INTERNATIONAL CONFERENCE ON ENGINEERING DESIGN, ICED'07
28 - 31 AUGUST 2007, CITE DES SCIENCES ET DE L'INDUSTRIE, PARIS, FRANCE

PREDICTING THE UNPREDICTABLE PROBLEMS IN MECHATRONICS DESIGN

Valentina D'Amelio and Tetsuo Tomiyama

Delft University of Technology

ABSTRACT
An excellent product should be designed without failures. While multi-disciplinary design generates products with better performances, it can cause design troubles difficult to fix. On one hand, there are interactions among elements of the system that enhance the functionality of the system (*"desired" interaction).* On the other hand, interactions that could create non-trivial problems in the system *("undesired" interaction).* In designing a multi-disciplinary product, the distinction between these two deserves attention. This paper aims to develop a method to detect unpredictable behaviors in multi-disciplinary design such as mechatronics. A tool to detect "*desired*" or "*undesired*" interference among different design domains (components, physical phenomena, and parameters) can be developed based on qualitative physics. The ideas developed in the paper will be illustrated with an example of rotary encoder. This tool is useful to shorten the product development time by "predicting the unpredictable" failures.

Keywords: Design, mechatronics, failures, interaction, predictable interactions, unpredictable interactions, constructive couplings, destructive couplings, Design Interferences Detector.

1 INTRODUCTION
The complexity of products is increasing drastically and consequently, the mechatronics product development processes also has become complex. In designing mechatronics systems the engineer has to deal with multi-disciplinary devices but there is not a unified method like a conventional engineering discipline. The main obstacles are strong separation of related disciplines, and the lack of a common language. Due to these obstacles, companies need longer development processes than expected. Because of unpredictability of the process, a physical prototype is often essential for the development of a mechatronics product. Because experiments and verification of the conceptual design can only be performed on such a physical prototype, this process is repeated many times until results become the desired one. This is time-consuming and cost-inefficient.
An ill-machine is a machine with failures and there are three fundamental reasons. First, the large amount of domains and components involved in such complex systems increase human errors. Engineers have to handle a wide variety of elements crossing several disciplines. While there are research efforts to develop automated design synthesis tools [1], these tools are limited to a single domain.
Second, the lack of a common language among disciplines causes many troubles. There is no unified design methodology to develop a mechatronics machine. This forces an engineer to follow trial-and-error methods. This lack of a common language is a consequence of engineering education, still anchored only to one domain [2]. Indeed "*Mechatronics is a synergistic combination of precision mechanical engineer, electronic, control and system thinking in the design of products and manufacturing processes"* [3]. This definition indicates that mechatronics is not a pure, single discipline but a combination of disciplines. Technologies of systems engineering, such as Bond Graph [4], [5], can in part fill this gap. However, these approaches do not allow easy reuse of models after experimental validation, easy data exchange among different domains and extrapolated use of models in various contexts, or creation of a link between various physical representations of the same object [2].

Third, mechatronics is not only the *"constructive coupling"* of domains that enable a system to work but even the *"destructive coupling"* of domains that generates unpredictable problems. *Unforeseeable (unpredictable)* problems can appear typically in those multi-disciplinary products. The more complex the machine is designed, the more chances of unpredictable problems can happen. It means that two components or modules can interact in an unexpected and unwanted way, dragging the system to work improperly. These unpredictable problems are the consequence of insufficient integration of domain knowledge. This third point is the main topic and engine of this article.
The paper is an attempt to introduce a method to deal with unpredictable problems as consequences of destructive couplings. Such interactions cannot be easily detected. Often only after building a physical prototype, nasty connections appear and they become even more destructive because they require changes at the conceptual design level. The destructive coupling complicates the design process, and as a consequence, time and budget overrun. In the future a tool called a Design Interferences Detector (DID) shall be developed. With DID, the design team can envision destructive problems well in advance, but not only after a prototype machine is developed. Ahead of the prototype, the DID aims to provide a proof for the design.
The DID is well different from both FTA (Fault Tree Analysis) and FMEA (Failure Mode and Effect Analysis). Indeed, they deal with problems coming from deterioration of the system and faults are consequences of such anomalous circumstances. In contrast, unpredictable problems are generated by the intrinsic design of the system; in nominal circumstances the machine can generate anomalous or unexpected behaviors. The article will show how problems are generated by a combination of physical phenomena and states of the entity.
We first give some definitions of terms, and then the third section will show an example of unpredictable problems, constructive and destructive couplings with descriptions of a rotary encoder to clarify the concept. Although this is an established, well-known component, tricky problems can happen. Section 4 shows the complexity in a mechatronics design quantified by feasible interactions among entities and how the system can be simplified and tested within DID. Model-based reasoning and an extension of Function-Behavior-State modeling [6] constitute the foundation of the methodology. We present in Sections 5, 6, and 7 the details of the method to build DID's knowledge representation, a qualitative physics reasoning system (QRS), and a filter that can select unpredictable behaviors among subsystems and diminishes spurious solution. Conclusions end up the paper.

