Transactions on Information and Communications Technologies vol 20, © 1998 WIT Press, www.witpress.com, ISSN 1743-3517

Producing designs of physical systems through model transmutations

Elio Toppano
Dipartimento di Matematica e Informatica, Università di Udine
Via delle Scienze 206, Loc. Rizzi, 33100 Udine, Italia
EMail: Elio.Toppano@uniud.it

Abstract

In this paper, engineering design is considered from the point of view of modeling i.e. the construction and manipulation of models of (possible) physical realities. Consequently, the design activity has been analysed in terms of patterns of inference called model transmutations. Three categories of transmutations namely, transformations, combinations and prototype copying, have been discussed with reference to the Multimodeling approach for representing physical systems. The major goal of the paper is to provide a conceptual framework for analysing existing design systems and for addressing questions concerning their competence such as what types of inference patterns underlie different design strategies e.g. case-based, compositional and analogical design; what kind of design solutions a design system is able to generate from what kind of input specification and prior design knowledge; what is the logical relationship between specification and prior design knowledge. A second goal is to provide a basis for the development of a general theory for reactive multistrategy design that aims at combining a range of different design strategies dynamically, in order to take advantage of their respective strengths and address a wider range of practical problems.

1 Introduction

Engineering design can be abstractly characterized as a constrained function-to-structure mapping [7], [8], [16]. It takes as input a functional specification of the artifact to be built, including desired goals and constraints on design, and a description of the available technology and of general physical principles. It produces as output a description of an artifact that satisfies the specification and contains enough information to allow the manufacturing, fabrication or

construction of the desired system. One method for solving design problems is PCM i.e. propose, critique, and modify [4]. The method have the sub tasks of proposing partial or complete design solutions, verifying proposed solutions by identifying causes of failure if any, and modifying proposals to satisfy design goals.

In this paper, we characterize the PCM method from the point of view of modeling i.e. the construction and manipulation of models of (possible) physical realities. Under this perspective the design process consists of a sequence of cycles. In each cycle the designer analyses the current design solution (i.e. a possibly incomplete model of the desired artifact) in terms of his/her background knowledge and problem specification and decides which action (i.e. model manipulation) to do next in order to improve the solution. As a domain of application we have chosen electrical and fluid-mechanical devices whose schematic description can be expressed as a network of lumped-parameter idealized elements. We are mainly concerned with the early phase of design that is, conceptual design, which results into the topology of the desired artifact without stating definite values for all constructive parameters (e.g., geometrical, physical, etc.).

One goal of the paper is to identify what kind of inferences, thereafter called model transmutations, underlie different design processes such as case-based, compositional and analogical design. The analysis is aimed at providing a conceptual framework by means of which: i) the design systems proposed in literature can be studied and compared on the base of the inference capabilities they presuppose i.e. on the base of their levels of competence; ii) model transmutations can be used as building blocks to explain or experiment with different design strategies. A second goal is to identify what is the logical relationship among model transmutations, input information (i.e. design specification) and the designer's prior knowledge in order to understand the preconditions and the circumstances under which each transmutation can be used. The presented ideas provide a basis for the development of a general theory for reactive multistrategy design. By multistrategy design we intend the composition of two or more types of inferences in the same design process. The composition is made dynamically as the design process unfold according to the demands of the current situation.

The paper is organized as follows. In section 2 we illustrate the basic elements of the representation system used to describe design solutions. We then introduce four basic elements of design problems - namely design specification, operational model, background knowledge and design operator - which are used to model the design process (sections 3). The next two sections are devoted to illustrate basic transmutations and to show some examples of multitype inferences using transmutations. Finally, section 6 illustrates the main features of SECS, a system developed to experiment with multistrategy design in the

electrical fluid-mechanical domains, while section 7 discusses related work and draws conclusions.

2 Representing design solutions: design fragments and prototypes

The following definitions introduce the basic elements of our representation system which is based on the Multimodeling approach for reasoning about artifacts that we proposed in recent years [5].

• *Conceptual Scheme*. A conceptual scheme, CS, is defined by a triplet <E, R, A/D> where: E is a set of entity types, R is a set of relation types, and A is a set of attributes representing properties of entities or relations. Each attribute has an associated domain (D) specifying the range of values the attribute may take. Conceptual schemes are selected from among a prefixed set of alternatives varying in perspective. A *perspective* is a choice of values on the following orthogonal dimensions:

- ontology: the category of entities, relations, and properties that are presupposed to exist in the world and, thus, has been included in the scheme, for example, macroscopic, microscopic, atomic or subatomic entities. Several ontologies have been proposed to represent physical systems. See [2] for a thorough discussion of various alternatives;
- epistemological type: the type of knowledge the entities and the relations of a scheme allow to represent about reality. We identified four epistemological types [5]:
  1. Structural knowledge i.e. knowledge about system topology. This type of knowledge describes which components constitute the system and how they are connected to each other;
  2. Behavioral knowledge i.e. knowledge about the potential behavior of components. This type of knowledge describes how components can work and interact in terms of the physical quantities that characterize their state and the physical laws that rule their operation;
  3. Functional knowledge i.e., knowledge about the roles components may play in the physical processes and phenomena in which they take part. In the multimodeling approach, functional knowledge deals with: i) generalized substances and currents (e.g., electrical charge and current, angular displacement and velocity, angular momentum and torque), ii) functional roles (e.g. the generator, the conduit, the barrier, the reservoir), iii) processes and phenomena (e.g. transporting, reservoir charging and discharging, oscillation). The concept of function is understood as a bridge between behavioral and teleological knowledge. It is used to interpret the

behavior of a system in term of flow structures and describe how these flow structures contribute to the achievement of the purpose of the device;

4. Teleological knowledge i.e., knowledge about the purpose (goals) of a system, the expected behavior of the system and the operational conditions that allow the achievement of the goals through correct operation;

