Infused Design: I. Theory

Article *in* Research in Engineering Design · September 2004

DOI: 10.1007/s00163-004-0047-7

2 authors:

Offer Shai
Tel Aviv University
47 PUBLICATIONS **596** CITATIONS
SEE PROFILE

Yoram Reich
Tel Aviv University
193 PUBLICATIONS **2,858** CITATIONS
SEE PROFILE

Research in Engineering Design (2004) 15: 93–107
DOI 10.1007/s00163-004-0047-7

ORIGINAL PAPER

Offer Shai · Yoram Reich

Infused design. I. Theory

Received: 20 August 2002 / Revised: 15 July 2003 / Accepted: 24 March 2004 / Published online: 15 July 2004

Abstract The paper introduces infused design: an approach for establishing effective collaboration between designers from different engineering fields. In infused design, the design problem representation is brought up to a mathematical meta-level, which is common to all engineering disciplines. The reasoning about the problem is then done by using mathematical terminology and tools that, due to their generality, are the same for all engineers, disregarding their background. This gives engineers an opportunity to infuse their work with knowledge, methods, and solutions shared by specialists from other engineering fields. When these knowledge, methods, and solutions cross disciplinary boundaries, they are provably relevant to any problem in another domain to which it can be transformed. The suggested meta-level consists of general discrete mathematical models, called combinatorial representations (CR). Specific mathematical basis for the combinatorial representations chosen in this paper is graph theory although other representations are possible. We explain the theory of infused design and carefully contrast it with other approaches. This comparison clearly demonstrates the advantages of infused design and its potential. We conclude with several practical issues related to the introduction of infused design into practice and briefly discuss the role of information systems in infused design. A companion paper includes several examples that demonstrate the details of infused design.

Keywords Collaboration · Concurrent engineering · Creativity · Combinatorial representations · Graph theory · Knowledge sharing · Knowledge infusion · Product quality · Design process

O. Shai (✉) · Y. Reich
School of Mechanical Engineering,
Faculty of Engineering, Tel Aviv University,
69978 Tel Aviv, Israel
E-mail: shai@eng.tau.ac.il

1 Introduction

Design is a social, collaborative process. Customers, designers, and other professionals exchange information about the product objectives and their roles in its development. They divide the design tasks, monitor the process, detect and resolve conflicts, and continually negotiate the meaning of information related to, as well as generated in, the project (Finger et al.1995; Konda et al.1992; Schrage 1991).

In most design projects, the initial process stages involve arriving at mutual understanding of terminology and the scope of the work. When parts of the work become clear and agreed upon, work progresses by individual problem solving. As design progresses, the nature of collaboration is directed to minimize and resolve conflicts.

Since collaborative design is inevitable, the competitiveness of organizations depends on their ability to exercise quality collaborative design efficiently and effectively. Concurrent engineering (CE) (Prasad 1996; Reich and Subrahmanian 1992; Walton 1991) is seen as a collection of techniques and development philosophy that could improve design once put in place to support collaboration. Concurrent design happens at the interface between designers and their design support systems (Finger et al. 1995). Concurrent design helps to consolidate the different perspectives that are active in a design project. It also engenders the creation of new knowledge when different perceptions meet, thus extending the boundaries of each with additional concepts, relations, and operations.

To date, this interface has been the focal point of many studies attempting to support the concurrent work of engineers on large projects. In practice, a well-defined, agreed upon interface could become the backbone of successful projects whereas its omission would be an impediment. Work on the interface can be subdivided into technical and organizational perspectives (Reich and Subrahmanian 1992). In both perspectives, work has dealt with issues at various levels. The technical perspective, which has been the main focus of academic research, has dealt with topics such as:

1. Defining languages for specifying the design parameters (McMahon and Xianyi 1996; Clarkson and Hamilton 2000)
2. Creating product models (Eastman and Jeng 1999), including systems for configuration and product data management (PDM) and electronic mock-ups (Bechkoum 1997)
3. Defining the interface in terms of shareable parameters (Clarkson and Hamilton 2000), constraints (Bowen and Bahler 1992), choices, and other knowledge elements such as ontologies (Neches et al. 1991). These definitions could sometimes be augmented with mechanisms for automated detection of conflicts (Bowen and Bahler 1992) or with agents' technology for automated execution of well-defined engineering tasks (Park et al. 1994).
4. Developing synchronous communication channels (Subrahmanian et al. 1993b; Minneman and Leifer 1993)

The organizational perspective, which has been the essence of most practical implementations of concurrent engineering, has dealt with issues such as:

1. Creating the context (e.g., structure and procedures) for the working of multifunctional teams
2. Developing mechanisms that operate on the interface for coordinating work (Whitfield et al. 2000) including ensuring effective communication between professionals and between them and customers (Akao 1990; Reich 2000; Reich et al. 1996)
3. Structuring design projects such that the interface becomes small and manageable (small interface means minimal interactions and minimal chances for conflicts) (Eppinger et al. 1994; Steward 1981)

Studies that touched both perspectives include tools for asynchronous communication channels (Subrahmanian et al. 1993b), design rationale capture methods (Reich 2000), or more general infrastructures for information modeling (Reich et al. 1999).

The underlying assumption in all these studies was that engineers collaborate through an interface and otherwise conduct their work individually using the domain methods and tools developed within their own discipline. In many situations, an interface exists even within one discipline where professionals use different terminologies to discuss their work (Sargent et al. 1992). In CE, the *interface is a hurdle* to be minimized.

Our work revises this situation. We see the interface issue in CE as an *opportunity*. We see the opportunity of creating new design knowledge at the interface between designers and thus work to extend the interface in order to fuse design knowledge from diverse disciplines into coherent usable knowledge that can solve disciplinary as well as interdisciplinary problems. We not only support "transcending boundaries while retaining the technical expertise afforded by specialization" (Finger et al. 1995, p. 89); but also improve technical expertise by fusing interdisciplinary expertise into one amalgam. Consequently, we address the call that "research should focus on the creation, maintenance, and extension of links between the various participants" (Finger et al. 1995, p. 90).

Our work presents a new way of doing design called "infused design." Infused design supports the exchange of focused, high quality, critical knowledge between project participants. In this practice, engineers not only collaborate through the traditional interface, but also through new interfaces on conceptual as well as detailed design activities. They exchange knowledge about problem modeling, analysis, and design procedures through combinatorial representations (Shai 2001b), thus bootstrapping engineering work in ways not previously conceived. In some cases, design is made possible, and in others, its quality is improved due to fusing knowledge and methods from other disciplines. Moreover, such benefits may also shorten the development processes.

Following an analysis of state-of-the-art work in concurrent design (Sect. 2), we present the origin of infused design in a multidisciplinary combinatorial approach (MCA) (Shai 2001b) (Sect. 3), and present the concept of infused design in detail (Sect. 4). Subsequently, we analyze infused design in relation to other approaches (Sect. 5). We carefully show how infused design is more powerful due to its mathematical foundation. We also delineate the issues involved in the introduction of infused design into design practice. In the sequel paper, we provide several examples of infused design that demonstrate its strength.

2 Collaborative and Concurrent engineering

Design has always been done collaboratively unless one designs and produces for oneself in a solitary act. Almost all products of interest require the involvement of teams of participants from diverse disciplines for their development and certainly for their manufacture. Concurrent engineering is a relatively new approach to developing products where interdependent tasks are carried out simultaneously by different engineers or teams. Concurrent engineering involves the cooperation of people from multiple disciplines in a way that incites sharing goals, methods, perspectives, needs, and other concerns related to the project. When customers or users are heavily involved in the development process, it is called participatory design (Reich et al. 1996).

