INTERNATIONAL DESIGN CONFERENCE - DESIGN 2008
Dubrovnik - Croatia, May 19 - 22, 2008.

DESIGN 2008

A MODEL OF CK DESIGN THEORY BASED ON TERM LOGIC: A FORMAL CK BACKGROUND FOR A CLASS OF DESIGN ASSISTANTS

A. O. Kazakci, A. Hatchuel and B. Weil

Keywords: CK design theory, term logic, concepts, knowledge, design assistant

1. Introduction: models of CK theory

1.1 CK theory: a brief overview

The CK theory is a theory of design reasoning introduced by Hatchuel and Weil (1999, 2002, 2003). The theory describes design based on the interaction of two interdependent spaces. The space K of knowledge corresponds to the available knowledge (such as engineering models and scientific facts). The space C of concepts corresponds to the partial descriptions of objects (which captures the notion of design briefs or broad specifications). The design process can be described then by the interaction and co-evolution of these two spaces through the application four types of operators; C→K, K→C, K→K and C→C.
Design proceeds by partitioning the concepts of the C space, i.e., by adding or deleting properties, coming from the K space. Two kinds of partitioning are possible; *restrictive partitions* add usual properties of the object whereas *expansive partitions* introduce novel properties. The partitioning of a concept may result in an expansion of K – that is, learning: new knowledge is necessary to pursue creative expansions of space C. During design, the available knowledge does not allow to decide whether the objects represented in the C space are feasible.
The aim of the design process is to expand simultaneously C and K spaces in order to create, on the one hand, new concepts (which leads to creativity), on the other hand, new knowledge (which leads to learning) allowing the realization of these concepts. Design ends when the properties introduced into the concept can be validated in K space, i.e., it can be confirmed in K that such an object may exist.CK theory has been the focus of different research efforts along various dimensions such as organising design activities, capturing design rational, design assistants and formal models of the theory (Kazakci and Tsoukiàs, 2005, Elmquist and Segrestin, 2007; Ben-Mahmoud Jouini *et al.*, 2007; Hatchuel *et al.*, 2007). In this work, we are interested in this last aspect - formal models of CK theory.

1.2 Models of CK theory: the impact of the structure and the properties of K-space

Recently, Hatchuel *et al.* (2007) established a correspondence between CK theory and Forcing. Forcing is a method of the modern set theory invented by Paul Cohen for the creation of new sets (Cohen, 1963, 1964). Using Forcing, Cohen showed that, starting with a model M verifying the axioms of the set theory and a proposition P which is *undecidable in M* (i.e., it is impossible to state the truth or falsity of P in M), it is possible to construct progressively a new model N (containing M) in which P (or its negation) is true. The construction proceeds by adding successive conditions that allows controlling the properties of N and thus leaving intact the meaning of the propositions of M.

Forcing can be seen as a special model of CK design theory where space K is reduced to set theory. The proposition P is a (partial) description of an object that we cannot determine the truth or the falsity in the beginning of the design process. M is the initial knowledge space. The conditions introduced successively correspond to properties introduced in the C space by partitioning. Design ends when a new collection of sets N is constructed where P becomes part of the N and it is possible to state the logical status of P. The correspondence deepens the theoretical roots of CK design theory but it also shows that when constructing formal models of design (seen from the CK theoretic perspective), the choice of the structure of K space has a decisive impact on the definition of C space and design operators. *Which other models of K space would allow capturing CK reasoning?* The current work begins to investigate this question with a particular model of K space based on term logic and discuss some perspectives related to building design assistants based on CK theory.

1.3 Which models of CK theory for building design assistants? – Some requirements

The question of building design assistants based on CK theory has already been investigated by Kazakci and Tsoukiàs (2005). They suggest that introducing a third space, an environment space E is necessary for the tool to communicate with a user and other external entities (such as databases). The desirable properties of a model for design assistants follow directly from the possible interactions between C, K and E spaces:

- **Distinction and interaction of C and K:** The model should distinguish between C and K spaces by allowing the formation of new concepts and expansive partitions. It should allow activating relevant knowledge and suggesting restrictive partitions.
- **Distinction and interaction of E and K:** The model should allow a dynamic linkage with an environment space to make knowledge expansions possible: to acquire knowledge that does not exist in K space (nor can be inferred in it) an assistant should interact with external sources. The model should also allow propagating the results of knowledge acquired this way and revise the assistant's existing knowledge.
- **Standard K→K operations:** The model should allow classical operations on a knowledge base such as query operations or inference.