- generality: level in a typological hierarchy of the entities and relations of a conceptual scheme. For example, the type of entity "component" in a structural scheme is more general than "electrical component" which, in turn, is more general than "resistor";
- phenomenic coverage: the range of phenomena taken into account by the entities and relations of a scheme and the kind of simplifying assumptions that they presuppose. For example, a conceptual scheme for representing the behavior of electrical circuits may include the entities E={voltage (V), electromotive force (E), current (I), resistance (R), constant (c)}, the relations R={V=RI, E=c, R=c, $\sum I=0$, $\sum V+\sum E=0$}, and the attributes A={value, unit} with domains D(value)={reals}U{unknown} and D(unit)={volt, ohm, ampere,..}. The above scheme can be used to describe the phenomenon of electrical conduction in electrical circuits. However, it is unable to describe the effect of magnetic induction of the current flowing through a wire. Moreover, a physical law, such as "E=c" in the above scheme, may be represented at a different level of accuracy by explicitly representing the dependence of electromotive force from the charge level CL in a battery (e.g. E=f(CL));
- resolution: number of distinctions allowed by the domains associated to the attributes of a conceptual scheme. For example, the resolution of the behavioral scheme described above can be lowered by relaxing real valued variables and using qualitative domains of values such as D(value)={negative (-), zero (0), positive (+)}U{unknown};
- detail: degree of granularity of the knowledge represented by the entities and relations of a scheme. For example, a structural scheme for representing electronic devices may include entities at the level of major subsystems (e.g. filter, amplifier) or can be further refined at the level of elementary components (e.g. resistor, transistor, diode).

We assume that the space of possible conceptual schemes is strictly layered i.e. any individual scheme is allowed to encompass only one specific choice about ontology, epistemological type, generality, coverage, resolution, and detail. For example, we never mix entities belonging to different ontologies or epistemological types in the same scheme. Moreover, conceptual schemes are (partially) ordered along dimensions according to their perspectives.

• *Plex*. A plex , PL, is a labelled structure consisting of nodes having an arbitrary number, n, of distinct attaching points used to join nodes together. A node of this

kind is called an n-attaching point entity (NAPE). Attaching points of napes are not connected directly together, but are connected via intermediate points known as tie-points (TP). A single tie-point may be responsible for connecting together two or more attaching points belonging to different napes.

• *System description.* A system description, SD, is a coherent collection of facts that describe a system from a particular perspective. It is represented by a plex whose structural elements (i.e. nodes, attaching points and tie-points) are instances of the entity and relation types of a conceptual scheme. The labels of the plex represent attribute/value pairs.

• *Design Fragment*. A design fragment, DF, is a pair <SD, CTX> where SD is a system description and CTX is a context specifying the conceptual scheme (and thus the perspective) used to build the system description as well as a set of operating and codesignation assumptions that explicitly state conditions under which the system description may be considered as a valid representation of the artifact to be designed.

Design fragments are the basic components of design solutions. Section 4 will discuss the main operators that can be used to construct, modify or compose design fragments together in order to generate design solutions. Another way of generating designs is through selection from among a set of generic descriptions of classes of devices called design prototypes.

• *Design prototype*. A design prototype of a category of devices X, thereafter DP(X), is a collection of interrelated design fragments representing X under different perspectives or operating modes. As an example, a design prototype DP(X), can be constituted by several fragments each one devoted to describe a specific class of epistemological features (i.e. structural, behavioral, functional, and teleological) about the category of devices under consideration. The fragments (more specifically their system descriptions) are related together by *links* which make explicit the dependencies existing among entities and relations of different representations. For example, the link between a structural description of X and a behavioral one is realised by associating each terminal, component, and connection in the structural description with the physical quantities and equations that describe the system behavior in the behavioral description. The link between behavior and function is established by associating equations governing the behavior of components in the behavioral description with appropriate roles (e.g. generator, conduit, barrier, reservoir) in the functional (role) description. Links between functional (roles) representations and functional (processes and phenomena) representations associate i) cofunctions - namely, functional role networks capable of supporting primitive processes such as transporting, reservoir charging and discharging - to

the processes they describe, and ii) organizations (i.e. process networks) to the phenomena they support. Finally, the link between function and teleology is realised by associating goals in the teleological description with the phenomena (or primitive processes) represented in the functional representation which are used to achieve them. The relation between components and goals is many-to-many since a component may participate to the realisation of several goals and, conversely, a goal can be fulfilled by utilising the behaviors of several components of a device.

The proposed approach for representing design solutions has several noteworthy properties. First, the concept of design fragment supports the explicit representation of a model contextual information such as the ontological, representational and operating assumptions lying behind its construction. As a consequence models of artifacts developed using this approach may serve as a record of commitments, helping the designer i) to focus only on relevant information at each stage of the design process, ii) to maintain system coherence among representations or iii) to represent mutually incoherent description of the same artifact (e.g. the description of an artifact behavior under different operating modes). Second, the approach supports the representation of several types of knowledge about an artifact such as structural, behavioral, functional and teleological knowledge. Because design specification is usually expressed in a language remote from solution description, this feature is used to support the conceptual mapping as from goals, into function, then into the artifact behavior, and, finally, into its structure that design effects. Moreover, the use of separate descriptions each one devoted to an epistemological type allows for a better focusing of the design process since it is possible to take into account only those pieces of knowledge that are relevant in a given stage of the design process (e.g., teleological and functional descriptions in the early phase of design, behavioural descriptions in the phase of analysis and evaluation of design solutions). As a consequence, efficiency and cognitive plausibility are improved. Third, the possibility to representing the designed system at different levels of detail, coverage and resolution supports incrementalism i.e., a step-wise production of descriptions of lower and lower aggregation level and of greater accuracy. This enables the designer to exploit a step-by-step methodology, thus reducing the computational complexity of the overall design process. Fourth, by explicitly representing links between design fragments within a prototype, the approach support the co-operative use of system descriptions for explanation and evaluation. Following the links bottom-up (i.e. from structural fragments to teleological fragments through behavioral and functional ones) it is possible to provide a teleological explanation of the behavior of a component or to explain the reasons behind a particular arrangement of components. This is valuable when we want to evaluate the effect of a modification of a design prototype such

as the elimination of a component or the substitution of a component with another one. Analogously, it is possible to follow the links top-down (i.e. from teleological fragments to structural ones) to assign blame to components i.e. to identify which components are responsible for the achievement of a given set of goals. This task is accomplished, for example, when it is needed to establish which part of a structural description of a prototype can be reused to fulfil a given set of new goals.