2 DEFINITIONS

Predictable problems: behaviors that a designer takes into account in the conceptual design based on his/her own experience. The system is reconfigured using for instance redundancies. Predictable problems are also named predictable or expected behaviors.
Unpredictable problems: unexpected behaviors that occur within a domain or by interactions of domains. They are named also unexpected behaviors.
Desired interaction: interactions that enable the system to work properly
Undesired interaction: causes of failures in the system, generally given by unwanted communication of different domains.
Constructive coupling: desired and predictable interactions that results in desired and predictable behaviors.
Destructive coupling: undesired and unpredictable interactions that results in undesired and unpredictable behaviors.

For obvious reasonos, the combinations between desired and unpredictable behaviors and undesired and predictable behaviors have no reason to be analyzed.

3 DESIGN IN A MULTI-DISCIPLINARY DOMAIN

This section illustrates a design case in which multi-disciplinarity causes undesired and unpredictable couplings among design parameters or phenomena, although multi-disciplinarity itself is useful to achieve superior functionalities. This section aims also to clarify the previous definitions by means of an example.
Figure 1 shows a rotary encoder used very frequently in mechatronics machines to measure (angular) speed or position. Depending on the application, an encoder with appropriate accuracy has to be selected. Although this seems trivial and easy to perform, it can fail due to unpredictable coupled with

other elements. For instance, an encoder is mounted on a shaft to control the angular speed of the shaft. This connection between the encoder and the shaft is necessary for the encoder to perform its function and is *predictable* and *desired.* We can call this a *constructive coupling*. The encoder contains a photo-detector that collects and transforms angular information into a signal. This connection with the photo-detector is a constructive coupling, too. The encoder is the interface between shaft and software to obtain the position information of the shaft and to control the shaft rotation.
In a real physical system, however, the eccentricity of the shaft is unavoidable and behaviors resulting from the eccentricity can disturb other parts of the system. The eccentricity provides periodic errors that will probably have a sinusoidal shape and a repetition of the fault every period. If there is a repetition of errors, we can in some way predict the error and compensate the transmitted information.
Although it might lead to a *destructive coupling*, the eccentricity of the shaft is *undesired* but *predictable*, because a good designer can predict it from experiences and come up with workaround measures.
This eccentricity problem happens due to the design choice, therefore is a good example of *complexity by design* [7]. Other examples of those undesired but predictable problems that the designer may face during the design involving rotary encoders include tilted axis rotation, radial deviation of the bearing, torsional stiffness, and axial oscillation of the track.

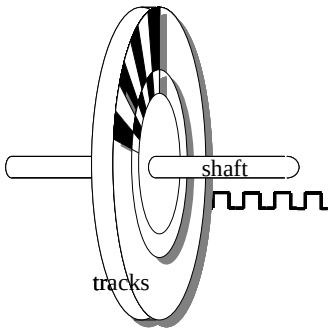

Figure 1. Rotary encoder

However, there is another type of troubles that cannot be solved easily. Suppose that another component operates at the same frequency of the error generated by the eccentricity. There can be interference between the two signals due to cross talk, which causes a peak in the frequency spectrum and the information provided by the encoder can become incorrect. Even if the eccentricity's error remains acceptable, this new situation leads to a *destructive coupling*.
One solution for this problem could be to change the frequency of one of the two subsystems, but this may result in other problems and cause a chain of changes that become too hard to fix. Another solution is to introduce a notch filter to cut off undesired signals. However, in any case, both of the solutions result in major design changes at the conceptual level that can mean extra costs and delay of the project.
In the next section a method to detect such problems will be introduced.

