The Use of Hybrid Technology in the Construction of an Evolving Knowledge-Base Design system.

Neander F Silva

Department of Architecture & Building Science
University of Strathclyde
Glasgow
UK

This paper focus in one vital aspect of design computing: the knowledge-base extension and maintenance. It describes a hybrid approach where a rudimentary evolving knowledge-base design system is proposed. It draws inspiration from three areas of artificial intelligence: knowledge-base systems, connectionist models, and case-based reasoning. Its main contributions are: an incremental self-adjustment able to minimise substantially the dependency on knowledge engineer intervention, and an interactive support to innovation.

Keywords: precedents, connectionism, knowledge-bases maintenance, innovative design.

1 Introduction

Research in Computer Aided Architectural Design, CAAD, has seen, in recent years, a growing interest in the role played by previous knowledge in the design process. It is relatively well accepted now that design does not rely only on methods but also on a process of adaptation, transformation and re-creation based upon previous design experiences [3, 5, 6, 10, 11, 12, 14].

However, if design does rely on precedents it is also well established that it is essentially non-monotonic: the domain knowledge is changeable. That is, what is known about the design objects changes, even contradicting what was known before, as more knowledge is acquired, new techniques and paradigms are introduced or as the design context changes.

Research in design theory, CAAD and Artificial Intelligence, Al, has produced models and systems providing either interactive interfaces, but with static knowledgebases, hard to be consistently extended without the intervention of a knowledge engineer, or knowledge-acquisition tools with passive and awkward interfaces. They have failed to deliver consistently unified models [7]

Therefore, this paper focus in one vital aspect of design modelling: the knowledgebase extension and maintenance. The necessary conditions to allow the continuous extension of knowledge-bases, with minimum knowledge-engineer intervention, while at the same time preserving their original consistency, are the ability to generate reliable new solutions in a knowledge-representation scheme compatible with the existing one. The testing of these conditions and the delivery of a hybrid model that complies with those requirements were the goals of the research described in this paper.

The proposed model draws inspiration from three approaches in artificial intelligence: knowledge-base systems [8, 9], connectionist models [1, 2] and case-based reasoning [3, 10, 11, 12, 14].

What is being proposed here is not the simply gathering of three well-known processes, but a hybrid approach that also makes contributions to the three mentioned techniques. It differs fundamentally from conventional implementations of those approaches, although strongly inspired in their underlying theories. These are the main

contributions to knowledge: firstly, the knowledge-based system module is not a rule-based one. It is a unique implementation hosted on an innovative pattern-recognition-based shell developed at the University of Strathclyde in the early 90's [8, 9]. Secondly, the connectionist system does not operate at the front-end, but in the background of an intelligent interface. Its method of optimisation and validation is also innovative. It does not perform the classification of unknown cases in the light of known ones, it rather learns general trends to produce suggestions of new cases. Its validation process is unconventional because is based on a verification of result's likelihood rather than on specific unknown cases testing. Thirdly, and the most important contribution, are its results. It is, to the best of my knowledge, the first model to offer an incremental selfadjustment able to minimise substantially the dependency on knowledge engineer intervention, and to provide interactive support to innovation.

2 Knowledge representation issues and alternative knowledge-base system shells

The addition of new knowledge in a conventional rule-based system can make its knowledge-base inconsistent, requiring a complete review of the knowledge already in the system, particularly if it is a solution that represents an exception to a more general rule in the system. A simple example of this may be an animal classification domain, where the addition of the solution duck-billed platypus, the only egg-laying mammal in the world, may contradict general rules stating that "no mammals lay eggs" and 'birds do not give milk". A solution is to link the duck-billed platypus to the super classes 'mammals' and 'birds' at the same time, to overcome the inconsistency. Therefore, new exemption rules need to be created and the old ones revised. This makes a conventional rule-base system extremely dependent of a knowledge engineer maintenance intervention. Alternative knowledge-base systems, based on pattern-recognition rather than on rules, have been devised [4, 8, 9]. Mustoe [8] argues that evidently the function of the network of rules in conventional knowledge-based systems is to place a set of individual productions into a correct relationship with a particular solution. In other words, solutions are classified according to their question set, while questions are classified by reference to the solutions that they verify. Therefore, knowledge is represented in his system through a Boolean classification structure in which all the solutions are classified independently from each other in a set of binary relationships with the questions that verify them. A drawback is that, in complex domains, a huge number of descriptors, or questions, can be generated by this approach. However, this is largely compensated by the Popperian [13] control system. The control algorithm of his knowledge-base system functions by rejecting falsified solutionx. It will finish either with one or more unfalsified solutions, or with a confession that it cannot find a solution within that particular domain. it operates in a process of zeroing-in upon a successively shortened list of still-possible solutions. It might be argued tat this procedure does not actuatlly model thougt. However, it does simulate the behaviour of a human expert, who will begin by overviewing the scope of the problem before concentrating his or her attention upon the most promising of the remaining solutions.