A characteristic feature of our representation system that makes it different from other approaches that use multiple etherogeneous descriptions such as FBS [20] and the FR scheme [8] is that substantive reasoning can occur at any epistemological level i.e. the higher levels (functional, teleological) are not merely for control and explanation but can be used by themselves to perform complex tasks (e.g. design synthesis). We call this property: multilevel operationality.

3 Modeling the design process

Our analysis of the design process is constructed around four conceptual elements: design specification, operational model, background knowledge, and design operator. These elements are briefly described below.

Design specification, SPE, refers to design goals, constraints and objectives, that is SPE=<G, C, B>. A goal represents the ultimate purpose that the design is intended to achieve when put in use. In our approach a goal consists of: i) a goal pattern, that is, a goal name and a set of arguments specifying the variables relevant to the definition of the goal, ii) a set of operational conditions for the achievement of the goal and iii) the intended behavior, that is, the relationships expected to hold among the values of output and operational conditions when the goal is achieved. Constraints express relations among some properties of the proposed artifact (or of the design process) and its environment. Objectives are stated as to maximize or minimize a device (or process) attribute or a function of them.

The *operational model*, M, of an artifact is a, possible, incomplete, representation of the artifact to be designed. It is specified by one or more design fragments or prototypes. The design specification and the operational model coevolve during the design process: goals and constraints are gradually turned into descriptive statements that are included into the operational model; at the same time design analysis and evaluation supports the discovery (or the relaxation) of new (old) constraints and objectives which modify design specification. As we will be using specifications and operational models at different design steps we will use the index "n" to indicate the step under consideration. For example, SPE(n) is the specification at step n. We then define the *design state* at step n, DS(n), to be the pair <SPE(n), M(n)>.

Background knowledge, BK, represents the prior designer's knowledge. This knowledge includes two main components: physical knowledge and design knowledge. Physical knowledge represents the understanding that engineers have of physical phenomena. It consists, for example, of general principles and laws of physics, techniques of applied mathematics, ontologies of engineering concepts, conceptual schemes. Design knowledge refers to empirical knowledge that describes how design can be performed. This knowledge may include, for example, libraries of design prototypes, knowledge about the possible decompositions of design goals into subgoals, knowledge about how goals can be attained by the use of functional compositions together with the behaviors and structures supporting these functions, general principles of good design that may be used to guide the design process.

Finally, *design operators*, OP, represent actions that can be performed during the design process. We distinguish between two categories of operators:

1. modeling operators: i.e. actions that change the design state. These are further classified into the following two subclasses: actions that affect the design specification and thus can be used for problem structuring and reformulation; actions that affect the operational model and thus serve for design development;
2. decision operators: i.e. actions that do not change the design state but have an effect in the internal state of the designer (human or artificial) viewed as a decision maker. This category includes, for example, actions for gathering information about the current state of design, actions supporting the analysis, interpretation and evaluation of design solutions as well as actions for selecting from among a set of alternatives.

Each action has a name, a precondition and a postcondition or effect. If the precondition is satisfied in the current design state than the postcondition is asserted in the new state.

We propose to view the design task as a bilevel reflective process constituted at the domain level (or base level) by a sequence of modeling steps which are guided at the meta level (or control level) by a decision activity which continuously monitors and redirects the activity at the domain level. A modeling step takes as input: the current design state DS(n), a modeling operator OP(n) and background knowledge BK. It produces as output a new design state DS(n+1) obtained by applying the specified operator to the current design state. At the control level, the decision activity can be decomposed into a sequence of decision steps. Each decision step takes as input the current design state DS(n) and background knowledge BK. It produces as output a modeling operator OP(n) and (possibly) a modified design state DS*(n) (e.g. additional constraints and objectives). The decision activity: i) analyses, interprets and evaluates the actual design state, ii) identifies what it needs in order to improve the overall design activity, iii) formulates internal objectives that are included in design

specification, and iv) select the modeling action that is most appropriate for pursuing these objectives. Hence, in our conceptualisation, the design process results in a ordered sequence of design states <[SPE(0), M(0)], ..., [SPE(k), M(k)]>, with SPE(0), M(0) representing the initial specification and operational model, respectively, and M(k) representing the final design solution i.e. a complete and sufficiently detailed operational model of the desired artifact that satisfies specification SPE(k).

4 Model transmutations

Model transmutations describe basic inferences that can be performed for constructing, modifying or combining design fragments together. In this section we discuss some important types of transmutations, but the list is hardly complete. Model transmutations have been classified into three classes depending on whether they:

1) transform existing design fragments by changing their perspective (Transformations);
2) combine together two or more existing design fragments (Combinations);
3) retrieve from background knowledge new design fragments (e.g. a prototype) that are added to those already existing in the current design state (Prototype copying);

Seventeen specific transmutations have been identified as specializations of the above abstract operators. Although the discussion focuses on operators for design development, most of the following transmutations - such as generalisation, specialization, conceptual abstraction and concretion - can be performed also for problem structuring and reformulation.