4 DID: MOTIVATIONS AND METHODOLOGY

This chapter first formalizes a complex system, second introduces the method DID is using to handle such complexity.
Well-known techniques to deal with complexities are Design Structure Matrix (DSM) [8] and Suh's Axiomatic Design (AD) [9]. Suh in his axiomatic design suggests a method for the elimination of this "spaghetti codes" through the identification of a logical structure to define classes. His famous independence Axiom is: Maintain the independence of Functional Requirements (FRs).
However, these methods cannot deal with cases well in which the complexity increases unexpectedly during the course of design due to *undesired* and *unpredictable* interactions among subsystems. Because a large amount of interactions appears, the probability of finding out unpredicted problems is very high, because it is not easy to consider each single possible interaction. In addition, neglected interactions can generate a chain of failures like a butterfly effect that can enormously change the output of the machine.

4.1 Motivations

A system is said to be complex when many disciplines are relevant and a large number of elements are present in the process. Even a very complicated machine can be decomposed into smaller modules. Within a module, we would like to assume that every interaction among components is well predicted and unpredictable problems do not appear inside the module. A healthy behavior of a module comes from sound design based on the designer's experience.
However, during the design, two different situations can happen: a new component needs to be added or the whole module needs to be redesigned. In the first case we need to detect eventual interactions caused by the added component, while in the second case interactions among all components need to be examined from the scratch. It is important to understand the influence of each component on the behavior of the entire system or of other modules. The interaction is bidirectional. A component can influence on a module and its environment, and *vice versa*. The consequence is a no-trivial analysis and in the worst case redoing of machine's conceptual design is requested. These considerations are significant even in a single domain and in a mechatronics system design, the degree of interactions increases in the following manner (1).
The number of interactions increases as the number of domains and the number of components increases not linearly but in the 2nd power. This emphasizes the relevance of possible interferences in a multi-disciplinary structure because the power of two appears related to the domains. The maximum number of possible interferences includes wanted, unwanted, predictable and unpredictable couplings as well as no-interactions.

$$Max\_num\_interactions = 2 \bullet m^2 \bullet \sum_{i=1}^{n}(i-1) \qquad (1)$$

$$= 2m^2 \frac{n(n-1)}{2} = m^2 n(n-1)$$

m= number of domains
n= number of components

Figure 2 shows how the situation gets entangled even with two components; in the worst case 32 interferences appear.
In modern mechatronics machines generally more than thousand components are included and many disciplines are interacting in the process. The consequence is an incredible amount of possible unexpected interactions that can easily lead to failures and that the designer can easily and erroneously neglect. These unexpected *interactions* lead to unexpected *behaviors* and unexpected behaviors leads to unexpected *functions* for the machine.

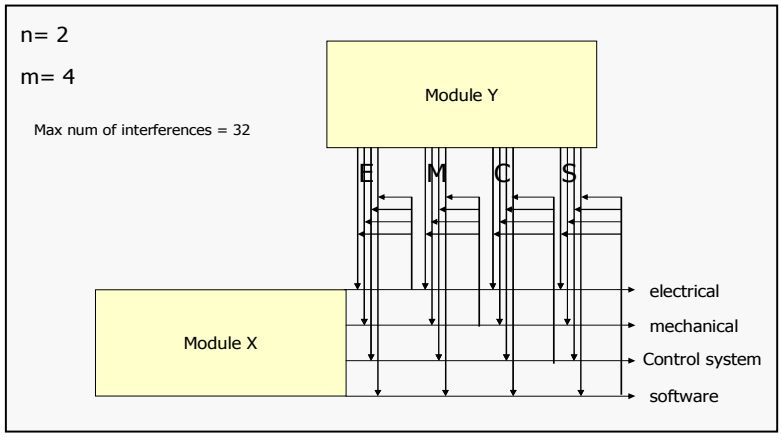

Figure 2. Interferences among two components

This suggests the need to develop a tool to automatically analyze the conceptual design of the machine, to predict if nasty interactions can ever occur, and to avoid unexpected relationships among domains at a later stage of the design. The *Design Interferences Detector (DID)* aims to accomplish this goal.