Whenever collaborative or concurrent engineering takes place, different people, or people and systems, interact. The interaction serves to exchange knowledge between all parties that need it for the progression of work. There are two broad interdependent concerns related to this interaction:

1. How, when, and among whom is it performed?
2. What is exchanged?

These issues could be studied from two perspectives: studying design activities and creating support tools to address these concerns. Empirical studies of design have been conducted in many settings and have attempted to answer the above questions (e.g., Ahmed et al. in press; Dutoit 1996; Boujut and Tiger 2002; Subrahmanian 1991). From the support perspective and following the findings of the empirical studies, the first question deals with issues such as mechanisms for effective communication, selection of partners, and the organization of work to optimize the exchange (e.g., minimize interdependencies between tasks). The second question deals with the sole purpose of the communication: the knowledge being communicated, its complexity, and value to participants. The dependency between these concerns is illustrated by issues such as: increased concurrency leads to designing with increased amount of unknown information that results in future conflicts. Using coordination methods, the project structure and the interactions are designed for the purpose of minimizing conflicts or knowledge exchange.

A recent review dealt with two perspectives of coordination:

1. The *strategic* perspective dealing with coordination techniques, cooperation and collaboration techniques, organizational models, project management, workflow management, and task models (Whitfield et al. 2000)
2. The *operational* perspective dealing with resource management, planning, and scheduling (Coates et al. 2000)

Independent of the perspective or particular coordination technique, the fundamental facilitator of communication is a common language. Without a common language, communication is meaningless or very limited. By language, we mean the concepts and how they are assembled (e.g., grammar) by an agent to deliver its intent or even its state to the outside world. Inherently, no two agents, natural or artificial speak exactly the same language. When addressing the same physical object, the two agents might refer to it by different names, and the same name might mean different objects for these two agents. In product development, concepts mean different things for participants from vertically different departments (e.g., customers, marketing people, designers, manufacturers, distributors, or maintainers) and horizontally different disciplines (e.g., mechanical, electrical, software, or biomedical engineering). Similar naming problems occur even *within* a discipline (Sargent et al. 1992) and in research (e.g., Messac and Chen 2000).

Automated or computational approaches to concurrent engineering attempt to address the language problem by creating a universal language with definitions of the meaning of terms that could be understood by all interacting agents (Erkes et al. 1996; Neches et al. 1991; Park et al. 1994). Such language is often referred to as ontology.

In real domains, terminology evolves continually (Coulter et al. 1998). Consequently, in such situations, fixing language *a priori* is impossible, while creating dynamic environments conducive of language evolutions including thesaurus and indexing mechanisms is the only option. This observation is strengthened by the limited success that computational approaches to CE have had since their inception. Therefore, we focus on concurrent engineering performed by people.

Overcoming the language problem is part of design (Dutoit 1996). In the early stages of a project, or when confronted with new tasks, the development team must (and also has a tendency) to develop a vocabulary with which to interact about the project (Garrod 1998). This vocabulary gets refined as the project evolves when discrepancies between different peoples' understanding of terms emerge and are being resolved. The earlier a design team settles on a vocabulary and aligns all its members, the better are the prospects of the project (Dutoit 1996).

Given the quintessential role of common language, the crucial issue is how could procedures and tools support the process of vocabulary formation or language alignment? Such support must address the initial language formation as well as the subsequent language evolution that is driven by design conflicts or various misunderstandings (Reich el al. 1993).

We can summarize the state-of-the-art of collaboration and concurrent engineering technology and practice through Fig. 1a. Project participants such as designers or teams of designers work apart using their own tools and knowledge and interact by exchanging information about concepts, parameters, constraints, or other relevant information that constitutes their shared interface. The interaction or sharing acts can be done in design

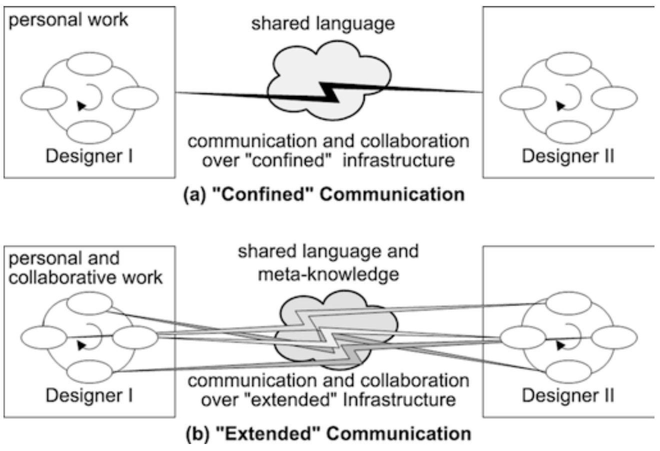

personal work
shared language
Designer I
communication and collaboration over "confined" infrastructure
Designer II
(a) "Confined" Communication
personal and collaborative work
shared language and meta-knowledge
Designer I
communication and collaboration over "extended" Infrastructure
Designer II
(b) "Extended" Communication

Fig. 1 Confined versus extended communication and knowledge transfer

reviews or in other formal or informal settings. Once an exchange takes place, the participants proceed with their work in isolation. Since this sharing makes use of a confined interface of shared language, we term it "confined" communication and the infrastructure that supports such language sharing and communication is termed a confined infrastructure.

Confined communication is narrow in scope, (only terminology and project parameters such as constraints and issues are shared) and usually, is short in duration (confined to design reviews, detected conflicts, etc.). This is the nature of the interface between disciplines in present engineering practice.

We feel that much more sharing of knowledge between design participants is possible. If we had better understanding of diverse disciplinary knowledge, we could form meta-knowledge about these disciplines that would allow us to:

1. Provide tighter integration between designers through better and deeper understanding of disciplinary concepts. In doing so, reduce effort in, or make more effective, the informal exchange of information between designers and others involved
2. Support the transfer of technical expertise including the transfer of solution methods and the creation of new knowledge
3. Provide a basis for disciplines remote from engineering to understand part of the engineering work and even contribute to it
4. Provide a basis for more efficient managing of projects involving multidisciplinary integrated systems

These provisions are valuable for any collaboration between different designers or disciplines and are representative of the advantages of infused design. We demonstrate the first two provisions by the simple examples in this paper and through the more detailed examples in the companion paper. Demonstrating the last two provisions remains a future task. In Fig. 1b, the *extended communication* infrastructure would consist of mechanisms for sharing language and managing meta-knowledge about diverse domains. The communication could therefore be much more intense than before, and would last longer and occur more often during the development process. Knowledge could be shared between disciplines and communication can become useful even for the purpose of executing tasks presently perceived as purely disciplinary.

Infused design deals precisely with such methods. *Infused design fundamentally changes the nature of communication between design project participants.* More communication does not necessarily mean extra overall design time; communication and cooperation would be more focused and intense and of higher quality. Recall Dutoit's (1996) study showing that extra effort spent in arriving at a common vocabulary reduced design complexity and improved its quality. Consequently, the time saving due to resolving design difficulties or impasses would outweigh the increased effort in communication. Infused design could lead to solving problems with presently unknown solution or could prevent the reinvention of solutions from scratch when readily available solutions exist in other disciplines. As we see later, infused design rests on a firm formal foundation. Nevertheless, studying it requires about a 50 hr course, and executing it does not burden the process since the process steps are simple and can be executed quickly (see examples in the sequel paper, Shai and Reich 2004). Of course, trying to use infused design on simple problems or on each design issue might be unbeneficial and thus waste unnecessary resources.

3 Foundation of "infused design"

Infused design supports the exchange of focused, high quality, critical knowledge between project participants by making use of the relationships between the representations of knowledge in these and other disciplines. If such quality exchange of design knowledge is done across projects, the benefit of infused design is amplified. Even without infused design or the transfer of disciplinary knowledge across disciplinary boundaries, there is a wealth of information that needs to be managed for effective reuse. When new sharing opportunities such as infused design, are introduced, a new layer of possible knowledge transfer is added and needs to be managed for its effective dissemination and reuse.

While we recognize immediately the important role of computational support for such knowledge management, and even though we can offer good solutions for such management (e.g, *n*-dim) (Reich et al. 1999; Subrahmanian et al. 1997), infused design can be done manually. Similar to CE that requires no computer support (Prasad 1996; Reich and Subrahmanian 1992) infused design needs merely careful organizational support. In this paper, we concentrate on the fundamental concept of infused design and leave the computational support to future studies.

Infused design was made possible by *multidisciplinary combinatorial approach (MCA)*, an approach that allows for exchanging disciplinary knowledge across disciplines. Presently, it is seen as the facilitator of infused design by providing the mechanisms for exchanging detailed knowledge across disciplines. Nevertheless, we do not restrict infused design to be dependent solely on MCA as long as alternative approaches for exchanging knowledge are made available.