For achieving C-expansivity, K-expansivity is necessary (Hatchuel, Weil and Le Masson, 2007) and for achieving K-expansivity, mere inference is not sufficient: the assistant should be able to communicate with the E space and *acquire* new knowledge (Kazakci and Tsoukiàs, 2005). Another question of interest to this work is thus: *Are there models of K which capture both properties of CK theory and offer the required computational properties for building design assistants?*

1.4 Purpose and outline of the paper

Hatchuel and Weil (2003) suggest that, *a priori*, any kind of logic can be used to capture the key notions of CK design theory. Following a proposition by Kazakci (2005, 2007), we investigate the use of NAL, a term logic introduced by Wang (1995, 2006) which offers promising computational properties for the development of design assistants based on CK theory. We show the way NAL captures notions of CK theory and how it can be used for design assistance. The paper is organised as follows.

- In section 2, we present NAL and we discuss its suitable properties for capturing CK theory.
- Section 3 shows how to interpret the key notions of CK theory within the framework of NAL.
- Section 4 discusses how NAL can be used to support design activity and related issues.
- In section 5, we summarize the results and conclude.

2. NAL, a term logic with suitable properties for modelling CK theory and building design assistants

2.1 Term logics: an overview

Term logic is a kind of syllogistic logic which is close in spirit to the traditional Aristotelian logic. The main characteristics of term logic are (Wang, 1995; Englebretsen, 1981):

- Each proposition is of the form "subject-copula-predicate".

- Subjects and predicates are terms. Terms can be considered as symbols or names of concepts.

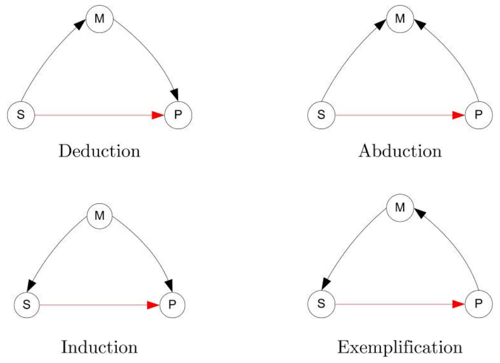

Figure 1. Basic inference rules in NAL; M is the common term appearing in two propositions S ® M **and** M ® P **; the conclusion of the inference is** S ® P

- Copulas correspond to relations that may be interpreted as “is a” or “associated with” or “related to”. Different copulas can be used within a given term logic.
- Inference is syllogistic, that is, two propositions having a term in common are used to produce a third one – a conclusion; Figure 1.

Hence, terms can be combined using a copula to form propositions and from two propositions, a third one can be inferred. Given these properties, term logic can be seen as an associative reasoning system.

2.2 NAL, a term logic with experience grounded semantics

2.2.1 A bidirectional inheritance relation

The language NAL (Non-axiomatic logic) is based on a special copula called “inheritance” which is denoted by “® ”. Given two terms *S* and *P*, the proposition S ® P means “ *S is a specialisation of P and P is a generalisation of S* ”. Let us precise that this definition is different from the usual concept of (extensional) inheritance and it implies bidirectionality of the relation; *S inherits the properties of P just as P inherits the instances of S.* Intuitively, with this kind of representation we can represent associations between terms, e.g. *Eagle* ® *Animal* , where the subject is less general than the predicate. More subtle representations are possible within the framework of NAL (such as the consideration of variables, other copulas); the reader is referred to (Wang, 2006).