4.2 Methodology

DID is composed of a *conceptual design model* that the engineer describes, a *reasoning mechanisms* to detect all possible behaviors, and a *filter* to focus on possible problems (Figure 4). Model-based reasoning and Function-Behavior-State modeling constitute the foundation of the methodology.
The following explains how DID detects "unexpected" behaviors.

- The designer describes behaviors that are required to happen together with conceptual description of the machine.
- DID reasons out all possible physical behaviors that can occur using a QPT (Qualitative Process Theory) [10] based reasoning system (QRS). These generated behaviors can include both expected and unexpected behaviors.
- DID filters out unexpected behaviors by separating expected behaviors from all possible reasoned out behaviors. Behaviors that are not included in the predefined (expected) behaviors might cause potentially unexpected functions.
- The designer can now distinguish predictable expected behaviors from unexpected problems. The designer is now informed about the future potential problems, which would suggest the necessity for further analysis of problems and modification of the design (Figure 3-4).

The next chapter describes all the steps to detect unpredictable problems with more details. .

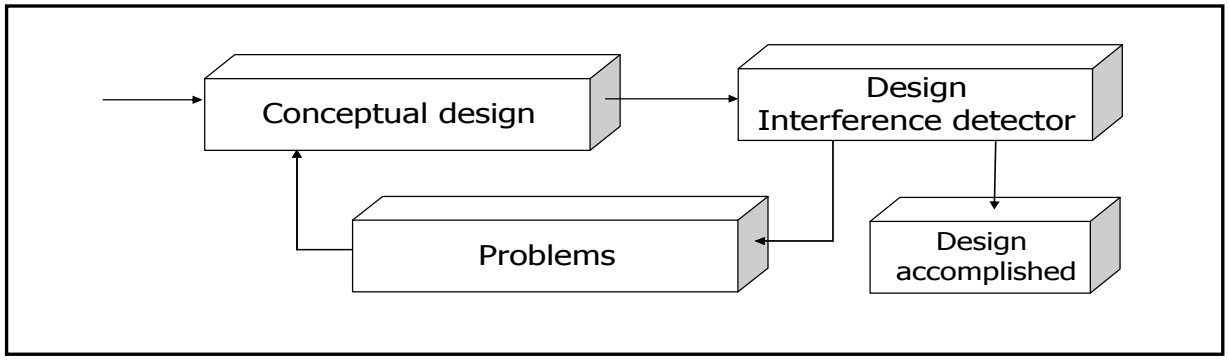

Figure3. Location of DID

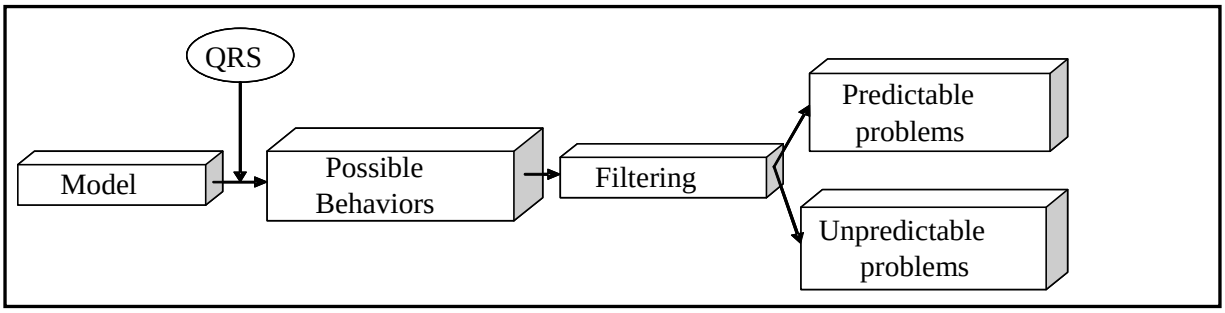

Figure 4. DID processes

5 MODEL: KNOWLEDGE REPRESENTATION SCHEME

The DID model is an extension of the FBS model [11,6]. The FBS model focuses on the design process and provides both a methodology to structure knowledge and QRS reasons out all the possible behaviors that can occur, thereby establishes connections between states and behaviors [12]. This design methodology is based on decomposing high level functions into lower level sub-functions till they can be associated with some physical embodiment. The physical embodiment is conducted, based on catalogues of physical features by which functions are associated with components through behavior.

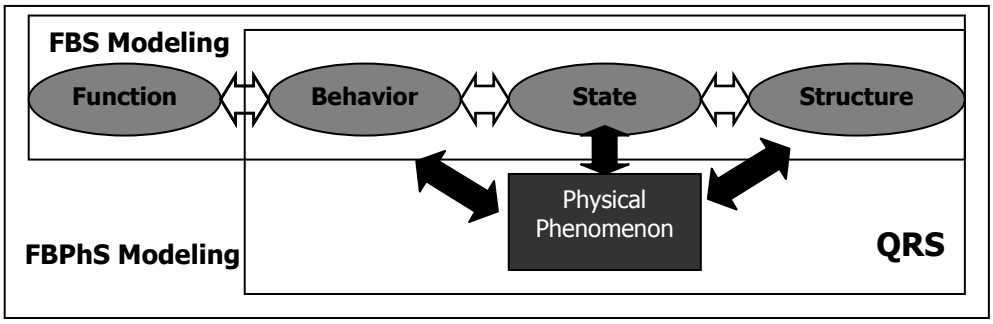

Figure 5. FBPhPhS general model description