The idea behind MCA begins with developing general discrete mathematical representations, called combinatorial representations (CR). Then, each combinatorial representation is thoroughly investigated for its embedded properties and relations with other CR. Afterwards, the CR are used to represent diverse engineering problems. The mathematical foundation of the CR mainly comprises three branches of discrete mathematics: graph theory, matroid theory, and discrete linear programming, while the examples in this and the sequel paper exclusively employ the graph representations. We purposefully chose graph theory examples due to the intuitive nature of graphs for representing complex systems but infused design applies well to the other types of representations. Each representation offers additional knowledge about concepts, solution methods, and solutions to specific problems. Additional representations could be added if they could be tied to the meta-level representation.

MCA has already been applied to different engineering fields, leading to significant results. Some of these results are listed in Table 1.

This section introduces the cooperation opportunities made possible by MCA in the context of design activities. We focus on two general types: cooperation through common combinatorial representation (CCCR), and cooperation based on the relations between the combinatorial representations (CRCR). The former cooperation is demonstrated in the sequel paper through cooperation between electrical and mechanical engineers and the latter through cooperation between mechanical and civil engineers, where both are implemented to obtain solutions for design tasks.

3.1 Cooperation through common combinatorial representation—CCCR

CCCR is made possible when designers from different fields use the same combinatorial representation to represent their engineering systems. Sharing the same combinatorial representation opens a channel for transferring knowledge, methods, and designs from one field to the other through this common representation. Furthermore, this transfer of knowledge enables utilization of the knowledge embedded in the common representation.

By means of CCCR, engineers can introduce their colleagues to a method, (either known or new) in the field of the latter, of which their colleagues are unaware. Consider for example, electrical and mechanical engineers working for the same organization. Suppose, the mechanical engineer is requested to analyze an indeterminate truss, an issue that he[1] has not dealt with for a long time. Usually, he would look for help only from a mechanical engineer, but CCCR opens up additional opportunities, such as cooperating with an electrical engineer by means of resistance graph representation (RGR), as depicted in Fig. 2. Such cooperation is possible since the RGR is the isomorphic representation of both electrical circuits and indeterminate trusses (Shai 2001b). The electrical engineer may assist his colleague by applying the "node method" (Balabanian and Bickart1969) from electrical circuits to the RGR (Shai 2001d), expanding it to a multidimensional case using the knowledge embedded in the RGR. This expansion concludes in a systematical manner the "displacement method" for trusses, which is a known method of solving problems for the mechanical engineer (West 1993). This example demonstrates the first three provisions in Sect. 2.

Finally, CCCR enables an electrical and mechanical engineer to assist their colleagues by dealing with specific design tasks. This can be carried out by infusing an existing design from one field into another through the common combinatorial representation. A case study for such an infusion is shown in the sequel paper by Shai and Reich (2004) where a known electronic circuit was transformed into a mechanical design through a common combinatorial representation.

3.2 Cooperation through relations between combinatorial representations (CRCR)

As explained before, one of the main activities in MCA is to explore the relations between different representations. CRCR suggests sharing knowledge and methods between engineering fields that are represented by *interrelated* (and not necessarily isomorphic) combinatorial representations.

One such relation is the duality between the two CR—potential graph representation (PGR) and flow graph representation (FGR) (Shai 2001a). Since the former is used to represent mechanisms and the latter to represent trusses, infused design employs this relation to establish cooperation between structural and mechanical engineers. This issue is outlined in Fig. 3 and will be employed in the design case appearing in the second paper by Shai and Reich (2004).

One of the consequences of CRCR between structural and mechanical engineers is the possibility to infuse in one field ready-made designs from the other. In the sequel paper, Shai and Reich (2004) present an example where a structural engineer is required to design a static system, which amplifies a given externally applied force. Suppose that he is unaware of how to solve the problem and is unable to retrieve the needed information from the literature. When he employs infused design a new possibility emerges, since now he may deal with the kinematical dual problem instead by asking for help from his colleagues from the mechanical engineering community. This dual kinematical problem, on the other hand, has well known and published solutions. What is now left for him is to transform the kinematical system provided by his colleagues and leap back into the field of structures by building the dual static system.

The above two cooperation doctrines—CCCR and CRCR—are only a part of a variety of ways of cooperation that are being established on the basis of the properties in and between combinatorial representations. These types of cooperation together form a framework for convenient and systematic implementa-

[1] The terms "he" and "his" may equally refer to "she" and "her" or "they" and "their."

Table 1 List of achievements derived through MCA

Application	The idea behind the application	Specific applications	References
1. Representing and analyzing integrated engineering systems	Some of the graphic representations were found to be applicable to represent a wide variety of engineering fields, and thus capable of representing in a unified way systems that comprise elements belonging to different engineering domains	Analysis of integrated engineering systems	Shai 2001b; Shai and Rubin 2003
		Representing and analyzing MEMS	Shai et al. 2002; Shai and Rubin 2003
2. Systematic design of engineering systems	Graph representations pave new channels between diverse engineering disciplines, thus enabling transfer of known designs from one engineering field to another. In many cases, known designs in the original field, upon transformation, become new designs in the secondary field.	Known electronic circuits were transformed to yield new mechanism and static system designs	Shai 2003b
		New static systems were devised from known mechanisms	Shai 2002a
		A new concept of a micro-mirror scanning device that overcomes the problem of inter-axial coupling was derived in a systematic way after applying the design techniques partially described in the paper	Currently being patented
3. Finding relations between different engineering fields	Due to the mathematical relations between the graph representations, new relations between engineering fields represented by them were derived	Duality relation between trusses and mechanisms	Shai 2001a; Shai 2002a
		Duality relation between Stewart platforms and serial robots	Shai 2002b; Shai 2003b
		Duality relation between gear systems and beams	Shai 2002b;
4. Deriving new theorems and methods in different engineering domains	Engineering theorems and methods were transformed from one engineering field to another through the relations between the graph representations, thus yielding new theorems and methods	A new graphical method for analyzing trusses was derived from the known vector resolution method in machine theory	Shai 2002a
		A method for decomposition of trusses to basic static groups obtained from the Assur group method	
		Validity theorem in trusses was derived based on the concept of mechanism mobility	Shai 2002a
		A new method for analyzing trusses was derived from the Willis method, a known method in planetary gear trains	Shai and Mohr 2004
5. Revealing hidden properties of engineering systems	Graph representations enable transforming implicit knowledge to explicit knowledge. Properties that are unknown, or "hidden", in one engineering system, become clear in the transformed engineering systems.	New type of force variable was revealed in structures through the duality relation	Shai and Rubin 2003
		Singular configurations of engineering systems, such as trusses and robots were detected by means of their dual engineering systems	Shai 2002b; Shai 2003a
6. Checking the validity of engineering systems	Problems with checking the validity of engineering systems are transformed to problems of checking the formation of the corresponding graph representations	This issue has been employed to establish validity checking methods for engineering systems belonging to a number of engineering fields, such as: dynamical systems, trusses, plane linkages, planetary gear systems and others	Shai and Preiss 1999; Shai 2001b