2.2.2 Beliefs and their truth values

NAL has an experience-grounded semantics; i.e. the truth value of a proposition S ® P is defined (on a theoretical level) based on the number of times the relation (or the association) has been experienced by the system.
A propositional sentence S ® P has an associated truth value $< f,c >$, where f Î $[0,1]$ is the frequency, representing the positive evidence relative to all evidence experienced by the system, and where c Î $[0,1)$ is the confidence, representing the conviction of the system considering the amount of evidence the system has on the relation and the possible future evidence (Wang, 1995, 2005). Propositions with truth-values are called *beliefs*. Remark that *Eagle* ® *Animal* with an associated truth value is a valid formula just as *Animal* ® *Eagle* , although one would expect the second proposition to have lesser frequency and confidence. A proposition without a truth value is treated as a logical question.

2.2.3 Inference

Inference in NAL, like in all other syllogistic logic, consists in taking two beliefs as arguments and returning a belief which is the conclusion of the input arguments. NAL defines many syllogistic inference rules such as deduction, abduction, induction, exemplification, etc. Depending of the configuration of the arguments, different rules are used to derive a conclusion; Figure 1.

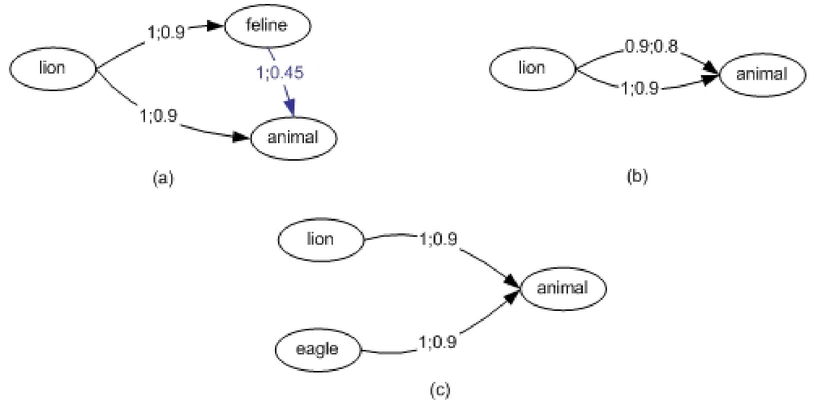

Figure 2. Examples of sets of propositions in NAL

The truth value $< f,c >$ for a conclusion is determined based on the truth values $< f_1,c_1 >$ and $< f_2,c_2 >$ of the arguments. Wang (1995, 2006) gives the following formulae:

- $F_{ded} : f = f_1.f_2$, $c = f_1.c_1.f_2.c_2$, for deduction,
- $F_{abd} : f = f_2$, $c = (f_1.c_1.c_2)/(f_1.c_1.c_2 + k)$, for abduction,
- $F_{ind} : f = f_1$, $c = (c_1.f_2.c_2)/(c_1.f_2.c_2 + k)$,for induction,
- $F_{exe} : f = 1, c = f_1c_1f_2c_2/(f_1c_1f_2c_2 + k)$, for exemplification.

where k is a constant (Wang, 1995). Note that as an inference chain goes longer, conclusions have lesser and lesser confidence values: they become more tentative and the system is unsure about them.

2.2.4 Powerful features of NAL: allowing open-endedness and unknown

There are two key properties of NAL which makes it suitable for using it in modelling CK theory and for building design assistants. First, NAL has experience grounded semantics (Wang, 1995, 1998, 2005, 2006): “true” in NAL means “the relation has been experienced (or observed)”. On the other hand, by default, NAL considers “false” all the relations that have never been experienced: for a given relation, “false” means simply that the relation is not “recognized”; *it does not mean that the negation of the relation is “true”* (in fact, negation does not even exist in the basic version of NAL that we consider in this work, but only in higher order versions of the logic; see (Wang, 2006)). The relations that have not been experienced are not taken into account in the construction of the experience grounded semantics of NAL and this gives us the possibility to consider the truth value of these relations as *unknown*! An unknown proposition then corresponds simply to an undecidable inheritance relation that has no associated truth value and that cannot be given one: *Then, NAL can be used in a way that tolerates the unknown, thus allowing the distinction of C and K!* Second, NAL is designed to work *with limited resources within an open environment* (Wang, 2006). Systems using NAL can be connected to an environment. Whatever a system using NAL observes in the environment can be easily integrated into his knowledge and when the system lack resources (knowledge and/or time), it continues functioning with whatever resources it has. These are important attributes for systems like design assistants that will operate in real time and open environments.