In FBS, a behavior is the link between function and state and eventually structure. The relation between function and behavior is subjective, because a function is a description of behavior abstracted by human through recognition of behavior in order to utilize the behavior [13]. In contrast, a relation between behavior and state is more objective based on a physical principle. A state of an entity is defined as a set of attributes and relations among relevant entities. A behavior is a temporal sequential state transition and physical phenomena determine behavior of an entity.

Once all entities and relations among them are defined on FBS, QRS reasons out physical behaviors based on QPT. The reasoning results of QRS include not only predicted behaviors but also other phenomena that are likely to happen. Such unpredicted physical phenomena can lead to unpredicted functions. However, since FBS does not deal with physical phenomena explicitly, FBS cannot reasons out unpredicted functions directly.

Behaviors can influence on state and structure, too. In other words, a behavior change can lead to a new state (because the behavior is basically a state change), which in turn initiates new behaviors. These mutual dependencies among behaviors and state (hence structure) are governed by physical phenomena but cannot be directly reasoned out, because of the separation of FBS and QRS.

These are the reasons why we need to develop DID as an extension of FBS that incorporates QRS. Figure 5 illustrates the general model representation for DID, which is called FBPhPhS (Function-Behavior-Physical Phenomenon-State model). FBPhPhS explicitly includes physical phenomena between behavior and state and is an integration of FBS and QRS. An example of how unexpected behaviors are generated by FBPhPhS is shown in Figure 6. The physical phenomena represented in the FBPhPhS can be due even to different domains.

The concept of *attributes of relations* is included in FBPhPhS to prioritize reasoned out behaviors. A relation such as “connected” should have an attribute to signify the distance of the connection. Contrarily, if two entities are not explicitly (physically) connected, FBS (hence QRS) will not consider that their coupling exists. This is not true when physical phenomena act from a distance (e.g., electromagnetic field and heat transfer). Therefore, any entity defined in FBPhPhS must have relations with other entities but such relations should have attributes (typically a geometrical distance) that can be compared with each other. However, this also means that QRS reasons out a huge number of unfeasible behaviors (spurious behaviors) as well.