4.1 Transformations

Transformations are unary operations that take as input a design fragment DF (the source fragment) and produce as output a new design fragment DF* by modifying the perspective of the conceptual scheme associated to DF. Transformations can be classified according to two main criteria: 1) the dimension used to change perspective; 2) the direction (up or down) followed by the operation along that dimension. A brief description of basic transformations follows.

• *Aggregation* (versus *refinement*): the conceptual scheme of the source fragment is replaced by a less (more) detailed one. As a consequence a set of related napes A ($Z \supseteq A$) of the plex Z associated to the source design fragment DF is contracted to a single nape in the transformed design fragment DF* which is given the adjacency relationships existing between A and Z-A. According to

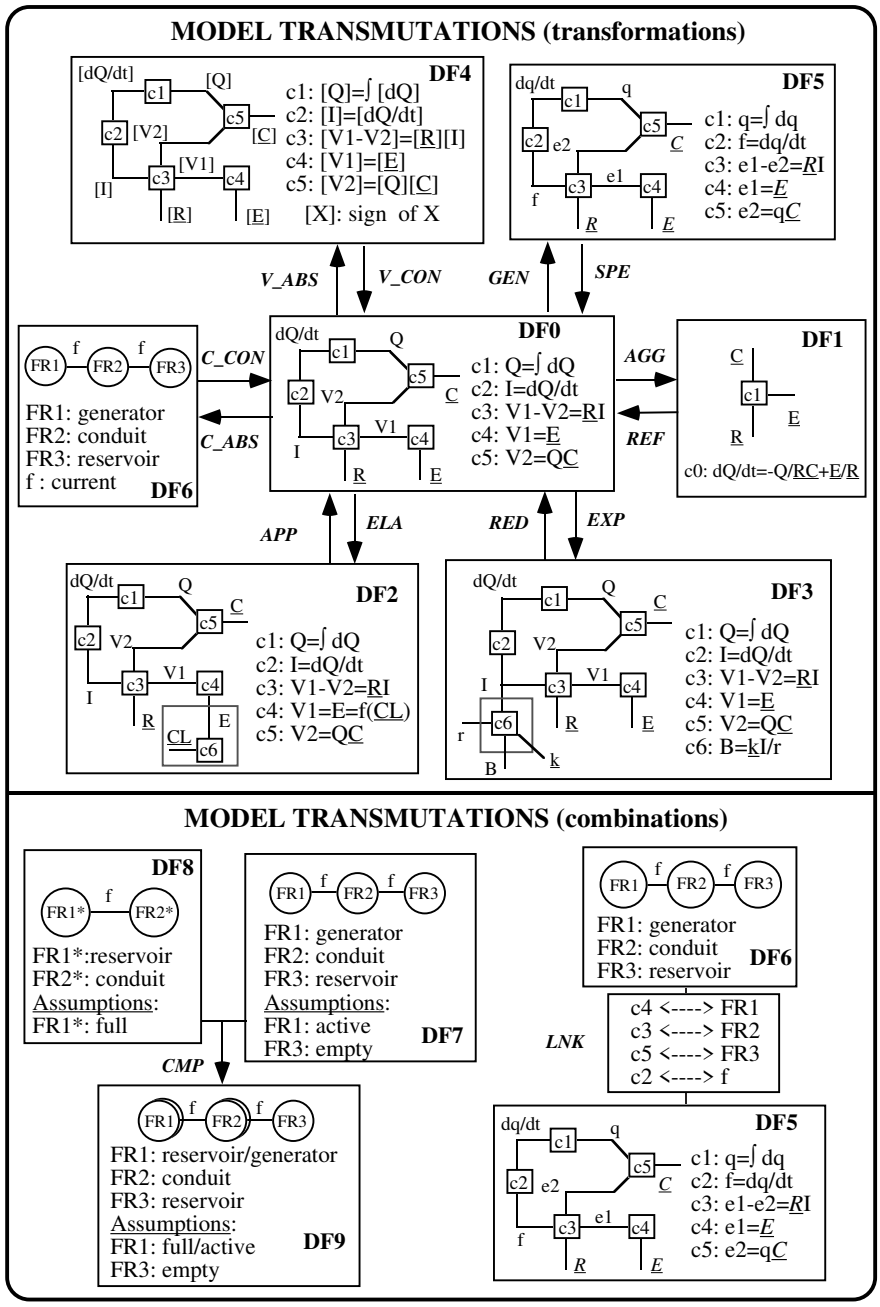

Figure 1. Examples of model transmutations

the epistemological type of the design fragment that is involved in the operation it is possible to further distinguish among structural, behavioral, functional, and teleological aggregations (refinements). Figure 1 (fragments DF0, DF1) shows an example of behavioral aggregation (refinement). A set of physical equations (c1-c5) representing the behavior of a simple RC circuit in the source fragment DF0 is contracted into a single equation (c0) in the transformed fragment DF1.

• *Approximation* (versus *elaboration*): the conceptual scheme of the source fragment is replaced by a new scheme representing the same range of phenomena of the original one but using less (more) accurate relationships. As a consequence a set of napes A of the plex Z associated to the source design fragment DF is replaced by a new set of napes A* which are given the adjacency relationships existing between A and Z-A. The concept of "less-more" accurate is defined according to some rank hierarchy existing among the entities and relations of a conceptual space. For example, equations of physics that represent the same aspect of reality based on different and contradictory assumptions can be organized into rank hierarchies according to a "simpler_than" relationship. The rank hierarchy RH1 shown in Figure 2 organises alternative ways to model the voltage across a battery. In Figure 1 (fragments DF0, DF2) it is shown a behavioral approximation based on this type of rank hierarchy. An approximation is performed by replacing the physical equation V1=E=f(CL) in DF2 with the "simpler" equation V1=E in DF0. (We underline a variable to mean that it is constant).