If no inconsistency is necessarily introduced on the knowledge-base already in the system with the addition of new solutions, new horizons are open for building systems in which the knowledge-bases can be consistently expanded. However, as the options must be manually set before hand, the system devised by Mustoe [8, 9] has no inherent knowledgeacquisition mechanism as any other knowledge-base system.

3 Connectionist models, innovation and interface problems

Connectionist models, as demonstrated by Coyne & Newton [1] and Coyne & Yokozawa [2], can provide support for the exploration of implicit, therefore, unexpected connections among information. An adaptive behaviour can be achieved through supervised learning. An autoassociative model, where solution's features map into themselves, as input and output, was used in their experiments. Figure 2 illustrates this design.

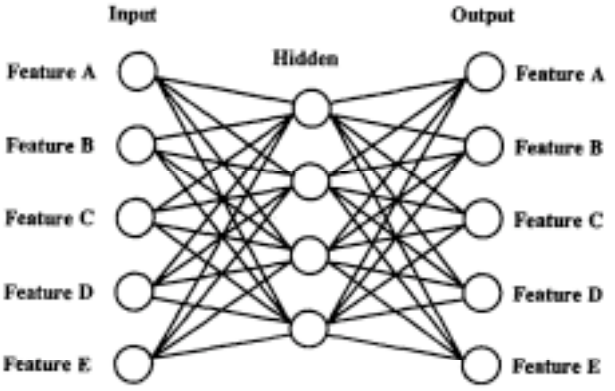

Figure 1. An autoassociative network.

The use of this design, might not be obvious. However, what happens with this network it is that, after exposure to a certain number of examples, represented by different sets of combinations of features, the system becomes aware of what features are mutually excitatory or inhibitory. In other words, it knows whether or not each pair of features tending to occur together in the same solutions.

Coyne & Yokozawa [2] implemented their model as a front-end mechanism, where the designer dealt with the atomic statements directly in a somewhat passive and awkward interface.

4 A precedent-based, self-evolving, innovation-supportive system

A pattern-recognition-based system built upon Mustoes foundations [8, 9], very often finds a solution with few questions, either without having to ask all questions in the questions set or without requiring the user to answer all the questions that were brought to the screen. What happens in these situations is that the user makes some basic, but definite choices among those features represented by the questions the system brings to the screen. For instance, the system behaviours on the following way: for a set of cases described by the features A, B, C, D, F, G, H, I, J, K, L, M, and given that the user answers by saying he or she wants a case in which B and G are present, A and D are absent, and he or she does not know if F is to be present, the system replies by informing that considering the solutions in the knowledge-base the most likely solution is the one in which the features B, E, F, G, K, L, are present.

However, a binary connectionist system, trained with the same set of solutions already in knowledge-base system can produce outputs that may represent new combinations of features. These new combinations of features have a format that compatible with the existing knowledge-base representation.

4.1 Algorithm outline

The central hypothesis of this paper is: if a trained connectionist sub-system is controled by an intelligent front-end inspired on Mustoes [8, 9] model, and operates over the same users input, new combinations of features will emerge and can be added to both sub-systems providing the basis of an evolving knowledge-base system. Following the example above, with B and G definitely activated, A and 0 definitely inactivated, the system may answer that the most likely combination of features is either the same provided by Cortex, B, E, F, G, K, L, or a combination of B, C, E, G, J, L. The outcome depends on two factors, basically: the strength of the connections established by the connectionst subsystem after training and which features the user selected at the interactive front-end of the pattern-recognition-based sub-system. The knowledge-base system acts interactively with the user by asking questions. The user's input is sent to the knowledge-base system that redirects it to the neural network. The first searches for existing cases satisfying the user's requirements. The second trains itself with the case-base and produces inputs considering

the general trends in the sample. If both results are identical only this known case is presented to the user. If they are different, both are presented to the user. Figure 3 shows a flow chart of this research prototyping scheme.