3. How to capture key notions of CK theory using NAL?

3.1 The knowledge space

How can we model a knowledge space with NAL? At an abstract level, it consists in a set of terms, a set of beliefs relating those terms and a set of inference rules and algorithms. The logical status of each

proposition belonging to K space is known and given by their truth values. In the simplest case, the resulting structure can be conceived as a graph. Figure 2 depicts some simple knowledge structures represented in NAL.
In Figure 2a, three propositions have been represented as bivalued edges of a graph where the valuations correspond to truth values. Hence, a sentence (a set of propositions) in NAL is a graph. Note that, in practical applications, terms may have an internal structure (for example, they may correspond to documents or component descriptions).

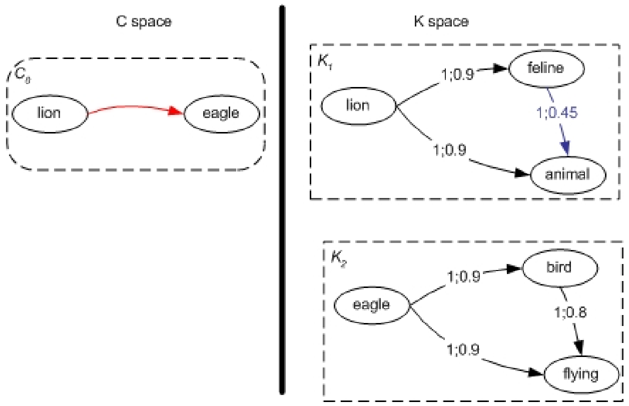

Figure 3. A semantic disjunction – an initial concept

3.2 K→K operations: inference and query answering

The formalism of NAL allows answering logical questions. Different questions are possible. We can simply ask the truth value of a proposition. For instance, the question (*lion* ® *animal*)? asks the truth value associated with the relation; K space returns (*lion* ® *animal*) < 1; 0.9 > for the current example. If inconsistencies are allowed in K space, multiple answers indicating different truth values can be found; Figure 2b. For such situations, different mechanisms may be devised that will allow a choice between multiple answers, for example, a rule such as "return the proposition with higher confidence". Another type of question is a term query: (? ® *animal*) ; what is an animal? Depending on the state of K space, multiple answers may exist; Figure 2c. In such situations, K space must contain a mechanism for selection – this time, not for conflict resolution, but for choosing between alternative answers. Depending on the setting, returning the most typical answer or the least typical answer or an answer satisfying some other criteria may be sought. In the example of Figure 2a, the only possible answer is (*lion* ® *animal*) < 1; 0.9 > . However, in such situations, other answers may be derived. For example, if, by induction, the proposition (*feline* ® *animal*) < 1; 0.45 > can be inferred, a second answer, though less confident, becomes possible.

3.3 The concept space and decidability of a proposition

What constitutes a concept with respect to previously described structure of the K space? The definition states: A concept is a proposition whose logical status cannot be readily determined in K space. For instance, if the K space contains a model of the usual set theory, undecidable propositions that cannot be verified nor refuted using the axioms of set theory form concepts (Hatchuel, Weil and Le Masson, 2007). In the case of NAL, we need a slightly different interpretation: although the underlying structure of the logic may be seen as a term algebra (with a specific set of axioms), the logic itself has an experience grounded semantics – it is not axiomatic. Said in other terms, the propositions in NAL are not derived from a set of axioms; they are experienced relations between terms and propositions derived thereof with an associative inference scheme. In this case, a

proposition that cannot be decided is simply *a proposition which cannot be inferred in any number of steps* given a specific configuration of the K space!
In the example of Figure 3, we have such a situation: given the current state of space K, it is impossible to determine the truth value of (*lion* ® *eagle*), a lion which is an eagle, solely by inference; Figure 3. It is not recognized in K; it is *unknown*. Technically, in its simplest form, a concept in NAL is an inheritance relation that will connect the two disjoint subgraphs. In the more general case, *any graph containing at least one non-valued edge and connecting two or more disjoint components of graph in K form a concept*. Thus, a chain formed by (*lion* ® *eagle*) and (*lion* ® *bird*) is a concept just like the chain formed by (*lion* ® *eagle*) and (*lion* ® *animal*) < 1; 0.9 > .