With the attributive information about relations, QRS is able to prioritize reasoned out phenomena in the order of, for instance, significance or plausibility. An example of how attributes for relation can generate a not expected physical phenomenon is given in Figure 7. In the first figure physical phenomenon is generated by the state of one entity and it permits then the state transition. The same physical phenomenon can act on another entity if they are directly connected. In the second figure the physical phenomenon is generated by the combination of two states of entities. They can also be indirectly connected. Then this physical phenomenon realizes the state transition.

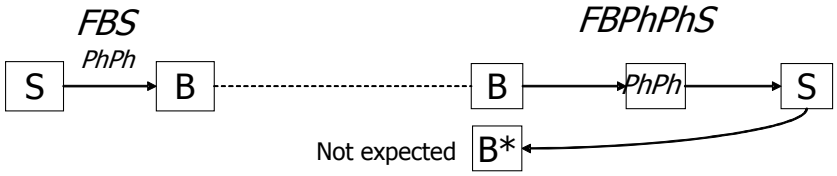

Figure 6. Difference between FBS and FBPhPhS due to Physical phenomena

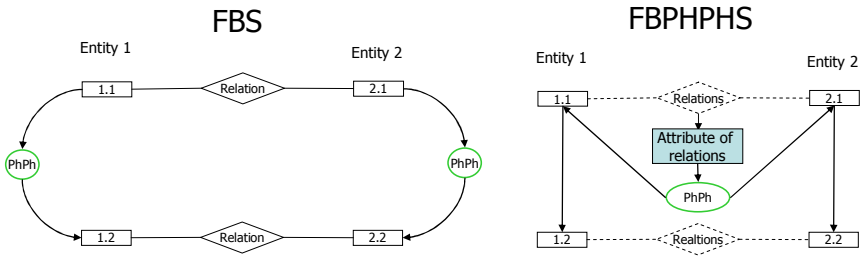

Figure 7. Difference between FBS and FBPhPhS given by attributes for relations

6 QUALITATIVE REASONING SYSTEM

The previous section described the DID model structure (FBPhPhS) and the knowledge representational scheme. Here it is explained better about the reasoning system (QRS) based on QPT.
DID is a tool that aims to work at an early stage of the design where only qualitative information is available.
DID uses qualitative physics based reasoning [10], among others based on Forbus' QPT [14] that envisions possible physical phenomena that can happen to the system.. An early attempt of using QPT for design can be found in [15], [16]. QPT is an appropriate theory to describe dynamical situations and therefore to reason about processes. By envisioning these phenomena, DID also finds interactions among different theories. These interactions signify knowledge level interferences that cause two types of complexity in product development, *viz*., *complexity by design,* disciplines or *intrinsic complexity of multidisciplinarity* [17].
QPT provides a method for encoding knowledge about the physical world and some methods for reasoning with that information. The fundamental knowledge within QPT is a network of entities and physical phenomena that can happen to these entities when certain conditions are satisfied. It captures basic knowledge about physical world that corresponds to commonsense knowledge. In order to achieve consistency in representation, it is important to stick on ontological obligations while collecting the basic knowledge. QPT can represent entities, physical phenomena, attributes, values of attributes and connections of FBPhPhS using its fundamental concepts individuals, processes, quantity conditions and influences. Functions of FBPhPhS is not represented within QPT.

7. FILTERING BEHAVIORS

In principle, FBPhPhS reasons out all possible physical phenomena including both predictable and unpredictable, many of which are even superfluous. For this reason the system needs a filter. QRS employs QPT to encode knowledge about the physical world including a network of entities and physical phenomena that can happen to these entities when certain conditions are satisfied. QRS generates all possible physical phenomena that can happen. Before the inference, the designers should define predictable phenomena, which are usually main functions of the machine and known phenomena. Anything else is by definition an unpredictable problem. However, these unpredictable problems can be prioritized as described below.