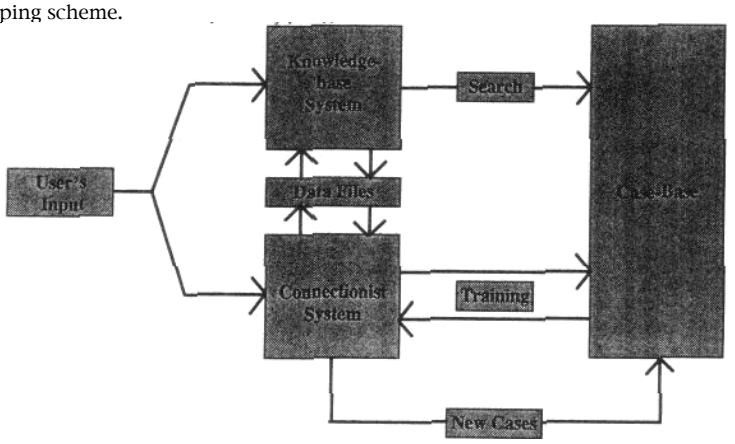

Figure 2. Prototyping scheme.

5 Experimentation

5.1 Methodology

The verification of the hypothesis was developed by experimentally inspecting the consistency and relevance of those new solutions, in terms of architectural knowledge, and also by verifying how often they might occur. The building of a knowledge-base into a loose prototype that could test the logic and potential of the hybrid approach was necessary.

A representative sample of all possible user s answers to the knowledge-base system, built on a combinatory basis, was not a goal. The sample shown referred to in this paper was not built by combining 'yesses' and nos at random, but through a conceptual analysis of the architectural meaning of each question presented by the front-end in the context of a particular design task. The same approach was adopted in verifying the consistency of the combinations of features generated by the neural network. The checking of the absence of mutually exclusive features and initial selections persistency is part of the methodoloy. However, it is the architectural meaning of each of these situations that is actually taken into final consideration.

The efficacy of the procedures stated in the algorithm outline were verified by executing its actions into separate but loosely integrated modules and by keeping track of its results.

5.2 Data preparation

A set of 122 instances of entrance doors was collected from buildings at the University of Strathclyde, Glasgow city centre, Glasgow University and architectural magazines. No pre-set conditions were established in this task. The only constraints were the technical and legal possibilities of photographing them. Apart from this, the selection of instances was undertaken at random. At this point it is important to emphasise that, despite the similarities, the resulting sample has in common with the one used by Coyne [2] only the theme. Besides the obviously distinct instances, the description of the problem is much deeper than the one used by Coyne.

The resulting sample of 122 instances was binary described according to 80 features. Features can represent either abstract concepts, such as flow function', or geometric characteristics such as "top flat moulding". It may be argued that the majority of the descriptors of the sample are geometric features. However, I believe this is proper of the design task at hand and its level of abstraction. It is a well-known trend that design moves away from high level of abstraction to low level as the process moves from conceptual

design to detailed design. Therefore, an entrance door generally represents a task closer to detailed design than conceptual.

A careful knowledge representation was very important. Connectionist systems have their learning capability as their most attractive and useful feature. However, it is naive to suppose that we can simply input data and expect them to find all the relevant excitatory and inhibitory connections. They do have limitations, but it is the identification of those limits that provide the means of exploiting their best features properly and, more important, the means of building improved algorithms. Connectionist models have particular difficult to deal with features in different levels of abstraction, that is, the difficult in establishing the proper connections between concepts like "internal flat layout" and the 'building overall style". Coyne explores this problem and suggests it can pre-empty the emergence of new schemes [2]. I would go further and say that it can also lead to the establishment of undesirable connections. There are only two possibilities: first, this kind of connection can never be established in certain cases and, therefore they should be left outside the set of descriptors. Second, that this kind of connection can only be properly established by building carefully a bridge between the two features. This is applicable to cases where a connection does exist in the real world. It can be achieved by extending the number of descriptors through the breaking down the features into lower level of abstraction ones. Them the descriptors that became simply labels of sub-sets of other ones in the set are removed. Their meanings are still represented there implicitly and can be used if necessary without generating undesirable connections.