• *Reduction* (versus *expansion*): the conceptual scheme of the source fragment is replaced by a new scheme representing a narrower (wider) range of phenomena than the original one. As a consequence a set of napes are eliminated (added) from the plex associated to the source design fragment. As an example, design fragment DF0 in Figure 1 is obtained by removing from the source fragment DF3 the equation B=kI/r representing the phenomenon of magnetic induction of electrical current. Note that the reduction operator (and its inverse) can be used to control which aspects of reality are considered immaterial for the design problem at hand and thus must be eliminated from the representation and which aspects, on the other hand, are considered relevant and thus must be included. Instead, the approximation operator (and its inverse) can be used to control how relevant phenomena must be represented i.e. what kind of simplifying assumptions are appropriate.

• *Generalization* (versus *specialization*):the conceptual scheme of the source fragment is replaced by a new scheme obtained by moving up (down) a "type-of" hierarchy of entities and relations. We constrain the two conceptual schemes to share the same ontology and epistemological type. As an example, Figure 1

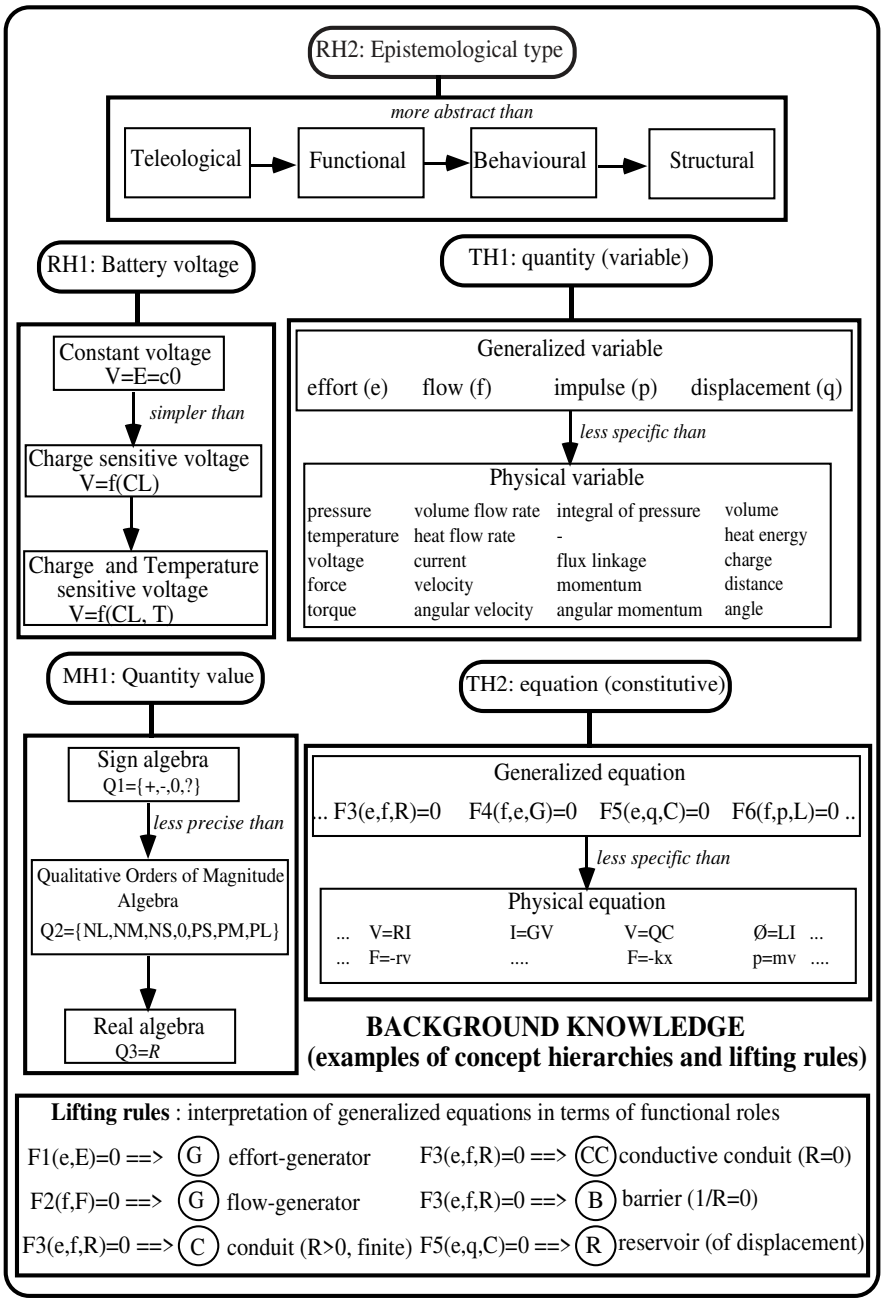

Figure 2 Background knowledge (partial view)

(fragments DF0, DF5) illustrates an example of generalization (specialization) involving a behavioral design fragment. Generalization is performed by substituting physical quantities and equations in the behavioral design fragment DF0 with corresponding generalized entities. The transformation is based on the use of the typological hierarchies TH1, and TH2 shown in Figure 2. According to hierarchy TH1, physical quantities Q (electrical charge), I (current), V (voltage), R (resistance), E (electromotive force) and C (capacity) are replaced by corresponding generalized quantities namely, q (displacement), f (flow), e (effort), *R* (generalized resistance), *E* (generalized electromotive force) and *C* (generalized capacity). Similarly, physical equations are replaced by their generalized counterparts e.g. the physical equation V1-V2=RI (i.e. an instance of the Ohm's law) is generalized into the equation e1-e2=*R*f.