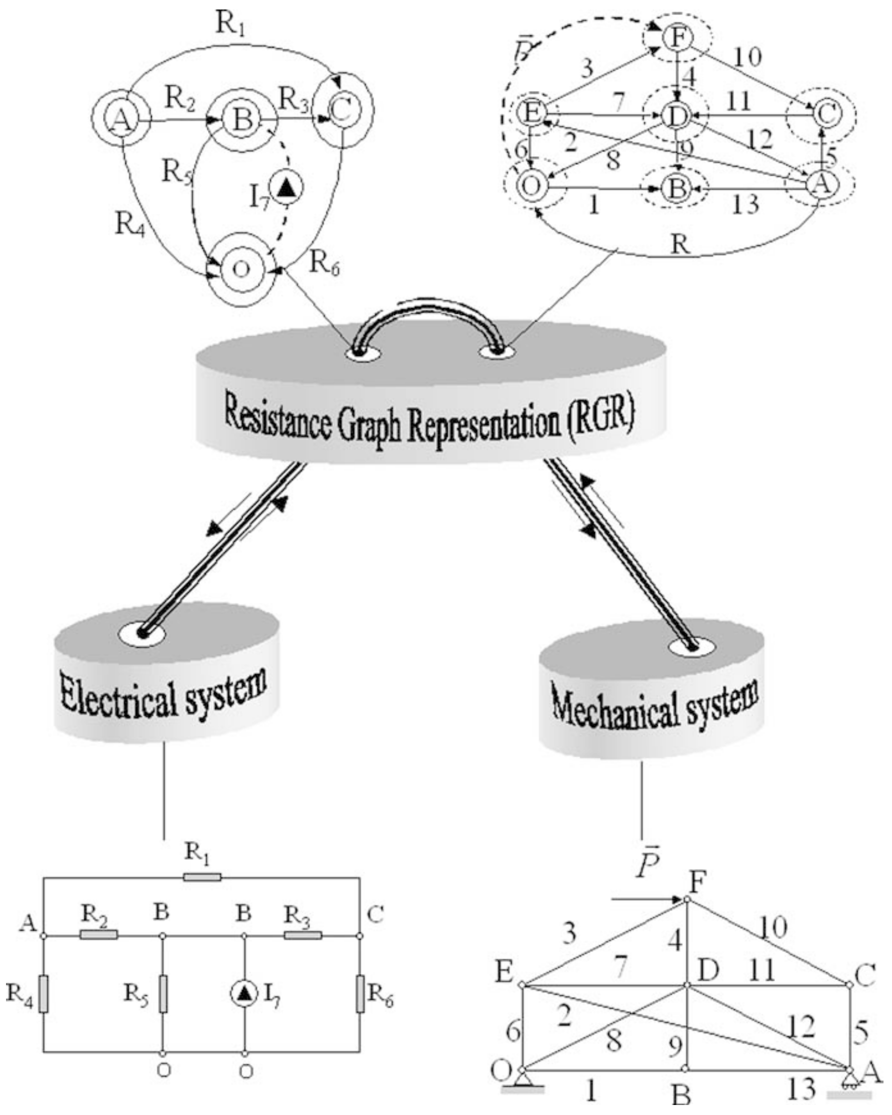
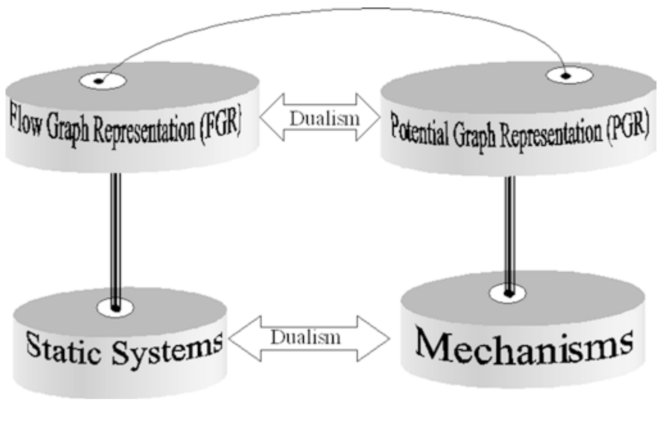


Fig. 2 Implementing CCCR for establishing new knowledge transfer channels between analysis methods for structures and electrical



Flow Graph Representation (FGR)
Dualism
Potential Graph Representation (PGR)
Static Systems
Dualism
Mechanisms

Fig. 3 Derivation of the dualism relation between static systems and mechanisms through CRCR

tion of infused design principles. The effectiveness of this framework mainly stems from the following three factors:

1. It has a representation that is more general than other special representations that it integrates
2. Duality or other relations between different representations are established formally
3. The general representation can be mapped (isomorphically) into the special representations

4 Infused design

Infused design is a new way of designing. We describe it through a sequence of several activities (see Fig. 4): problem formulation, problem modeling and representation, problem solving, and product analysis. This description is more abstract than specifying the process by design phases such as: need identification, conceptual design, and detailed design (e.g., Hubka and Eder 1988; Pahl and Beitz 1984). This sequence may vary and may iterate in many ways, yet, the essence of using infused

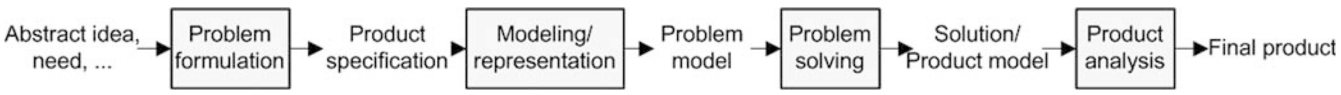


Fig. 4 Design process

design is invariant of the precise process configuration or activity naming.

4.1 Notation

In order to explain the nature of problem representation in infused design we define a notation. In the notation,

- Capital letters denote sets or classes of problems or models
- Small letters denote single items or instances of classes
- Superscripts denote different types of models or representations
- Subscripts denote different sub-parts of a system

Problem related notation includes:

- Initial problem information *p*
- Product specification *ps*

The team possesses a particular disciplinary perspective *D* consisting of:

T	a set of alternative disciplinary terminologies (e.g., different ways to describe force, rod, connection, support, area that are used in structural engineering)
M	the type of the combinatorial representation that governs the creation of a model (some of these types are more common than others within the perspective *D*)
MP	a set of alternative modeling primitives for constructing systems available in *M* (e.g., alternative building blocks for modeling a truss as a graph with force definitions, constraints, etc.)
DK	domain knowledge that assists practitioners in deducing properties of models (e.g., equilibrium equations, Hook's law, compatibility equations, knowledge of truss behavior)
SM	various solution methods that solve the models (e.g., stiffness method, flexibility method)

We define the disciplinary perspective to be $D?T\times MP\times DK\times SM$. With the *D* perspective, the team can use a particular piece of knowledge *dk* and some primitives *mp* to transform the initial problem information *p* into a model *m* described with a particular terminology *t*. The model and its terminology are denoted by the pair (*m, t*). This pair is called the *representation of the problem*. The model of the problem *m* would subsequently be solved by some method *sm*. We proceed to describe the different infused design activities with this notation.

4.2 Problem formulation

Design starts with a written or verbal problem description, an abstract idea, or in another unspecified manner. The initial information is limited, uncertain, and ill defined. Problem formulation is a process of creating a product specification *ps* from the initial problem information *p*. Problem formulation is quite informal and involves (1) making assumptions about the focus of the problem and the importance of different aspects, (2) information consolidation from diverse structured and unstructured sources, and (3) authoring the specification.

The product specification *ps* might include, in addition to its verbal description, drawings, computer models, physical objects, and other available information. It could be incomplete or even inconsistent, thus requiring future revisions.

At this stage, interaction is most valuable. Sharing of information and knowledge emerges from communicating different product perspectives resulting from subjective viewpoints. A shared language is initiated as the basis for future interactions. Disciplinary knowledge can cross boundaries to ease understanding and the evolution of concepts.

In *traditional concurrent engineering*, problem formulation is done collaboratively. Ideally, the design team creates the product specification together and subdivides the problem into sub-problems to be handled by different multi-disciplinary teams. The subdivision is often done to minimize interactions, thus potential conflicts in subsequent design phases. Techniques such as the design structure matrix (DSM) (Browning 2001, Eppinger et al. 1994) could be used to attain such a goal.