5.2.1 General trends in the chosen sample

A frequency table reporting the presence of each feature 'y" given that "x" is present was built. It is a matrix of 80 columns and 80 rows. Each row represents the frequencies of each column label feature given that the row label feature is present. For instance, if the feature "Flow function: main entrance" is present, the features "Flow function: secondary entrance", "Flow control: public access" and "flow control: restricted access" have a probability of 0.0%, 60.78% and 39.22%, respectively, of being present. The probability values in that table were normalised to values between 0 and 1 to facilitate the comparison with the network settings later on.

5.2.2 Domain knowledge setup and data collection

The feature's descriptors were converted into questions in the setting up of a domain knowledge-base.

It was also necessary to generate a representative sample of users' inputs and knowledge-base system outputs. A set of 46 "user's profiles" was built covering a wide variety of different preoccupation and contexts. As mentioned earlier, this sample of user's preoccupation or expectations was not built at random, but on the basis of the architectural meaning of each question presented by the system in the context of a particular design task.

A second aim of this set of "user profiles" is to demonstrate the richness of the knowledge represented in domain. The "user profiles" show that, with the set of descriptors used in the sample, it is possible to construct a wide variety of prototypes representing different preoccupation. Is also important to emphasise that it is possible to build combinations of features mapping into categories and concepts not explicitly represented in the sample, such as "modem high-tech". It may be argued that it is difficult to distinguish a "modern high-tech" entrance from a "modern functionalist" one. However, the "user profiles" were not built with the intention of providing a complete description of what "modem hightech" architecture or "modern functionalist" architecture may be. They were built with the intention of providing a description of how a designer, with any of those profiles, would answer to those questions constrained to the boundaries of the specific design problem "entrance door".

5.2.3 Connectionist model design, training, optimisation and validation

An innovative method of design and validation was developed in the connectionist sub-system. It did not aim the classification of unknown cases in the light of known ones as common in other applications such as computer vision. In those domains cases unknown are presented to the trained neural network to verify if it is able to associate a particular existing pattern with the correct concept or label.

However, the objective here was to tram the connectionist model to create the new or unknown cases themselves. Its optimisation and validation processes were unconventional and based in general results likelihood verification rather than on specific unknown cases testing. Therefore, a series of 20 connectionist systems was created in the search for the optimum one.

The variation in their design resided on the amount of hidden nodes or units. The training started with the minimum possible setup of 80 input, no hidden, 80 outputs. In the next stages one hidden layer was included, starting with 10 nodes and going up to 130. The neural networks unable to reach the learning rate were discarded. This happened with only two: those with 10 and 20 hidden nodes. Therefore, 18 remained to be validated. Each network generates weight connections matrixes. The first one, with no hidden nodes, generates one matrix of 81 columns and 80 rows. Each row represents the 80 connection's weights between an output node and every node in the input layer. The 81st value in each row represents the threshold.

The networks with one hidden generate two weight connection matrixes each. For instance, in the network with 50 hidden nodes the first matrix will have 81 columns and a number of rows equal to the number of hidden nodes, that is 50. Each row represents the 80 connection's weights between a hidden node and every node in the input layer. Again, the 81st value in each row represents the threshold. The second matrix will have a number of columns equal to the number of hidden nodes, that are, 50, plus one for the threshold, and 80 rows. Each row represents the connection's weight between an output node and every node in the hidden layer.

The connection's weight matrix of the network with no hidden layer provides a good idea of what features might be mutually excitatory or inhibitory. This is due to the fact that there is only one way through which each input unit can influence each output unit. The higher the weight the more mutually supportive are the units connected. The lower the weight the more mutually excluding are the units connected.

However, the interpretation of the weight matrixes for the networks with one hidden layer becomes much more complex and less straightforward. An input unit can influence the setting of an output unit in as many ways as the number hidden units. Therefore, it was clear that a better and more straightforward method of validation was supposed to be found to determine which network represented the optimum learning of the general trends in the sample. If it is assumed that the frequency table constitutes a good indicator of what are the general trends in the sample, then the most reliable network should be the one with the closest likely results to that table or at least the one less prone to contradict it.