3.4 Partitioning a concept: restrictions versus expansions

How do we continue to elaborate a concept? Both the terms figuring in the expression of the concept, *lion* and *eagle*, exist in K space and there is some knowledge available about them. This knowledge can be activated (a C→K operation) using logical questions:

- (? ® *eagle*) – What are the instances of an eagle?
- (*eagle* ® ?) – What are the properties of an eagle?
- (? ® *lion*) – What are the instances of a lion?
- (*lion* ® ?) – What are the properties of a lion?

Answers to these questions like "lion is an animal" (*lion* ® *animal*) < 1; 0.9 > or "eagle is a bird" (*eagle* ® *bird*) < 1; 0.9 > may be used to partition the concept C_0 in the C space; Figure 4. These are K→C operations that correspond to restrictive partitions: it is known in K that eagle is a bird and a lion is an animal. Also, K→K operations can be used to infer new knowledge that may offer other possibilities for partitioning. For example, (*feline* ® *animal*) < 1; 0.45 > can be induced (by an operation K→K) and this knowledge can be used for partitioning (by a K→C operation). This would still be a restrictive partition since, once inferred, *feline* is a known specialisation of *animal* in K space. Then, what is an expansive partition? A partitioning by any proposition whose truth value cannot be determined by simple inference in K – like a *flying lion*, (*lion* ® *flying*) !

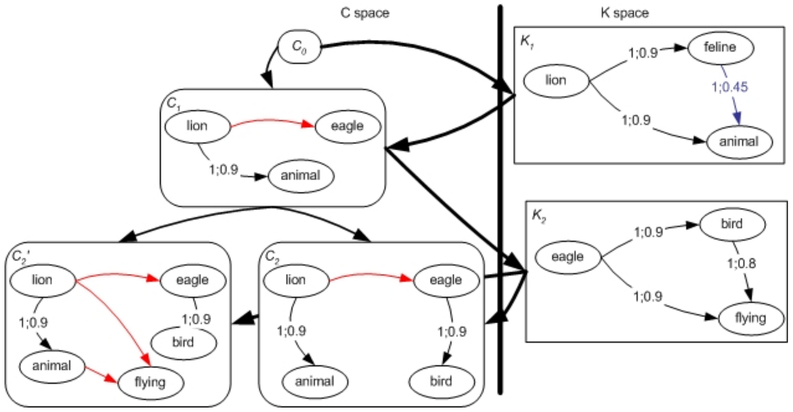

Figure 4. Restrictive and expansive partitions

In other terms, any proposition that would have formed a semantic disjunction! Adding such a proposition in the concept space is creating other edges in a graph of the C space that is impossible to evaluate at the time of partitioning; Figure 4. We see that expansive partitions are defined with respect to a knowledge space just like concepts. A proposition that forms a concept or an expansive partition for a designer might be simple knowledge for another: for a designer who has already seen a Griffin, say, in a cartoon movie, neither (*lion* ® *eagle*) would form a concept, nor (*lion* ® *bird*) an expansive partition!

3.5 Expansion of K and semantic conjunction

Given a concept as defined previously, it is possible to continue by successive restrictive partitions until all the terms of the K space related to the initial concept (that is, to the terms *lion* and *eagle*) appear in C space. However, this would still not allow determining the logical status of the concept: we need more than standard inference – it is necessary to have a knowledge expansion. The K space should include knowledge that will allow doing inference on the value of (*lion* ® *eagle*). Said in other terms, *we need a proposition that will allow connecting the two disjoint components* of the graph. One such proposition is (*bird* ® *animal*) $< 1;0.9 >$. If such knowledge can be acquired, its implications can be propagated by inference in order to derive further new results on the relation of the previously disjoint components. For the example, by deduction, (*eagle* ® *animal*) $< 1;0.81 >$ can be inferred. Then, it becomes possible to derive (*lion* ® *eagle*) $< 1;0.42 >$ by induction. At this moment, a semantic conjunction will be operated since in K space the valuation of all the edges forming the graph in C space is now available. The material presented in section 3 shows how notions of the CK theory can be modelled by the formal approach provided by NAL. This result is summarized in Table 1.