• *Conceptual abstraction (versus conceptual concretion)*: the conceptual scheme of the source fragment is replaced by a new scheme obtained by moving up (down) a rank hierarchy of ontologies or epistemological types. As an example, the rank hierarchy RH2 in Figure 2 organises epistemological types within a macroscopic ontology into six levels of abstraction according to the degree of context independence and function neutrality the entities and relations of an epistemology allow to attain. At the lowest levels of abstraction (e.g. at the structural or behavioral levels) entities and relations represent the objective language of physics and engineering sciences which is relatively neutral with respect to the intended functionality of an artifact. At the higher levels (e.g. at the functional or teleological levels) entities and relations derive from a subjective interpretation of behavior and structure in a context. Conceptual translations between levels of the hierarchy require a set of *lifting rules* specifying how the entities and relations of an epistemological type are appropriately reinterpreted in the light of a new epistemology. Figure 1 shows an example of conceptual abstraction (concretion). The behavioral design fragment DF0 is abstracted into the fragment DF6 by reformulating physical quantities and equations in the source fragment in terms of a functional role network i.e. a series of three functional roles namely a generator, a conduit and a reservoir connected by mutual dependency relations. The transformation is based on the lifting rules shown in Figure 2 (bottom).

• *Value abstraction* (versus *concretion*): the conceptual scheme of the source fragment is replaced by a new scheme obtained by replacing the domains of the attributes of the original scheme with new domains obtained by moving up (down) a measure hierarchy [12]. As an example, Figure 1 (fragments DF0, DF4) shows a value abstraction (concretion) applied to the attribute "value" of physical quantities in a behavioral design fragment. Value abstraction occurs by replacing the numerical (i.e. real) values of each quantity X in the source design

fragment DF0, with the "sign" of X (denoted by [X]) taking possible values in the set {negative (-), zero (0), positive (+)}. The transformation is based on the rank hierarchy MH1 shown in Figure 2.

4.2 Combinations

Combinations are binary operations that take as input two design fragments DFi, DFj and produce as output a new design fragment DF* by combining the elements of DFi and DFj according to some specified rule. Two types of combinations namely, compositions and links, have been identified.

• *Composition* (versus *decomposition*): two design fragments sharing a conceptual scheme are composed by forcing two previously separated elements of their system descriptions to be the same, thereby, requiring to add a codesignation assumption in the output design fragment. The rule associated to the transmutation specifies compatibility conditions that the elements selected to be unified must satisfy. The operational assumptions associated to the output design fragment are the union of the respective assumptions in the source design fragments. As an example, Figure 1 (bottom left) shows the composition of two functional (roles) fragments (DF7 and DF8). The fragments are composed by unifying compatible elements. Two roles are compatible if they are equivalent roles and i) they are identically instantiated, or ii) they are not yet instantiated or iii) one role is instantiated and the other is not. In the above example FR2* and FR2 are unified since they represent identical roles (i.e. two conduits of the same flow f). Analogously, FR1* (a reservoir) and FR1 (a generator) are unified according to a rule that establishes the equivalence of a not empty reservoir and an active generator. Notice that FR1* and FR3 (two reservoirs) are not compatible since they make contradictory operating assumptions (i.e. FR1* is assumed to be full while FR3 is empty).

• *Link* (versus *unlink*): the elements of the system description of a design fragment DFi are put in correspondence with the elements of the system description of another design fragment DFj. The set of entities described or referred to by the two fragments (i.e. their scope) cannot be disjoined. Fragments DFi and DFj may differ in perspective (e.g. they may have different epistemological types). Figure 1 (bottom right) shows an example of link between a behavioral (DF5) and a functional (DF6) design fragment. The operation is realised by associating equations governing the behavior of components in the behavioral design fragment with appropriate functional roles in the functional fragment. This operation can be used to build design prototypes.

4.3 Prototype copying

Prototype copying is a transmutation that generates new design fragments from scratch by recalling one or more design prototypes from background knowledge. The operator takes as input a library of design prototypes, a set of features used as indices, and a similarity metric. The operator matches indices to prototype attributes and retrieve those prototypes that are most similar to the input features. Since prototypes are recorded as a collection of design fragments having different perspectives, indices may be expressed at different abstraction levels and generality. Once a design prototype has been recalled from memory it is possible to use the above transformations and combinations for adapting it to the current design problem.

5 Multitype inference in conceptual design

Basic transmutations can be concatenated in various ways to produce complex inference patterns. Some recurring combinations of operators have a particular significance. For example, the operation of *similarization* (also known as cross contextual analogical reasoning) [14] substitutes the conceptual scheme of a design fragment with a sibling scheme in a typological hierarchy and reinterprets the system description of the source fragment in the new conceptualisation. This operation can be viewed as the concatenation of a generalization transformation followed by a (re)specialization in a different physical domain. As an instance, suppose that the problem is to design an aircraft rate-of climb sensor (denoted by Xb) whose purpose (G*) is to measure the rate of pressure change of a given pressure source. Suppose further that no prior prototype exists in design background knowledge for this class of devices. The problem may be addressed by exploiting cross contextual analogical reasoning as follows. First, the desired goal "G*: TO_SENSE_RATE_of p: pressure CHANGE" is generalised into "G: TO_SENSE_RATE_of e: effort CHANGE" by recognising that pressure is an instance of the generalised effort variable "e" in the hydraulic domain (see the typological hierarchy TH1 in Figure 2). This is an example of problem reformulation. Second, the library of design prototypes is searched to find a prototype whose teleological representation describes a goal that is a specialization of G in some physical domain. Suppose that such a specialisation exists in the electrical domain so the search succeeds giving a prototype (say DP1) for the class of electrical systems Xa (the source solution) having the goal "G': TO_SENSE_RATE_of v: voltage CHANGE". Goal G' is considered "similar" to G* since both share a common generalization G. Given the exemplar DP1, additional knowledge is derived about the source solution by following links top down from teleology to function and selecting its functional representation. The functional fragment