Tables with values of support varying between "0" and "1" can be produced, for each network by replacing the threshold step function by a sigmoid function, and by activating each input feature at a time.

A simple method of measurement was then applied to verify which network would be the closest to the frequency matrix. A proximity rate was calculated by subtracting from each level of support value found in the previous experiment, the correspondent frequency value. The sum of all the absolute values of those subtractions was then divide by the total number of operations to find an average. A network that tends to activate a unit with low frequency will receive a higher rate then the another that does not. A network that tends to inactivate a unit with high frequency will also receive a higher rate than another that does not. Therefore, the lower the rate the closer is the network to the general trends in the sample. This experiment gave a better idea of where the optimum network might be than the number of activations. Figure 3 shows a table with the resulting proximity rates.

Number of Hidden Nodes	0	30	40	50	55	60	65	70	75
Proximity rate	*0.3798*	*0.1340*	*0.1160*	*0.1155*	*0.1162*	*0.1195*	*0.1193*	*0.1258*	*0.1267*

Number of Hidden Nodes	80	85	90	95	100	105	110	120	130
Proximity rate	*0.1265*	*0.1236*	*0.1311*	*0.1260*	*0.1347*	*0.1243*	*0.1382*	*0.1353*	*0.1494*

Figure 3. Proximity rates.

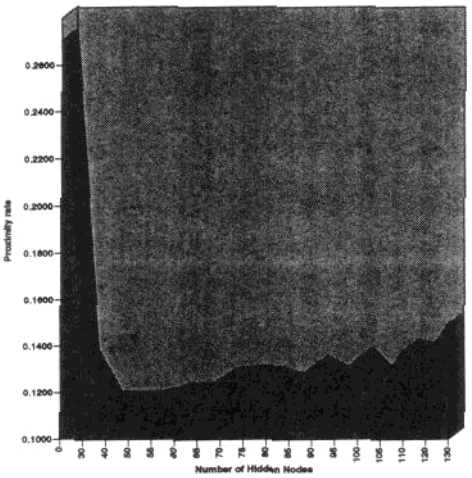

Figure 4 shows a graph of the relationship number of hidden nodes and proximity rate.

Figure 4. Relation between the number of hidden nodes and the proximity rate more clearly. More sophisticated and accurate methods of validation and optimisation can certainly be developed. What is important to emphasise here is that it is possible to construct a model of validation based on likelihood of results rather than on unknown cases testing. For the time being the described method is satisfactory because it provides a reasonable method for finding the optimum network. Therefore, the network with 50 hidden nodes was selected for experimentation because it was the closest to the general trends in the sample.

5.3 Experimentation and result analysis:

The 46 users profiles mentioned before were used for collecting data from the knowledge-base sub-system. This data is composed of the knowledge-base system questions traces and the users answers, as well as the solutions offered by the system.

The optimised network was tested by reproducing the procedures described in the algorithm. 9 were situations in which new cases emerged. These situations are those satisfying all the conditions controled by the proposed algorithm, that are: first, the solution provide by the neural network should be different from the one retrieved from the knowledge-base; second, the solution provided by the neural network should not contradict, at the end of the running process, any of the initial users choices. The third criteria of evaluation, not explicitly controlled by the algorithm, was the architectural consistency of the new solutions. But this was exactly one of the main aims of this research: to experimentally verify the ability of the proposed system to implicitly control those inconsistencies. At the outset of this research the possibility of new schemes emergence was thought to be something likely to happen rarely. This expectation was based mainly on the work of Coyne [1], one of the most important recent researches on connectionist models applied to architectural design. It was-believed that the neural network outcome would generally match with a case already in memory, and only exceptionally would produce a new solution. However, the rate of new solutions emergence in the present sample, almost 20%, does not suggest that the occurrence is just exceptional. I think there are two main reasons for this: first, the solution found by the knowledge-base module is based on the closest match of existing unique cases with the users request, which may have features that represent exceptions. Second, the solution presented by the connectionist module is based on generalisation, which takes into consideration the strongest trends in the sample that are compatible with the users request.

Figure 5 shows a table with the occurrence of new solutions in the whole testing sample and among those profiles with and without uncertainty.