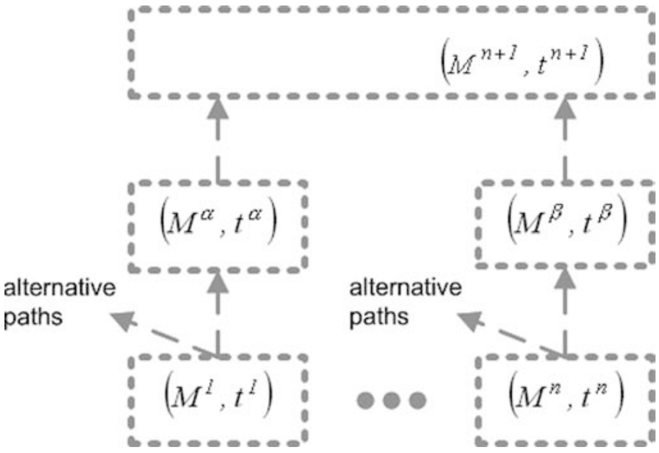
In addition to present practice and all its available tools, infused design facilitates information exchange at a deeper level. For example, while discussing the concept of a beam through a common shared representation, electrical engineers could easily recognize a control system (see Shai and Reich 2004). For mechanical engineers, a control system has a completely different meaning than a beam; therefore, such recognition enriches understanding of concepts, establishing pathways for further elaboration, understanding, and sharing of problem concepts. Infused design also provides ways to improve the problem decomposition by not necessarily minimizing interactions. We leave this provision to another study.

4.3 Problem representation

Problem representation involves creating formal descriptions or models of problems or sub-problems from the informal description in the product specification with further information elaboration. In *traditional concurrent engineering*, this process is done by practitioners working alone or in small groups on their assigned tasks. Infused design changes this activity radically by allowing *integrating* rather than *disintegrating* the activity, thus, juxtaposing knowledge from diverse disciplines to bear on the current problem.

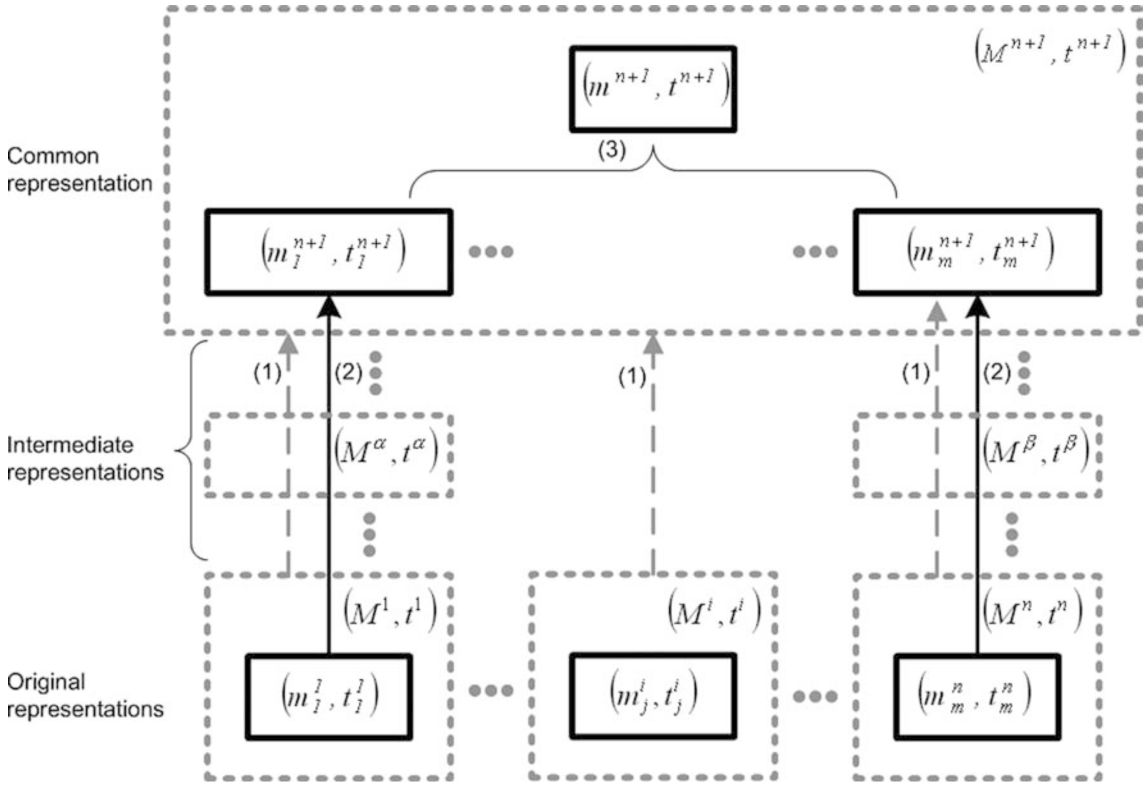
The system being developed can be composed of many components *j*, where *j* = 1,...,*m*, each with its own present representation *i*, where *i* = 1,...,*n*. We denote the component *j* in its representation *i* with $\left(m_j^i, t_j^i\right)$. These indices are independent since one component might have several representations at different times for different purposes; furthermore, several components might have the same representation. Infused design can proceed by integrating the different components representations $\left(m_j^i, t_j^i\right)$, $j = 1, \ldots, m$ into subsystems with a single representation $\left(m_j^{n+1}, t_j^{n+1}\right)$ that could be assembled into the complete system or product representation $\left(m^{n+1}, t^{n+1}\right)$ as shown in Fig. 5. In the figures, the dashed boxes refer to representations such as RGR (Fig. 2), FGR or PGR (both in Fig. 3), and the solid boxes denote specific models of the components. The integration is accomplished by identifying a common representation composed of model type and terminology $\left(M^{n+1}, t^{n+1}\right)$ that can accommodate all the original representations (Fig. 6). Step (1) in Fig. 5 shows this move from the original representations through intermediate representations (e.g., (M^α, t^α)) to the common representation. Subsequently, the original models $\left(m_j^i, t_j^i\right)$, $j = 1, \ldots, m$ are transformed into the common representation $\left(m_j^{n+1}, t_j^{n+1}\right)$ (step 2); and the models can be combined into the complete system model $\left(m^{n+1}, t^{n+1}\right)$ (step 3).

To better illustrate step (1), consider the following example. Members of the multidisciplinary team start by using their customary disciplinary model and terminology for each discipline $\left(M^1, t^1\right) \ldots \left(M^n, t^n\right)$, e.g., PGR for mechanisms and FGR for static systems. In order to integrate all the disciplinary representations they need to search for one common representation $\left(M^{n+1}, t^{n+1}\right)$ that accommodates all the original representations. This



$\left(M^{n+1}, t^{n+1}\right)$
$\left(M^\alpha, t^\alpha\right)$
$\left(M^\beta, t^\beta\right)$
alternative paths
alternative paths
$\left(M^1, t^1\right)$
●●●
$\left(M^n, t^n\right)$

Fig. 6 Identifying common representation



Common representation
$\left(M^{n+1}, t^{n+1}\right)$
$\left(m^{n+1}, t^{n+1}\right)$
(3)
$\left(m_1^{n+1}, t_1^{n+1}\right)$
$\left(m_m^{n+1}, t_m^{n+1}\right)$
(1)
(2)
(1)
(1)
(2)
Intermediate representations
$\left(M^\alpha, t^\alpha\right)$
$\left(M^\beta, t^\beta\right)$
$\left(M^1, t^1\right)$
$\left(M^i, t^i\right)$
$\left(M^n, t^n\right)$
Original representations
$\left(m_1^1, t_1^1\right)$
$\left(m_j^i, t_j^i\right)$
$\left(m_m^n, t_m^n\right)$