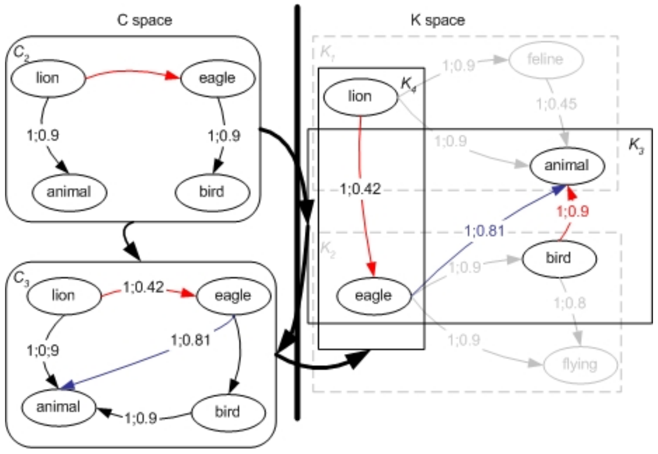

Figure 5. Expansion of K space and a semantic conjunction

4. Modelling CK theory using NAL: a powerful approach for design assistants

4.1 How to use NAL for design assistance?

We can now explicitly state steps of CK reasoning where a design assistant based on NAL can support a design process. The reader may notice that these properties correspond to the requirements set out in paragraph 1.3.

4.1.1 Undecidability checking and concept formation

Within the formalism, enumerating undecidable propositions becomes possible. The assistant can analyse the structure of the knowledge space and detect formally undecidable propositions. The formalism of NAL can then naturally be coupled by other computational approaches for analysing which of these propositions are interesting for the user. In any case, *a design assistant based on NAL might suggest concepts to its user.*

4.1.2 Operating restrictive and expansive partitions

Given a concept, the assistant can help the user to realise restrictive partitions or expansive partitions. Restrictive partitions are edges from the K space that are associated to a term already appearing in the concept graph and which are valued in K. The assistant can also look for expansive partitions, that is, for propositions which cannot be given a truth value in K and that will enable introducing new properties to the object description. Again, it is possible to use complementary approaches for allowing the assistant to determine which partitions are more interesting to operate given a specific context.

Table 1. Key notions of CK theory and their interpretations in NAL

CK Theory	Model in the term logic NAL
Knowledge space	- A graph (which may contain disconnected com- ponents) whose nodes correspond to terms and edges correspond to (valued) propositions of NAL - Algorithm(s) to operate on the graph - Algorithm(s) to inquire and to communicate with the external world (database, experts, etc)
Undecidable proposition in K	A proposition that cannot be given a truth value in any number of steps of inference, given a particular configuration of space K
Concept	A graph containing one or more undecidable propositions
Concept space	A treelike structure containing concepts
K→K operations	Inference and operations on the graph in K
K→C operation, Restrictive partition	Adding a valued edge (coming from K) into a graph in C
K→C operation, Expansive partition	Adding an undecidable proposition into a graph in C
Expansion of K	Adding a valued edge to K which is not obtained by inference (which comes from an environment space E or through an interaction with E)
C→K operations	Formulation of logical questions to activate K or the valuation of an undecidable proposition (to end the design process)
C→C operations	Relating concept graphs to form the treelike structure of C space
K→E operations	Inquiry of external databases, other design assistants, etc. and suggestions to the user
E→K operations	Acquiring knowledge as a result of observing the user or inquiring other entities

4.1.3 Orienting knowledge expansion

When the user creates a concept or operates an expansive partition, the assistant can prompt the user for communicating which are the knowledge expansions necessary for being able to valuate that concept. For the concept (*lion* ® *eagle*) , this means the assistant will report undecidable propositions such as (*lion* ® *flying*) or (*bird* ® *animal*) . More generally, these propositions correspond to those that connect the disjoint components of the graph that the concept is connecting. For the user, it might be easier to acquire knowledge on some of these propositions rather than others.