Category		Whole Sample (1)		Among cases with no uncertainty (2)		Among cases with uncertainty (3)	
		#	%	#	%	#	%
Total of new solutions (a)		9	19.6	7	33.3	3	12.0
New Solutions	Consistent	9	19.6	7	33.3	3	12.0
	Inconsistent	0	0.0	0	0	0	0.0

(1) New solutions in the whole sample of 46 user types.
(2) New solutions among the 21 cases in which the user did not answer any questions with a 'don't know'.
(3) New solutions among the 25 cases in which the user answered at least one question with a 'don't know".

Figure 5. Occurence of new solutions.

These results do not provide conclusive clue about the relationship between the occurrence of new solutions and the presence uncertainty in the users answers. However, it is important to notice the absence of architecturally absurd combinations of features. There seems to exist some sort of relation between the users uncertainty and levels of innovation in the new solutions, as shown in the table of Figure 6.

User Type Generator	New Solution Number	Questions from KBS			Level of Innovation		
		Total Number of Questions	Number of Don't Knows	Percentage of Uncertainty	Number of Active Nodes in union set: (a U b)sc*	Number of differences from KBS solution (d)	Innovation Percentage of (d) out of (c)
Gothic	1	11	0	*0.0*	17	6	*35.3*
Art Nouveau 2	2	11	3	*27.2*	19	4	*21.1*
Modern (functionalist 1)	3	13	0	*0.0*	15	3	*20.0*
Modern (brutalist 2)	4	15	0	*0.0*	14	1	*7.1*
Modern (high tech 1)	5	10	0	*0.0*	23	12	*52.2*
Modern (high tech 2)	6	13	2	*15.4*	22	11	*50.0*
Modern (high tech 3)	7	13	0	*0.0*	16	3	*18.8*
Post-Modern (eclectic 2)	8	13	0	*0.0*	17	4	*23.5*
Environmental control 4 (loose)	9	31	24	*77.4*	22	15	*68.2*

(*) Where 'a' is the set of present features in KBS solution and 'b' is the set of present features in new solution.

Figure 6. Relation between uncertainty and innovation.

The graph in Figure 7 confirms more clearly that there is at least a moderate relationship between the level of uncertainty in the users answers and the level of innovation in the new solutions.

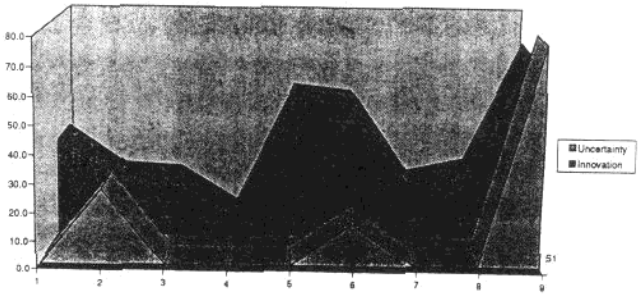

Figure 7. Uncertainty and innovation.

Two of the new solutions, numbers 5 and 6, suggested by the neural network together with their knowledge-base system counterparts are show in figures 8 and 9.

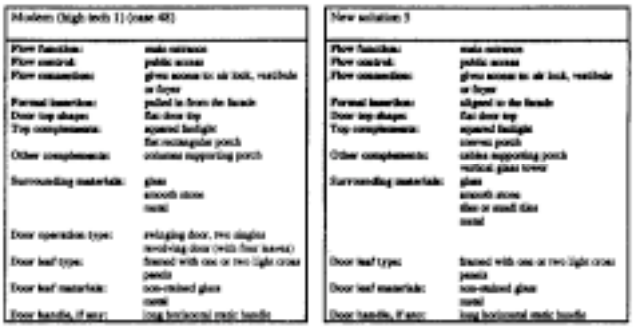

Figure 8. Existing and new solution 5.

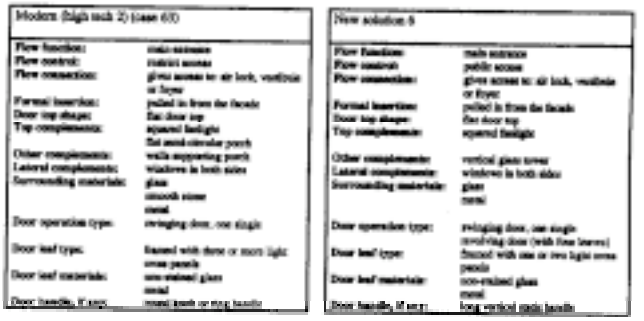

Figure 9. Existing and new solution 6.