Fig. 5 Transforming diverse representations into a common representation

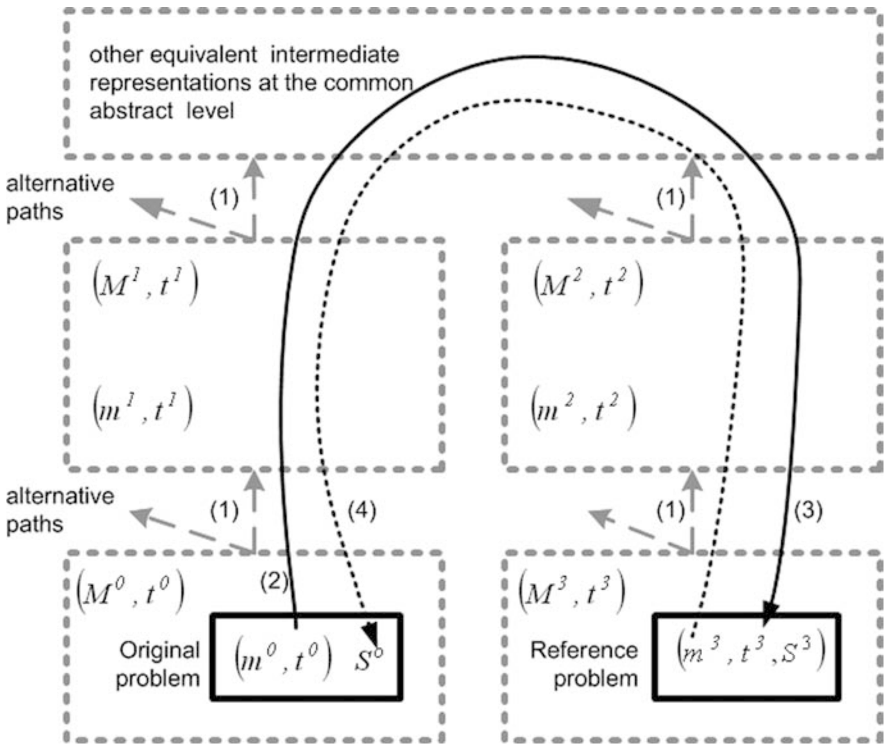


Fig. 7 An abstract model of the method

representation must have a known mathematical relation such as duality or generality with the original representations. This search could be done without considering the particular problem at hand. It is performed following relations found in previous studies.

Finding the path from each representation and terminology into the common representation and terminology $\left(\text{e.g.}, \left(M^1, t^1\right) \ldots \left(M^\alpha, t^\alpha\right) \ldots \to \left(M^{n+1}, t^{n+1}\right)\right)$ is the most critical step as it determines the plan of the process (see later in Fig. 7 and in the examples).

For example, consider solving a problem involving the integration of a Stewart platform and electrical circuit. The search for a common representation follows previously discovered relations between representations. Following Table 1 item 3, the Stewart platform part could be transformed into a serial robots problem; subsequently, this representation could be transformed into a truss problem represented in a resistance graph representation (RGR), as shown in Fig. 2. At the same time, the electrical circuit part could be transformed into RGR as well. Consequently, the search has found RGR to be a suitable common representation that is dual to the two original representations.

The search might not be trivial since alternative paths exist and some of them could be promising while others ineffective in that they would not lead to finding one common representation. As more relations are discovered between different disciplines, the number of candidates for each transformation increases, thus creating more alternative paths that make it possible to choose the most suitable path among the various candidates. As in any domain where multiple paths exist for solving problems, the solution would be based on expert judgment regarding the usefulness of choosing one path over another.

Given the importance of searching and finding the common representation, it is clear that without such representation, infused design cannot be exercised on a given problem. By continuing research on the foundations of infused design new properties between disciplines are discovered, leading to many useful results as summarized in Table 1. In addition, while the examples of transforming between representations discussed here are based on their duality, there are others relations that could be used, for example, the line graph relation between combinatorial representations, and the generalization relation that exists, for instance, when generalizing RGR (resistance graph representation) to resistance matroid representation (Shai 2001c).

4.4 Problem solving and product analysis

In *traditional concurrent engineering* as in classical engineering, problem solving is done by professionals working alone or in small disciplinary teams. In infused design, problem solving could be a collaborative work between the engineers directly in charge of the solution to all others whose knowledge could be valuable as recognized by MCA. The problem representations or models are brought up to the common combinatorial level, where there exist known relations between the combinatorial representations, such as duality. These relations enable engineers from different disciplines to raise new ideas, solution methods, or known solutions from their fields to solve the problem.

The common representation $\left(M^{n+1}, t^{n+1}\right)$ facilitates extended communication between the team members. They can now borrow solution or solutions methods from different disciplines.

For example, suppose one of the problems p of one practitioner is found to be very difficult in his discipline (see Fig. 7). The problem is modeled in m^0, a model of the model type M^0. We can denote this process by $p \xrightarrow{D} m^0$, clarifying that m^0 is created from p by some disciplinary perspective D. Through the previously constructed common representation $\left(M^{n+1}, t^{n+1}\right)$ (step 1), the practitioner transforms the problem model from m^0 to m^1 and onwards, while modifying the terminology (step 2), allowing other team members to translate it into the model m^3 described by their disciplinary terminology t^3 (step 3). The team members can now locate design solutions S^3 for the problem. This solution undergoes the opposite transformation (step 4), resulting in solutions to the original problem S^0.

4.5 Solution composition

Most real products are composed of parts designed separately and integrated together. Therefore, solutions found to sub-problems must be assembled into a complete system. In concurrent engineering, the interface between subsystems is maintained through administrative procedures (e.g., engineering change procedures), negotiations, and resolutions. Infused design can change this practice since it also supports the complete modeling, design, and analysis of multidisciplinary systems through the common representations of the system (mentioned in Sect. 4.3 and Fig. 5) (Reich and Shai2000). This allows a clear understanding of the influences between subsystems and can apply diverse methods brought from the different disciplines into the common representation as well as methods intrinsic to the representation to bear on the problem.

5 Comparing infused design to other approaches

Several aspects underlying infused design relate to well-known topics of contemporary engineering design research. The following is a brief literature review on these topics including 'design by analogy,' 'systematic design,' 'schematic synthesis,' and others, that enable differentiation of infused design from these approaches.

The new designs obtained by infused design are mathematically isomorphic to already known designs in other engineering fields. This is reminiscent of 'design by analogy' that has attracted many researchers in the engineering design and AI communities.

Balazs and Brown (2000), for example, used analogical reasoning for simplifying a design so as to reduce the computation complexity. A computational theory of analogy-based creative design called 'model based analogy' (MBA) was developed by Goel (1997); he used models to represent explicitly the structural elements of a device, its topology, and its function. Unlike infused design, where the mathematical foundation of graph theory and other properties of MCA underlie the process of deriving the design, the majority of the works on design by analogy attempt to simulate the process of how designers and engineers arrive at their solutions. Particularly, in this regard, one should mention the work of Gero dealing with situated analogy in design (Kulinski and Gero 2001).

Additional correspondence to infused design and design by analogy can be traced in works that also employ topological diagrams and graphs. For example, Börner et al. (1996) created a library of design concepts that express topological patterns, and employed the best matching algorithm to retrieve an appropriate design candidate. Another approach that used graphs to restore designs was developed by Qian and Gero (Qian 2002) who represented designs in the form of a function-behavior-structure model.

Infused design enables deriving systematically new engineering designs. This systematicity follows from the mathematical basis underlying combinatorial representations, which gives rise to deterministic rules for treatment of engineering systems. The systematic design approaches, where the designer proceeds with well-defined steps during the design process, are not new in engineering design. Pahl and Beitz (1988) conducted a thorough investigation of all the possible phases of the product design process, upon which they developed a structured methodology outlining the design steps to be followed by an engineer.

Another approach, called TRIZ, for systematic creative design was developed by Altshuller (1988). TRIZ is studied by many scientists and practiced in industry. The principles of TRIZ were developed upon investigating thousands of existing inventions and patents.

As before, the difference between infused design and other systematic design methods is the firm mathematical foundation that supports derivation with provable properties compared to prescriptions that are based on induction from significant engineering experience.

Infused design is by some means related to the works employing schematic description of engineering systems for design. Similar to graph representations used here, the schematic description can be seen as an abstract substitute of the design. Ulrich and Seering (2002) employed a schematic description describing the topology of engineering systems to produce new engineering designs. Given a design problem in a form of a function of input-output relation, they generated initial 'candidate' systems, constructed the corresponding 'compact descriptions,' and applied modifications upon them to adjust the system behavior to the problem requirements.

General design theory (GDT) assures that with sufficiently detailed knowledge, a specification can evolve incrementally until it results in an approximate solution (Tomiyama and Yoshikawa 1987; Reich 1995). This evolution is performed using models constructed from a meta-model picked from a meta-model set (Tomiyama et al.1989). However, the decision about which meta-model to choose remains outside the scope of GDT. In their subsequent work, Tomiyama and colleagues attempted to create a system that would perform or support such an operation by employing logic formalisms such as abduction; however, the results remained limited (Tomiyama et al.2002). In Braha and Reich (2003), a more general model of synthesis than GDT was presented but it also left the particular knowledge that drives the synthesis steps outside the scope of the theory.