For instance, it might be easier to find out whether birds are animals than to find flying lions. Different computational approaches can be devised so that an assistant would be able to evaluate, at least to some extent, which knowledge expansions are easier to realize and report to the user these expansion possibilities. Also, assuming that the assistant has access to entities other than the user (a wider environment space such as databases, other assistants.), it can carry out by itself some of the necessary knowledge expansions.

4.1.4 Inference and revision

NAL offers an intuitive inference schemes that consists in creating associations between terms. In this regard, the most important characteristic offered by NAL is that the propositions resulting from

inference have truth values that are almost always inferior to the parent arguments (with the exception of revision, see paragraph 2.2.3 and below). As a result, a system based on NAL do not become overconfident in its inference results and the longer the inference chain become, the more *tentative* its consequence will be. Nevertheless, with its associative inference scheme NAL may discover some direct consequence of a given K space configuration which has not been realised yet by its user.
The term logic we used to model CK theory provides a natural facility to revise beliefs: the semantics of the logic is built on the notion of evidence and as new knowledge becomes available (from an interaction with an E space), it may be naturally incorporated within K space by a belief revision rule (Wang, 1995, 2006). As new and old evidence are combined, the confidence value obtained by revision is higher than both the confidence of the parent arguments (Wang, 1995, 2006). Thus, as more evidence is discovered (e.g., through simulation or prototyping), the system become more confident about a relation. The possibility to revise beliefs allows reinterpreting the co-evolution of C and K spaces as a duality of *object revision and belief revision*.

4.2 Weak and strict forms of design: an arising issue

There is more in using continuous truth values than it appears in a first glance: for instance, we can now talk about weak truth values or strong beliefs. What does this tells us about types of design reasoning a design assistant might support? Instead of defining concepts as undecidable propositions, it becomes possible to accept propositions with weak values of frequency and confidence levels as *pseudo*-concepts and to look for ways of increasing these values using K space and interaction with the environment E! Consider, for example, the proposition (*lion* ® *eagle*) < 1; 0.42 > which is now known in K space. Is it possible to increase the confidence level of this proposition? Such situations may arise when trying to optimize an already known design solution. The formalism of NAL indicates easily which are the propositions to be revised in order to change the status of this proposition such as (*bird* ® *animal*) < 1; 0.9 > or (*eagle* ® *animal*) < 1; 0.81 > . For instance, if the confidence of the latter can be increased to 0.9, the confidence of the (*lion* ® *eagle*) is increased to 0.45. If the confidence of the former can be increased to 0,99, the final confidence obtained is 0,47. This fine control over the knowledge and its consequences offers the possibility to choose which type of knowledge must be sought after and what is the potential gain.
More generally, arbitrary threshold levels on frequency and confidence levels allow to model the point beyond which a proposition is considered as a concept and should be investigated. Consider (*lion* ® *eagle*) < *a*; *b* > with *a* £ e_1 and *b* £ e_2 where e_1 Î [0, 1] and e_2 Î [0, 1] are thresholds: what if we make the thresholds vary? In fact, we see that the use of continuous truth values for modelling K space allows *blurring the distinction of C and K spaces!* Then, a user can choose at which level of refinement he or she will work; whether he or she will realize a strict design (only undecidable propositions are considered concepts and expansive partitions) or realize a weak form of design by pursuing the elaboration of a pseudo-concept (any part of the knowledge space can be considered as a concept by setting appropriate threshold levels). Changing these thresholds dynamically would allow the designer to switch back and forth between these alternative design strategies.
The distinction between weak and strict forms of design is an intriguing issue for design practice as well: are there forms of design corresponding to this distinction and, if yes, what is the impact of these two distinct strategies to design performance and innovation? This is an interesting point that shall be pursued in future work.