The solutions, with good level of innovation, provide some interesting suggestions. "New solution 5", for instance suggests the present of some features such as "cables supporting porch" and "vertical glass tower" that make it closer to the user's profile than the solution suggested by the knowledge-base system. The same happens with "New solution 6", with the proposed absence of 'smooth stone" in the surrounding materials, the replacement of "round knob" by "long vertical static handle" and the suggested presence of "vertical glass tower" and "revolving door". All the new solutions are eligible to be added to the existing knowledge-base because their Boolean knowledge representation does not impose risk of inconsistency addition.

5.4 Relevance and implications

It was not a goal in this research to provide an algorithm for radical changes, but for an incremental self-adjusting model able to minimise substantially the knowledge engineer intervention dependency, and to provide interactive support to innovative design thinking.

I believe that the experimentation described in this paper strongly suggests that the necessary implementation conditions have been achieved: the ability to generate reliable new solutions in a knowledge representation schema that allows the continuous extension of the knowledge-base while at the same time preserving its consistency.

The system described in this paper does not have a graphical component. It was focused mainly on design though support, rather than on straightforward geometric generation. However, the results are promising in terms of a design decision-making supportive system.

Further research could investigate the means of integrating the system with parametrized types in shape grammars towards generative systems. Therefore, models not only suggesting textual descriptions of possible new solutions, but also graphical sketches could be developed.

6 Bibliography

[1] Coyne, R. D., Newton, S. and Sudweeks, F. (1993) A Connectionist View of Creative Design Reasoning, in Gero, J. S. and Maher, M. L. (Eds.) Modeling Creativity and Knowledge-Based Creative Design, Lawrence Erlbaum Associates, Publishers, Hillsdale, NJ.

[2] Coyne, R. D. & M. Yokozawa. 1992. Computer assistance in designing from precedent. In Environment and Planning B: Planning and Design, (19):143-171.

[3] Domeshek, E. A. and Kolodner, J. L. (1992) A case-based design aid for architecture, in Gero, J. S. (editor), Artificial Intelligence in Design "92, proceedings, Kluwer Academic Publishers, Dordrecht, The Netherlands.

[4] Frey, P. 1986. A bit-mapped classifier. In Byte, New York: McGraw-Hill, November issue.

[5] Koutamanis, A. 1993. The future of visual representations in architecture. In Automation in Construction, 2(1):47-56.

[6] Kuhn, C. & M. Herzog. 1993. Modelling the representation of architectural design cases. In Automation in Construction, 2(1):1-10.

[7] Liebowitz, J. 1993. Roll your own hybrids. In Byte, New York: McGraw-Hill, July issue. [8] Mustoe, J. 1990. Artificial Intelligence and its Applications in Architectural Design, Ph.D. thesis, Glasgow: University of Strathclyde.

[9] Mustoe, J. 1993. Cortex User's Guide version 1.5. Nottingham, UK, Resolution Software.

[10] Oxman, Rivka. 1990. "Prior knowledge in design: a dynamic knowledge-based model of design and creativity". In Design Studies, vol. 11, number 1.

[11] Oxman, Rivka & Robert Oxman. 1993a. Remembrance of things past: design precedents in libraries. In Automation in Construction, 2(1):21-29.

[12] Oxman, Rivka and Robert Oxman. 1993b. Precedents: Memory Structure in Design Case Libraries. In U. Flemming and S. V. Wyk (editors) CAAD Futures '93, proceedings, Amsterdam: NorthHolland.

[13] Popper, K. R. (1934) Logic der Forschung in Frank, P., and Schlick, M. (Eds.) Schriften zur wissenschaftlichen Weltauffassung, Verlag von Julius Springer, Vienna. Translated by K. R. Popper, J. Freed and L. Freed as "The Logic of Scientific Discovery" and published by Hutchinson and CO Ltd, London, 1959.

[14] Schmitt, G. 1993. Case-Based Design and creativity. In Automation in Construction, 2(1):11-19.

7 Acknowledgements

This research was supported by CAPES, The Brazilian Federal Agency for PostGraduate Education. I am indebted to Dr. Alan H. Bridges from Strathclyde University for his always wise suggestions.