specifies that the goal "G': TO_SENSE_RATE_of v: voltage CHANGE" is realised in this device by a physical process of the type "Reservoir charging" in the electrical domain whose cofunction (functional role network) is constituted by three functional roles - namely, a voltage generator, a conduit of current and a reservoir of charge - related by dependency links. The desired output (i.e., dv/dt) is measured across the conduit which is equivalent to a generator when it is in the functional state "crossed". Given the functional fragment for the class of systems Xa, this model is mapped over to the target domain (i.e., in the hydraulic domain) to represent the class of systems Xb. Thus, voltage, current and charge are mapped into pressure, volume flow rate and volume (of gas) respectively. As a consequence, the source cofunction in the electrical domain is mapped to a similar cofunction, in the hydraulic domain, constituted by a generator of pressure, a conduit of flow and a reservoir of volume (of gas) related by dependency links. Finally, given mapped target cofunction, the last step consists in searching the library of prototypes to find components that are capable of playing the roles considered in the cofunction. Since, the component which plays the role of the generator of pressure is known from the specification only the other two roles - namely, the conduit of flow and the reservoir of gas - must be instantiated. The retrieval process provides a fluid resistance (e.g. an orifice) and a fluid capacitance (e.g., a fluid accumulator) as candidate components. The design solution is thus constituted by composing the generator of pressure specified in input with an orifice and a fluid accumulator. The desired output (i.e. dp/dt) must be measured across the orifice. The plausibility of this conclusion rests on the assumption that the type of goal determines (or is relevant to) the functional organisation of a device independently of physical domains. Of course, this may be incorrect but often this heuristic provides a good starting point for the conceptual design stage.

An accurate analysis of the foregoing line of reasoning reveals a complex pattern of transmutations including similarization, prototype copying, conceptual abstraction and composition. Some of them (e.g. similarization, composition) are performed explicitly as a sequence of one or more individual steps. Others (e.g. conceptual concretion) are, in fact, compiled into a prototype and thus are executed implicitly by recalling the prototype from background knowledge and navigating between its fragments. The strengths and weakness of both approaches are well known. Explicit use of model transmutations rapidly becomes intractable as the complexity and the magnitude of the problem increase. However, since this approach is based on "first" and "second" principles it is especially suited for solving novel problems (i.e. for innovative and creative design). On the other hand, prototype-based reasoning since it reasons from previous experience may improve productivity and guarantee standardised solutions when applied to solve routine problems. However, its effectiveness is strongly limited if new problems or problems requiring some

form of creative processing are addressed. In real design problems a trade-off between efficiency and effectiveness is expected: some parts of the operational model are generated implicitly by recalling from memory design prototypes and by adapting them to the present situation using transformations while other parts of the operational model are generated by constructing and composing fragments using model transmutations explicitly.

5.1 Relationship among input information, prior design knowledge and model transmutations

At any step of the design process several transmutations may be applicable. Each transmutation may also be applicable to different design fragments of the current design state. A critical issue is, therefore, to select the focus of attention and the next action to perform. Two critical factors affect the choice of model transmutations: the "complexity" of the initial design state and the relation existing between design states and background knowledge. The "complexity" of the initial design state can be characterized along two dimensions: amount of unspecified structure and gap in abstraction level between input specification and desired solution. The "complexity" of the starting design state DS(0) affects the choice of the initial transmutation. This step, also known as incipient model creation, is very important since a model that is created first has a large influence on the result. The second factor concerns the relation existing between design states and background knowledge. At each step of the design process the current design state activates portions of the background knowledge which are relevant to the current specification or operational model. The selection of what action to do next depends on the relationships between relevant background knowledge and the information that is provided in the current design state. Several relationships can be envisaged [10]:

1) The current state is already known to the designer. This case occurs, for example, when design goals match exactly some part of BK (e.g. the teleological fragment of a prototype). In such a situation, the stored prototype is recalled and included into the current operational model.
2) The current state is implied (or implies) part of BK. This case occurs, for example, when design goals appear as subgoals in the teleological descriptions associated to one or more prototypes. By exploiting the links existing between design fragments of the prototypes it is possible to isolate the parts of the prototypes that are responsible for the achievement of the specified goals, copy these fragments within the current operational model and then try to compose these fragments together to obtain a possible model of the desired artifact.
3) The current state evokes an analogy to a part of BK. This case represents the situation when the design goal does not match any background knowledge

nor it is implied by it. Similarization can thus be used to generate a similar goal that is used to retrieve the operational model of an analogous system. This model is in turn transformed, again by similarization, in the desired physical domain as illustrated in the previous section.

4) The current state represents completely new information. A solution is not known or cannot be discovered by search but it must be generated from scratch using basic transmutations (explorative design). The designer may try to generalize (or specialise) the input specification, relax some goals or constraints, negate desired goals, etc. so that the new specification can account for information stored previously and the control can be passed to one of the above cases.

6 The SECS system

In order to experiment with the above ideas we developed SECS (an acronym for Small Electronic Circuit Synthesizer) a reasoning system aimed at supporting the electronic engineer in the phase of conceptual design of small electronic circuits. SECS has been implemented on Apple Macintosh machines using LPA Prolog. The system allows the designer to use either past experiences in the domain of application or basic transmutations to adapt or generate design solutions. Past experiences are recorded by means of a library of design prototypes which are organized into generalization (type-of) and meronimic (part-of) hierarchies. This type of organization is more or less standard but we identified specific levels of generalisation - namely, teleological class, design concept, variant, plan, scheme and case - according to the extension of the reference set described by a prototype. Prototypes are indexed by the goals they achieve and the operational conditions under which it is appropriate to use them. The operation of SECS is based on the classical Retrieve/Select/Adapt method [18]. In the current implementation of the system there is not an explicit decision phase. Switching from a strategy to another (e.g. from prototype copying to the explicit use of transformations or compositions) is failure-driven. It occurs when: i) there are no prototypes satisfying one or more of the specified goals, ii) there are two or more competing prototypes satisfying only a subset of specified goals and these subsets have some element in common. In case i) the system switches to cross contextual analogical reasoning using similarization. In case ii) (corresponding to case 2 in the previous section) the system switches to a compositional based strategy using partial prototypes represented at the functional level as elementary fragments. Composition-based design (at the functional or behavioral levels) is performed also when initial design specification is formulated in functional or behavioral terms. More detail about the SECS system can be found in [1]. Current research efforts are aimed at implementing the complete decision step (i.e. analysis and evaluation, critique,

and selection) by adapting the preference- based method proposed in [19] for task-adaptive model selection to the choice of transmutations.

