Enhancing Engineering Design and Analysis Interoperability Part 1: Constrained Objects

Miyako W. Wilson^{1,2}, Russell S. Peak¹, and Robert E. Fulton^{1,2}

Georgia Institute of Technology

¹ Engineering Information Systems Lab 813 Ferst Drive Atlanta, GA 30332-0560 USA miyako.wilson@eislab.gatech.edu russell.peak@eislab.gatech.edu http://eislab.gatech.edu/ ² Woodruff School of Mechanical Engineering 813 Ferst Drive Atlanta, GA 30332-0405 USA robert.fulton@me.gatech.edu http://www.me.gatech.com/

Version: Nov. 10, 2000 Submission for Review

Abstract

The wide variety of design and analysis contexts in engineering practice makes the generalized integration of computer-aided design and engineering (CAD/CAE) a challenging proposition. Transforming a detailed product design into an idealized analysis model can be a time-consuming and complicated process, which typically does not explicitly capture idealization and simplification knowledge. Recent research has introduced the multi-representation architecture (MRA) and analyzable product models (APMs) to bridge the CAD-CAE gap with stepping stone representations that support design-analysis diversity. This paper introduces constrained objects (COBs) as a generalization of the underlying representations.

The COB representation is based on object and constraint graph concepts to gain their modularity and multi-directional capabilities. Object techniques provide a semantically rich way to organize and reuse the complex relations and properties that naturally underlie engineering models. Representing relations as constraints makes COBs flexible because constraints can generally accept any combination of I/O information flows. This multi-directionality enables design sizing and design verification using the same COB-based analysis model. Engineers perform such activities through out the design process, with the former being characteristic of early design stages and vice versa.

This paper presents basic examples to illustrate the main COB concepts. To validate the COB representation, other work describes electronic packaging and aerospace test cases implemented in a toolkit called *XaiTools*TM. In all, the test cases utilize some 260 different types of COBs with some 370 relations, including automated solving using commercial math and finite element analysis tools. Results show that the COB representation gives the MRA a more capable foundation, thus enhancing physical behavior modeling and

knowledge capture for a wide variety of design models, analysis models, and engineering computing environments.

Key Words

constrained object (COB), constraint graph, multi-directional, multi-fidelity, CAD-CAE integration

1 Motivation

While computing tools continue to advance, Wilson [1] identifies the need for a physical behavior modeling representation that supports the following characteristics in a unified manner:

- Has tailoring for design-analysis integration, including support for multi-fidelity idealizations, product-specific analysis templates, and CAD-CAE tool interoperability.
- Supports product information-driven analysis (i.e., supports plugging in detail design objects and idealizing them into a diversity of analysis models).
- Has computer processable lexical forms along with human-friendly graphical forms.
- Represents relations in a non-causal manner (i.e., enables multi-directional combinations of inputs/outputs).
- Captures engineering knowledge in a modular reusable form.

This paper overviews recent work describing the above needs and surveying related topics in the literature. The following sections overview the constrained object (COB) representation that has been developed to address these needs [1, 2]. It illustrates the main concepts with several basic examples. Two other papers in this conference describe COB usage for design-analysis integration [3] and as a step towards multi-functional optimization [4].

2 COB Basics

2.1 COB Modeling Languages and Views

The COB representation includes several modeling languages and views as summarized in Figure 1. The COS and COI languages are the primary lexical forms which are computer interpretable. The other forms depict subsets of COS and COI model content and include graphical views that aid human comprehension. For example, Figure 2 summarizes the notation for the graphical constraint schematic notation, which emphasizes object structure and relations among object attributes.

The structure level languages and views define concepts as templates at the schema level (meta-level), while the instance level defines specific objects that populate one or more of

these templates. The next sections present several of these forms for COBs that represent basic engineering concepts.

2.2 Example: Spring Primitive

The upper left portion of Figure 3 shows the traditional form of an idealized spring object. A figure defines the variables and their idealized geometric context, and algebraic equations define relations among these variables.

The representation of this object as a COB spring template is shown in Figures 3a-3c, where the constraint schematic graphically depicts its relations and variables. Figure 3b is the COS textual form, which is the master template from which the other forms can be derived. Figure 3c is an encapsulated form known as a subsystem, which is useful for representing this object when it is used as a building block in other COBs (e.g., Figure 7).

In all these forms the relations can support any valid input/output combination. For example, in relation r1, attributes length and start can be inputs to produce end as the output, or end and start can be inputs to produce length as the output.