In order to allow the use of multiple models, Tomiyama et al. (2002) proposed to use a common ontology that also could support the consistency between the models. This ontology is domain dependent, time consuming to develop, and error prone. Infused design specifically shows how methods and solutions could be generated systematically from corresponding methods and solutions in other disciplines. It guarantees the correctness of results not by using domain dependent ontology but by relying on general ontology of systems that is embedded in the different representations.

6 Discussion and conclusions

Infused design is a new way of addressing multidisciplinary product development projects. It allows members of the development team to exchange knowledge in fundamentally different ways than is presently conceived. From the present state of sharing project documents or parameters and common language, we expand the scope of sharing to include a vital exchange of knowledge consisting of disciplinary concepts, problem formulations, analysis methods, and solutions. This exchange crosses and breaks disciplinary boundaries.

Infused design is most beneficial in breakthrough design since it enables transferring concepts from one field to another. It could apply also to variant design by checking whether we stay with the present concept or look for a better one in different disciplines. Whether one chooses to use it depends on the difficulty of the problem.

Infused design opens avenues for bootstrapping our understanding and supporting various design practices such as creative design. The bootstrapping effect comes from an interesting and non-trivial phenomenon of infused design. When we used infused design to perform design, we understand the underlying representation better. We can also uncover more properties of the representation that, in turn, would improve our ability for analysis. Subsequently, this can lead to yet better design ability. We leave the illustration and a more detailed explanation of this phenomenon to a future study due to its complexity and simply note its existence here.

Several questions arise with regard to attaining infused design in practice. Their answers are still preliminary and require more experience with the method:

1. Can infused design scale to handle large problems?
   In principle, as was explained in detail in the paper, the mathematical foundation of infused design is mainly graph theory. Graph theory is one of the branches of discrete mathematics, which is the mathematical foundation of computer science. Specifically, the processes described in the paper are amenable for polynomial time computerizations, which would make employing the methodology possible in large-scale systems.
   In practice, so far, we have experience with small size problems. Practical scalability to larger problems requires several things:
   a. The use of support systems to do the transformation between representations.
   b. Exercising careful modeling in which large problems are decomposed into smaller components. Infused design can apply to the sub-problems as well as to the integration.
2. How do we weigh the cost of utilizing infused design and its benefits?
   Engineers are quite adept at managing their resources and using them only when needed. Infused design can be viewed as a resource for addressing complex problems. These arise, for example, with the need to compete with existing quality products, or when a solution to a new design problem is unavailable. In such circumstances, infused design could provide a competitive edge whose utilization is inexpensive.
3. How can infused design be taught and at which student level?
   We have experience in teaching MCA—the foundation of infused design—to high school students with great success. High school students were able to use the method to solve complex problems that won project prizes from the Israeli government, academia, and industry (see Shai 2001c). We certainly experienced teaching it to undergraduate mechanical engineering students. Therefore, we see no obstacle to introducing infused design into design practice from this perspective.
4. How can infused design be embedded in an organization?
   Even though it is not difficult to teach infused design, there is no need to introduce it into an organization in a comprehensive way. It seems that a reasonable approach would be to introduce it as part of the initial stages of a new product development project, when introducing new technologies such as MEMS, or when engaging in new collaborative efforts. All these circumstances present serious opportunities for exercising beneficial infused design. The success from such uses would lead other organization members to study and use the method as well.

5. Does infused design undermine the need for disciplinary expertise?
   This might have been a tough issue for infused design if the answer was positive. However, the value of expertise does not diminish since we can only transfer the aspects that can be formalized in MCA and not the presently informalized practical experience that is rooted in context. The informal part of design is as significant as its formal aspect (Subrahmanian et al. 1993a). For example, while infused design can transfer solutions between disciplines, their quality in one discipline does not guarantee their quality in the second discipline. Furthermore, some of the steps in infused design involve searching in multiple paths or selecting between alternative translations between representations. These choices are based on experience and expertise. Moreover, infused design creates new opportunities for becoming experts, e.g., in the use of common representations to solve problems.
6. Might practitioners be reluctant to use infused design?
   Some practitioners might feel that infused design is too difficult due to the complex mathematical concepts involved. We have stated above that these concepts can be taught even at the high school level. Others might fear that infused design opens their knowledge to further review—the introduction of infused design forces them also to compete for excellence with other disciplines in their own field. This is only partially correct but unavoidable. Allowing deep knowledge transfer exposes deficiencies of existing knowledge. On the positive side, infused design allows experts to demonstrate their value in new disciplines.
7. What is the role of computational support in infused design?
   The advent of infused design makes information from seemingly irrelevant disciplines be highly appropriate for addressing various problems. In order to utilize this valuable knowledge properly, information systems could be used. Furthermore, real projects for which these technologies are rewarding are complex and interdisciplinary, involving multiple teams or organizations. The complexity of these projects and the desire to recoup all the opportunities of infused design make *information infrastructures* central to infused design. The task of the infrastructure is the managing of all types of information and knowledge and their communication between project participants.
   Infused design can be supported by information systems; however, it is independent of any such implementation. Infused design could be carried out in a similar way to CE, based solely on organizational support. Section 4 discussed the process of infused design independent of a support system although it is clear where such systems could fit in. Since the complexity of interdisciplinary projects can be overwhelming to handle manually, we regard a suitable information infrastructure as one of the foundations of infused design.

Infused design does not address the language problem discussed in Sect. 2 directly. It can even be argued that it exacerbates it with additional concepts (e.g., those related to MCA). Nevertheless, we have experience teaching the ideas to high school and undergraduate students and the results are promising. As we already mentioned, high school students, and undergraduate students were able to use infused design successfully to solve engineering problems. In addition, although not shown, we feel that infused design could possibly bootstrap the creation and evolution of shared language. When interdisciplinary team members discuss the naming of concepts and their meaning, they could benefit from the wealth of disciplinary knowledge they can now share, hence improving their shared basis and understanding.

References

Ahmed A, Wallace KM, Blessing LTM (2004) Understanding the differences between how novice and experienced designers approach design tasks. Res Eng Des (in press)

Akao Y (ed) (1990) Quality function deployment. Productivity Press, Cambridge, MA

Altshuller G (1988) Creativity as an exact science. (translated by Anthony Williams) Gordon and Breach, New York

Balabanian N, Bickart TA (1969) Electrical network theory. Wiley, New York

Bechkoum K (1997) Intelligent electronic mock-up for concurrent engineering. Expert Syst Appl 12(1):21–35

Balazs ME, Brown DC (2000) Design simplification by analogical reasoning. In: Proceedings of the KIC-4: fourth IFIP WG 5.2 workshop on knowledge intensive CAD, Parma, Italy

Börner K, Coulon CH, Pippig E, Tammer EC (1996) Structural similarity and adaptation. Advances in case-based reasoning. In: Smith I, Faltings B (eds) Proceedings of the third European workshop on case-based reasoning (EWCBR-96), Springer, Berlin Heidelberg New York, pp 58–75

Boujut J-F, Tiger H (2002) A socio-technical research method for analyzing and instrumenting the design. J Des Res 2(2)

Bowen J, Bahler D (1992) Frames, quantification, perspectives, and negotiation in constraint networks for life-cycle engineering. Artif Intell Eng 7(4):199–226

Braha D, Reich Y (2003) Topological structures for modeling engineering design processes. Res Eng Des 14(4):185–199

Browning TR (2001) Applying the design structure matrix to system decomposition and integration problems: a review and new directions. IEEE T Eng Manage 48(3):292–306

Clarkson PJ, Hamilton JR (2000) Signposting, a parameter-driven task-based model of the design process. Res Eng Des 12:18–38

Coates G, Whitfield RI, Duffy AHB, Hills B (2000) Coordination approaches and systems—Part II: an operational perspective. Res Eng Des 12:73–89

Coulter N, Monarch I, Konda S (1998) Software engineering as seen through its research literature: a study in co-word analysis. J Am Soc Inform Syst 49(13):1206–1223

Dutoit A (1996) The role of communication in team-based software engineering projects. PhD Thesis, Carnegie Mellon University, Pittsburgh, PA