7 Conclusions and related work

In this paper we propose to view design as a bilevel reflective process. This process is constituted, at the base level, by a sequence of modeling steps which are dictated, at the metalevel, by a decision activity which continuously monitors the progress of the design process at the base level and determines at run time what to do next and when to stop. The modeling activity has been analysed in terms of patterns of inference called model transmutations. Three categories of transmutations have been discussed with reference to the Multimodeling approach for representing physical systems. These are: transformations, combinations and prototype retrievals. The analysis of model transmutations builds on earlier research in compositional model-based design [3], innovative design [15], [21], prototype-based reasoning [7], [17] and cross contextual analogical reasoning [14]. The work presented in the paper also integrates several results obtained in the area of qualitative physics and automated modeling [6], [9], [13], and multistrategy task adaptive learning [10], [11]. The major contributions of our research derive from the use of the Multimodeling approach for representing both design experience (i.e. prototypes) and design solutions (i.e. design fragments and operational models). The approach provides a set of conceptual schemes for building different models of the same artifact and a systematic method for relating all these models together by links to support multilevel reasoning.

References

[1] Beltrame, A. & Toppano, E., Prototype-based conceptual design: the SECS system, *Applications of AI in Engineering X*, eds. R.A. Adey, G.Rzevski, & C.Tasso, Comp. Mechanics Pub., Boston, pp. 502-512, 1995.

[2] Borst, P., Akkermans, H., Pos, A., & Top J., The PhysSys Ontology for Physical Systems, *Proc. QR Workshop*, Amsterman, the Netherlands, pp. 11-21, 1995.

[3] Bose, P. & Rajamoney, S.A., Compositional model-based design. *Proc. IJCAI-93*, Chambery, France, pp. 1445-1450, 1993.

[4] Chandrasekaran, B., Design problem solving: a task analysis, *AI Magazine*, Winter 1990, pp. 59-71, 1990.

[5] Chittaro, L., Guida, G., Tasso, C. & Toppano, E., Functional and teleological knowledge in the Multimodeling approach for reasoning about physical systems: a case study in diagnosis, *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 23, No.6, pp. 1718-1751, 1993.

[6] Falkenhainer, B., & Forbus, K.D., Compositional Modeling: Finding the Right Model for the Job, *Artificial Intelligence*, 51, pp. 95-143, 1991.

[7] Gero, J.S., Design prototypes: a knowledge representation scheme for design, *AI Magazine* 11 (4), Winter 1990, pp. 26-36, 1990.

[8] Goel, A., & Chandrasekaran, B., Functional Representation of Designs and Redesign Problem Solving, *Proc. IJCAI-89*, Detroit, MI, USA, pp. 1388-1394, 1989.

[9] Iwasaki, Y., & Levy, A.Y., Automated model selection for simulation, *Proc. AAAI-94*, Seattle, WA, USA, pp. 1183-1190, 1994.

[10] Michalski, R.S., Inferential learning theory as a basis for multitrategy task-adaptive learning, *Proc. First International Workshop on Multistrategy Learning*, November 1991, Harpers Ferry, West Virginia, pp. 3-18, 1991.

[11] Michalski, R.S., & Hieb, M.R., Knowledge representation for Multistrategy Task-Adaptive Learning: Dynamic Interlaced Hierarchies, *Proc. Second International Workshop on Multistrategy Learning*, Harpers Ferry, West Virginia, pp. 3-17, 1993.

[12] Murthy, S., Qualitative reasoning at multiple resolutions, *Proc. AAAI-88*, Saint Paul, MN, USA, pp. 296-300, 1988.

[13] Nayak, P.P., Causal approximations, *Artificial Intelligence*, vol. 70, pp. 277-334, 1994.

[14] Navinchandra, D., Sriram, D., & Kedar Cabelli, S.T., Analogy-based engineering problem solving: an overview, *AI in Engineering: Tools and Techniques*, eds. Sriram & Adey, Computational Mechanics Pub., Boston, pp. 273-285, 1987.

[15] Neville, D., & Weld, D.S., Innovative design as systematic search, *Proc. AAAI-93*, Washington, D.C., USA, pp. 737-742, 1993.

[16] Pahl, G., & Beitz, W., *Engineering Design*, The Design Council, London, 1984.

[17] Rajamoney, S.A. & Lee, H., Prototype-Based reasoning, *Proc. AAAI-91*, Anaheim, CA, USA, pp. 34-39, 1991.

[18] Riesbeck, C.K. & Schank, R.C., *Inside Case-Based Reasoning*, Lawrence Erlbaum Associates Pulishers, Hillsdale, New Jersey, 1989.

[19] Toppano, E., Rational Model Selection in Large Engineering Knowledge Bases, *Applied Artificial Intelligence Journal*, vol. 10, No. 3, pp. 191-224, 1996.

[20] Umeda, Y., Takeda, H., Tomiyama T., & Yoshikawa H., Function, Behaviour, and Structure, *Applications of Artificial Intelligence in Engineering V*, vol.1, eds. J.S.Gero, Springer-Verlag, pp. 177-193, 1990.

[21] Williams, B.C., Interaction-based invention: designing novel devices from first principles, *Expert Systems in Engineering: Principles and Applications. Proc. Int. Workshop*, Vienna, Austria, pp. 119-134, 1990.