Figure 4 shows views of an instance of this spring entity in two main states. In state 1, spring constant, undeformed length, and force are the inputs, and total elongation is the desired output. The COI lexical form (Figure 4b) shows state 1.0 as this COB instance exists before being solved. State 1.1 shows it after solution (including constraint schematic form in Figure 4a), where one can see that length was also computed as an intermediate value, and that end and start have no value because there were not sufficient inputs to compute them. State 5 shows this same spring instance where the desired deformed length has been changed to be an input, and spring constant is the desired output.

Considering the engineering semantics of the problem, one sees that state 1 typifies a simple design verification scenario where the "natural inputs" (physical design properties and a load) are indeed inputs and a "natural output" (a physical response to the load) is the requested output. Hence, the design is being checked to ensure it gives the desired response. As a design synthesis (sizing) scenario, state 5 reverses the situation by making one natural output into an input and one natural input into the desired output; it effectively asks "what spring constant (a design-oriented variable) do I need to achieve the desired deformed length (a physical response)?" This COB capability to change input and output directions with the same object is a useful engineering capability which can be applied to more complex situations. It is a multi-directional capability in that there are generally many possible input/output combinations for a given constraint graph.

2.3 Example: Spring System

Given a system of two springs like in Figure 5a, with traditional approaches one would consider their free body diagrams, determine their relations and boundary conditions, and solve the resulting system of equations for the desired output. One could use computational math tools like *Mathematica* to aid this process and change input/output combinations. Yet essentially one would have a list of equations whose engineering

meaning would not be inherent in their existence (e.g., one could not query relation r1 and know that it is part of a spring). Furthermore, adding and deleting equations to change input/output directions for a large system of equations could become unwieldy. When one considers the constraint graph (Figure 6b) for this two spring system, one recognizes that the shaded portions are essentially duplications of the same kind of relations (e.g., r11 vs. r21). Traditionally, one would have to manually replicate and adjust these similar relations, which is a potentially tedious and error-prone process. COBs address these issues by grouping relations and variables according to their engineering meaning and placing them into explicit reusable contexts.

For example, by applying object-oriented thinking, the shaded regions in Figure 6b are represented by two spring subsystems in Figure 6a. There is no need to specify these relations in the corresponding COS lexical form (Figure 6c), as they are included in the spring entity per its COS definition (Figure 3). System level boundary conditions are the only other relations that need to be specified here. With this definition completed, the constraint graph can now be seen as another view derivable from the lexical form; it essentially is a fully decomposed constraint schematic where no subsystem encapsulations are present. Figure 7 gives one possible instance of this COB with state 1 being a design verification scenario. *XaiTools*TM is an analysis integration toolkit [1, 2] that implements these concepts directly from the COS and COI forms (Figure 8). It enables links with design tools and effectively provides an object-oriented constraint-based front end to traditional CAE tools, including math tools like *Mathematica* and finite element analysis tools like *Ansys*.

1.4 Example: Extensional & Torsional Rods

 $\Delta L = \frac{FL}{FA} + \alpha \Delta TL$

COBs can be used to represent analytical engineering concepts as analysis building blocks (ABBs) [1, 2]. In Figure 9, an ABB representing 1D linear elastic material behavior is re-used in building two continuum primitives: extensional rod and torsional rod. These in turn can be used to build other analysis models (Table 1), including product-specific contexts [1, 2, 3]. Note that traditional relations like Equation 1 (for total deformation in an extensional rod) are not explicitly given in Figure 9. Rather, these types of relations are derivable from the fundamental relations present in the COB and its subsystem(s); thus, their effects are automatically included.

Equation 1

3 Industrial Examples and Other Test Cases

Industrial applications of COBs and other test cases are given in [1, 2], and their COB statistics are summarized in Table 1. These include thermomechanical analysis for printed wiring boards and assemblies (PWA/Bs), structural analysis for airframes, and thermal analysis of electrical chip packages.

4 Summary

This paper introduces constrained objects (COBs) as a new representation of engineering concepts that has these overall characteristics:

- Declarative knowledge representation (non-causal)
- Combination of objects and constraint graph techniques
- COBs \cong (STEP EXPRESS¹ subset) + (constraint graph concepts and views).

Test cases show that COBs provide these advantages over traditional analysis representations:

- Greater solution control
- Richer semantics
- Capture of reusable knowledge

Envisioned extensions include capturing assumptions and limitations so that analysis results might be automatically verified to some degree.

Acknowledgements

This work builds on analyzable product models (APMs) by Diego Tamburini [2].