7.1 Prioritization of the detected behaviors

Since it is well known that a qualitative physics based reasoning system can reason out numerous, superfluous behaviors that can be neglected, it is crucial for the system to prioritize reasoning results.

For instance, it may reason out that anything in a gravity field receives gravity force in proportion to its mass. It also reasons out that anything that receives force might deform and depending on the material property, the deformation could be, for example, elastic deformation, plastic deformation, or even brittle fracture. The interesting part of this story is that the system even warns with a scenario that there is a possibility that the machine can be destroyed by its weight, which is something unusual. Using a qualitative reasoning system requires to remove such unlikely situations from the reasoning results. This can be done initiating the attributes for relations that were introduced before. Attributes of relations are defined by relations between attributes of entities, for instance distance, gradient of temperature, and difference in velocity and acceleration. With those characteristics the system can select which behaviors should be taken into account related to the required accuracy.

7.2 Introducing quantitative information to DID

Qualitative knowledge is used for reasoning about the physical world. "*The value of reasoning qualitatively resides in the translation from measurements to conceptual understanding. On its turn conceptual understanding is a prerequisite for controlling and troubleshooting a system"* [14]. However there are some situations in which quantitative information is also required.

DID identifies each single phenomenon but it is possible for a number of them to be superfluous. For instance, depending on the desired accuracy, life-cycle of the system, or operation of the device the detected phenomenon can be insignificant. To give an example, heat generated by a specific component can in time damage the behavior of another component in the system, such as a spring by changing its natural length. If the life-time of the spring is shorter than that of the component, or if the elongation has no influence on the final outcome, such information can be neglected. To do so, further information should be introduced into the system in order to avoid superfluous states confusing the engineer with irrelevant conclusions.

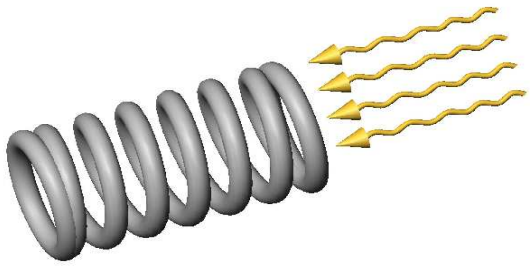

Figure 8. Spring influenced by high temperature

Prioritizing the reasoning results is not sufficient in this pattern, however. Quantitative information should be introduced to avoid ambiguities, or in explicit circumstances of superfluous behavior. This is because introduction of quantitative information might result in loss of general information about the system, which is undesirable from the point of view of evolution of the system. In other words, while using a quantitative reasoning system leads to eliminate ambiguous or irrelevant behavior, the system looses general information. The designer must evaluate the convenience of introducing data in the system.

8 CONCLUSIONS

This paper suggested a method for automatically detecting "unpredictable" problems in the conceptual design of complex machines such as mechatronics machines. In Section 2 some useful definitions are introduced. Using a rotary encoder as an example, Section 3 clarified the meaning of constructive and destructive behaviors (couplings) that result from a combination of desired and undesired, predictable and unpredictable problems. Then, in Section 4, DID is introduced as a tool to identify these unpredictable problems that are mostly resulting from complexity by design and can cause unpredictable behaviors of subsystems. DID first detects all physical behaviors that can eventually appear in a machine and then filters out unpredictable problems. In the succeeding sections more details were given about several aspects of DID.

Future tasks include the development of DID taking FBS as a starting point, including both the reasoning system and an adequate knowledge base. Although the system is yet to be developed, this approach seems feasible and promising.

ACKNOWLEDGEMENT

The authors gratefully acknowledge the support of the Dutch Innovation Oriented Research Program 'Integrated Product Creation and Realization (IOP-IPCR)' of the Dutch Ministry of Economic Affairs.

REFERENCES

[1] Schotborgh W. O. Tragter H., Kokkeler F. G. M., Van Houten F. J. A. M., Tomiyama T. Towards a Generic Model of Smart Synthesis Tools, *CIRP* 2006.