Eastman C, Jeng TS (1999) A database supporting evolutionary product model development for design. Automat Constr 8:305–323

Eppinger SD, Whitney DE, Smith RS, Gebala DA (1994) A model-based method for organizing tasks in product development. Res Eng Des 6:1–13

Erkes JW, Kenny KB, Lewis JW, Sarachan BD, Soboiewski MW, Sun JRN (1996) Implementing shared manufacturing services on the world-wide web. Commun ACM 39(2):34–45

Finger S, Konda S, Subrahmanian E (1995) Concurrent design happens at the interfaces. AI EDAM 9(2):89–99

Garrod S (1998) How groups co-ordinate their concepts and terminology: Implications for medical informatics. Method Inform Med 37(4–5):471–476

Goel A (1988) Design, analogy and creativity. IEEE Expert 12(3):62–70

Hubka V, Eder WE (1988) Theory of technical systems: a total concept theory for engineering design. Springer, Berlin Heidelberg New York

Konda S, Monarch I, Sargent P, Subrahmanian E (1992) Shared memory in design: a unifying theme for research and practice. Res Eng Des 4(1):23–42

Kulinski J, Gero JS (2001) Constructive representation in situated analogy in design. In: de Vries B, van Leeuwen J, Achten H (eds) CAADFutures 2001, Kluwer, Dordrecht, pp 507–520

Marcovici M (1999) Efficient mechanical rectifier. US Patent Number 5,931,062

McMahon CA, Xianyi M (1996) A network approach to parametric design integration. Res Eng Des 8:14–32

Messac A, Chen W (2000) The engineering design discipline: is its confounding lexicon hindering its evolution? J Eng Eval Cost Anal 3:67–83

Minneman S, Leifer L (1993) Group engineering design practice: the social construction of a technical reality. Proceedings of the international conference on engineering design (ICED'93), The Hague, Netherlands, pp 301–310

Neches R, Fikes RE, Finin T, Gruber TR, Senator T, Swartout WR (1991) Enabling technology for knowledge sharing. AI Mag 12(3):36–56

Pahl G, Beitz W (1988) Engineering design: a systematic approach. In: Wallace K (ed) The design council. Springer, Berlin Heidelberg New York

Park H, Cutkosky MR, Conru AB, Lee S-H (1994) An agent based approach to concurrent cable harness design. AI EDAM 8(1):46–61

Prasad B (1996) Concurrent engineering fundamentals, vol I: integrated product and process organization. Prentice Hall, New Jersey

Qian L (2002) Creative design by analogy. In: Chakrabarti A (ed) Engineering design synthesis—— understanding, approaches and tools. Springer Berlin Heidelberg New York, pp 245–269

Reich Y (1995) A critical review of general design theory. Res Eng Des 7:1–18

Reich Y (2000) Improving design rationale capture with QFD. Eng Comput 16(3–4):236–252

Reich Y, Shai O (2000) MEMSIS—A MEMS information system. In: Shpitalni M (ed) Proceedings of CIRP design seminar, Haifa, Israel, pp 193–197

Reich Y, Subrahmanian E (1992) Concurrent engineering: an extended view. Proceedings of the AID'92 workshop on concurrent engineering, Kluwer, Netherlands

Reich Y, Konda S, Levy SN, Monarch I, Subrahmanian E (1993) New roles for machine learning in design. Artif Intell Eng 8(3):165–181

Reich Y, Konda S, Levy SN, Monarch I, Subrahmanian E (1996) Varieties and issues of participation and design. Des Studies 17(2):165–180

Reich Y, Subrahmanian E, Cunningham D, Dutoit A, Konda SL, Patrick R, Westerberg AW, The n-dim group (1999) Building agility for developing agile design information systems. Res Eng Des 11(2):67–83

Sargent P, Subrahmanian E, Downs M, et al. (1992) Materials' information and conceptual data modeling. In: Barry TI, Reynard KW (eds) Computerization and networking of materials databases: third volume. ASTM STP 1140, American Society For Testing and Materials

Schrage M (1991) Shared minds: the new technologies of collaboration. Random House, New York

Shai O (2001a) The duality relation between mechanisms and trusses. Mech Mach Theory 36(3):343–369

Shai O (2001b) The multidisciplinary combinatorial approach and its applications in engineering. AI EDAM 15(2):109–144

Shai O (2001c) Combinatorial representations in structural analysis. Comput Civil Eng 15(3):193–207

Shai O (2001d) Deriving structural theorems and methods using Tellegen's theorem and combinatorial representations. Int J Solids Struct 38:8037–8052

Shai O (2002a) Utilization of the dualism between determinate trusses and mechanisms. Mechan Mach Theory 37(11):1307–1323

Shai O (2002b) Duality between statical and kinematical engineering systems. The sixth international conference on computational structures technology, Prague, Czech Republic, 4–6 September

Shai O (2003a) Transforming engineering problems through graph representations. Adv Eng Inf 17(2):77–93

Shai O (2003b) Design through common graph representations. In: ASME Design Engineering Technical Conferences, Chicago, 2–6 September

Shai O, Mohr E (2004) Transforming engineering knowledge through graph representations: transferring the Willis method to linkages and trusses. Eng Comput 20(1): 2–10

Shai O, Preiss K (1999) Isomorphic representations and well-formedness of engineering systems. Eng Comput 15:303–314

Shai O, Reich Y (2004) Infused design: practice. Res Eng Des (in press)

Shai O, Rubin D (2003) Representing and analyzing integrated engineering systems through combinatorial representations. Eng Comput 19(4):221–232

Shai O, Eliahou-Niv S, Rubin D, Slavutin M, Andrusier A (2002) Multidimensional graph representations for MEMS and their applications. ISRAMEMS—the 1st national conference of the Israeli MOEMS society, Haifa, Israel, 21 October

Steward DV (1981) The design structure system: a method for managing the design of complex systems. IEEE T Eng Manage 28:71–74

Subrahmanian E (1991) Notes on empirical studies of engineering tasks and environments. Invited position paper in NSF workshop on information capture and access in engineering environments., Ithaca, NY, pp 567–578

Subrahmanian E, Konda SL, Levy SN, Reich Y, Westerberg AW, Monarch IA (1993a) Equations aren't enough: informal modeling in design. AI EDAM 7(4):257–274

Subrahmanian E, Coyne R, Konda SL, Levy SN, Martin R, Monarch IA, Reich Y, Westerberg AW (1993b) Support system for different-time different-place collaboration for concurrent engineering. Proceedings of the 2nd IEEE workshop on enabling technologies infrastructure for collaborative enterprises (WET ICE), Los Alamitos, CA, IEEE Computer Society Press, pp 187–191

Subrahmanian E, Reich Y, Konda SL, Dutoit A, Cunnungham D, Patrick R, Thomas M, Westerberg AW (1997) The n-dim approach to building design support systems. Proceedings of ASME design theory and methodology DTM '97, ASME, New York

Tomiyama T, Yoshikawa H (1987) Extended general design theory. In: Yoshikawa H, Warman EA (eds) Design theory for CAD. North-Holland, Amsterdam, pp 95–130

Tomiyama T, Kiriyama T, Takeda H, Xue D, Yoshikawa H (1989) Metamodel: a key to intelligent CAD systems. Res Eng Des 1(1):19–34

Tomiyama T, Yoshioka M, Tsumaya A (2002) A knowledge operation model of synthesis. In: Chakrabarti A (ed) Engineering design synthesis—understanding, approaches and tools. Springer, Berlin Heidelberg New York, pp 67–90

Ulrich KT, Seering WP (2002) Synthesis of schematic descriptions in mechanical design. In: Chakrabarti A (ed) Engineering design synthesis—understanding, approaches and tools. Springer, Berlin Heidelberg New York, pp 153–177

Walton GF (1991) Concurrent engineering. Special feature of IEEE spectrum, July 1991

West HH (1993) Fundamentals of structural analysis. Wiley, New York

Whitfield RI, Coates G, Duffy AHB, Hills B (2000) Coordination approaches and systems—part I: a strategic perspective. Res Eng Des 12:48–60