5. Results and future work

The paper introduced the notion of "models of CK theory" and raised the question "Are there models of CK theory with suitable computational properties allowing an implementation of design assistants". We analyzed a particular model based on NAL, a formal term logic. We have shown that NAL has some desirable properties for building design assistants and that, using NAL, it is possible to capture all the fundamental notions of CK theory. A description of how a NAL based system can assist its user

in designing has been given. In particular, we have seen such a system can support the dual expansion of concept and knowledge spaces; both in weak and strong forms of design.
Many questions such as the control mechanisms for NAL based assistants, their real time performance and the effects of the cooperation between designers and assistants have not been considered in the current work. Each of these specific points is being pursued (see e.g. Kazakci, 2005, 2007) and some experiments have already been realized with the help of engineering design students of Istanbul Technical University (Kazakci, 2007). These points will be addressed in future work.

References

Ben-Mahmoud Jouini, S., Charue-Duboc, F. and Fourcade, F. (2007). Managing creativity process in innovation driven competition. International Product Development Management Conference.
Cohen, J. P. (1963). The independance of the continuum hypothesis I. Procedings of the National Academy of Science, USA. pp. 1143-1148.
Cohen, J. P. (1964). The independance of the continuum hypothesis II. Procedings of the National Academy of Science, USA. pp. 105-110.
Elmquist, M. and Segrestin, B. (2007) Towards a new logic of front end management: from drug discovery to drug design in pharmaceutical R&D. Journal of Creativity and Innovation Management, ***16*** *(2),*
Englebretsen, G. (1981) Three Logicians: Aristotle, Leibniz and Sommers, and the Syllogistic, Van Gorcum, Assen, The Netherlands.
Hatchuel, A. and Weil, B. (1999). Pour une théorie unifiée de la conception, Axiomatiques et processus collectifs. CGS Ecole des Mines, GIS cognition-CNRS. pp. 1-27.
Hatchuel, A. and Weil, B. (2002). La théorie C-K: Fondaments et usages d'une théorie unifiée de la conception. Colloque Sciences de la Conception, Lyon.
Hatchuel, A. and Weil, B. (2003). A new approach of innovative design : an introduction to C-K design theory. ICED'03, Stockholm, Sweden., pp. 14.
Hatchuel, A., Weil, B. and Le Masson, P. (2007). Design as Forcing:Deepening the foundations of Ck theory. 16th International Conference on Engineering Design - ICED 2007, Knowledge, Innovation and Sustainability, Paris, France.
Kazakci, A. (2005). DesigNAR, An intelligent design assistant based on C-K design theory. IDETC/CIE 2005 ASME 2005 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference, Long Beach, California, USA.
Kazakci, A. (2007) La théorie CKE comme fondement théorique pour les assistants de conception: DesigNAR; un assistant de synthèse de concept basé sur la théorie CKE, LAMSADE, Université Paris IX Dauphine, Thèse de Doctorat, Paris.
Kazakci, A. and Tsoukiàs, A. (2005) Extending the C-K design theory: A theoretical background for personal design assistants. Journal of Engineering Design, ***16*** *(4), pp. 399-411.*
Wang, P. (1995) Non-axiomatic reasoning system: Exploring the essence of intelligence, Indiana University, PhD Thesis, Indiana.
Wang, P. (1998). Grounding the meaning of symbols on the system's experience. Working Notes of the AAAI Workshop on the Grounding of Word Meaning: Data and Models, Madison, Wisconsin. pp. 23--24.
Wang, P. (2005) Experience-Grounded Semantics: A theory for intelligent systems. Cognitive Systems Research, ***6*** *(4), pp. 282-302.*
Wang, P. (2006) Rigid flexibility : the logic of intelligence, Springer, Dordrecht. 414 pages.

Dr. Akin Osman KAZAKCI
Postdoctoral Research Fellow
CGS - Centre de Gestion Scientifique
Ecole des Mines de Paris
60 boulevard Saint-Michel, 75272 PARIS Cedex 06
Tel.: 00 33 1 40 51 92 08
Fax.: 00 33 1 40 51 90 65
Email: kazakci@lamsade.dauphine.fr
URL: http://www.lamsade.dauphine.fr/~kazakci