[2] Rault A., Mechatronics and Bond Graphs in the Automotive Industry, ed. By E. T. S. V. Scintilla in Mechatronics, *pioneering or self-evident?* Univeristy of Twente, 1992, pp. 68-77

[3] Roberts G., Intelligent Mechatronics, *Computing & Control Engineering Journal*, 1998

[4] Seo K., Hu J., Fan Z., Goodman E. D., and Rosenberg R.C., Automated Design Approaches for Multi-Domain Dynamic Systems Using Bond Graphs and Genetic Programming, *The international Journal of Computers, Volume 3: Systems and Signals*, n.1, pp.55-70, 2002

[5] Seo K., Hu J., Fan Z., Goodman E. D., and Rosenberg R.C., Towards an Automated Design Method for Multi-Domain Dynamics Systems Using Bond Graphs and Genetic Programming. *Mechatronics, Volume 13*, Issues 8-9, pp. 851-885, 2003

[6] Umeda Y., Ishii M., Yoshioka M., Shimomura Y., Tomiyama T., Supporting conceptual design based on the function-behavior-state modeler, *AI for engineering Design*, Analysis and Manufacturing, Vol. 10, No. 4, 1996, 275-288.

[7] Tomiyama T., D'amelio V. Toward Design Interference Detection to Deal with Complex Design Problems, *CIRP* 2006.

[8] Browing R., Applying the Design Structure Matrix to System Decomposition and Integration Problems: A Reviwe and New Directions, *IEEE transition on Engineering Management, volume 48*, 2001

[9] Nam P. Suh, *The principle of Design*, Oxford Series on Advanced manufacturing, 1990

[10] Forbus KD, Qualitative Process Theory, 1984, *Artif Intelligence* 24, 1984, 85-168

[11] Yoshioka M., Umeda Y., Takeda H., Shimomura Y., Nomaguchi Y., Tomiyama T., Physical concept ontology for the knowledge intensive engineering framework, Volume 18: *Advanced Eng. Informatics*, No. 2, 2004, 69-127.

[12] Kiriyama T., Tomiyama T., Yoshikawa H., Qualitative reasoning in conceptual design with physical features, in: Recent advanced in qualitative physics, Eds. B. Faltings, P. Struss, *The MIT Press, Cambridge, MA*, 1992, 375-386.

[13] Kuipers, Benjamin , "Qualitative Reasoning: modeling and simulation with incomplete", *Artificial Intelligence*, 1994

[14] Forbus K. D., De Kleer J. *Building Problem Solvers*. Artif Intelligence, Cap. 6-11, 1993

[15] Kiriyama T., Tomiyama T., Yoshikawa H., A Model Integration Framework for Cooperative design. In: Sriram D, Logcher R, Fukuda S (eds) *Computer aided cooperative product development*, Lecture notes in computer science 492. Springer-Verlag, Berlin 1992, pp. 126-139

[16] Kiriyama T., Tomiyama T., Yoshikawa H., Qualitative reasoning in Conceptual Design with Physical Features. In: Faltings B, Struss P (eds) *Recent Advanced In Qualitative Physics*. The MIT Press, Cambridge, MA, pp 375-386

[17] Tomiyama T., Knowledge structure and Complexity of Multi-Disciplinary Design, 2006

[18] Kuipers B., Qualitative Reasoning/ Modeling and Simulation with Incomplete Knowledge, p. cm. Artificial Intelligence, 1994

[19] Barr, Avron Cohen, Paul R. , Feigenbaum, Edward A., *The handbook of artificial intelligence*. Vol. 4", Artificial intelligence, 1989

[20] Bobrow, Daniel G, "Qualitative reasoning about physical systems", *Artificial Intelligence: an international journal* volume 24, 1985

[21] Weld D., de Kleer J., Reading in Qualitative Physics, Morgan Kauffman, Palo Alto, *CA*, 1987

Contact: Valentina D'Amelio
Delft University of Technology
Faculty of Mechanical, Maritime and Materials Engineering
Intelligent Mechanical System
Mekelweg 2, 2628 CD Delft
The Netherlands
Tel: +31 (0)15 27 85608
Fax: +31 (0)15 27 84717
Email: v.damelio@tudelft.nl