38. North-Holland, 1981.

Y. Zeng. Axiomatic theory of design modeling. *J. Integr. Des. Process Sci.*, 6:1–28, August 2002. ISSN 1092-0617.