References²

- 1. Wilson MW. *The Constrained Object Representation for Engineering Analysis Integration*. Masters Thesis, Georgia Institute of Technology, Atlanta, May 2000.
- 2. Peak RS. X-Analysis Integration (XAI)³ Technology. Georgia Tech Engineering Information Systems Lab Technical Report EL002-2000A, March 2000.
- 3. Peak RS and Wilson MW. Enhancing Engineering Design and Analysis Interoperability - Part 2: A Multi-Fidelity Example. First MIT Conference on Computational Fluid and Structural Mechanics. Cambridge MA, 2001.
- 4. Peak RS and Dreisbach RL. Enhancing Engineering Design and Analysis Interoperability - Part 3: Steps toward Multi-Functional Optimization. First MIT Conference on Computational Fluid and Structural Mechanics, Cambridge MA, 2001.

Figure

See MS PowerPoint 97 file.

¹ STEP EXPRESS [ISO 10303-11] is an object-flavored information modeling standard geared towards the life cycle design and engineering aspects of a product. For further information, see http://www.nist.gov/sc4/.

² Some of these references are available at *http://eislab.gatech.edu/*.

 $^{^{3}}$ X = design, manufacture, sustainment, etc.

Tables

						Attributes		Relations			
		Structure (COS)		COB Libraries Used	Entities	Total	Aggregate	Total	Oneway	Aggregate Operation	Aggregate Instance
general(lib)	geometry.cos				4	11		3			
	abbs.cos				108	68		30			
	apm.cos			lib\geometry.cos	12	34		22			
	materials cos			3	9		1				
product specific	link	apm.cos		lib\apm.cos lib\materials.cos	1	11		10			
	flap	cbams.cos		apm.cos	5	25		36	2		
	pwa/b	apm.cos		Physician	77	152	8	19		9	
		cbams.cos		apm.cos	5	21		23	3		5
	airplane		abbs.cos	lib\apm.cos	24	39		12	3		
			cbams.cos	lib\geometry.cos lib\apm.cos airplane\lib\abbs.cos	2	7		16			
			fastener.cos		3	7					
		lib	materials.cos		1	38					
			apm.cos	lib\geometry.cos lib\apm.cos airplane\lib\materials.cos airplane\lib\fastener.cos airplane\lib\cbams.cos	4	23		20			
		bikeframe	cbams.cos	airplane\bikeframe\apm.cos	2			20			
	electrical chip package (cp	lib	pwb_board.cos		13	21	2	5			
			apm.cos	lib\geometry.cos cp\lib\pwb_board.cos lib\abbs.cos	53	177	6	103		3	22
		bga (ball grid array)	cbams.cos	cp\bga\apm.cos lib\geometry.cos	1	12	4	19			15
		afp(augd flat pack)	apm.cos	lib\abbs.cos	25	/6	1	18			15
Totals	Ψ	yip(yuau iiat pack)	CDallis.COS	cp/qit/apiii.cos	344	753	25	376	8	12	59

Table 1 - COS structure statistics⁴ for COB test cases [1]

⁴ Not all COBs in abbs.cos are fully developed. Many exist as placeholders for future work. Approximately onefifth of the COBs are fully usable, thus a more accurate COB entity count in abb.cos would be ~25 vs. the 108 shown. This change gives the 260 total COB entities identified in the abstract.









Wilson et al Enhancing Engineering Design and Analysis Part 1								
Figure 4 - Multi-Directional (Non-Causal) Capabilities of a COB Instance: Spring Primitive								
a. Constraint Schematic-I	b. Lexical COB Instance (COI)							
$\begin{array}{c c} \hline \textbf{Design Verification} & \textbf{example 1, state 1.1} \\ \hline \textbf{SNmm} & \textbf{Spring constant}, k & \hline \textbf{T} = k\Delta L & force, F \\ \hline $	<pre>state 1.0 (unsolved): state 1.1 (solved): INSTANCE_OF spring; undeformed_length : 20.0; spring_constant : 5.0; total_elongation : ?; force : 10.0; END_INSTANCE; state 1.1 (solved): INSTANCE_OF spring; undeformed_length : 20.0; start : ?; end : ?; length : 22.0; total_elongation force : 10.0; END_INSTANCE;</pre>							
Design Synthesisexample 1, state 5.120 N/mm $spring constant, k$ $r3$ force, F40 N20 mm $undeformed length, L_0$ $x_1 = L - L_u$ length, L22 mm10 mm $start, x_1$ $L = x_2 - x_1$ x_2 $x_1 = x_2 - x_1$ 32 mm end, x_2 $r1$ $r1$	<pre>state 5.0 (unsolved): state 5.1 (solved): INSTANCE_OF spring; undeformed_length : 20.0; spring_constant : ?; start : 10.0; length : 22.0; force : 40.0; END_INSTANCE; State 5.1 (solved): UNSTANCE_OF spring; undeformed_length : 20.0; start : 10.0; length : 22.0; force : 40.0; END_INSTANCE; State 5.1 (solved): UNSTANCE_OF spring; undeformed_length : 20.0; start : 10.0; length : 22.0; force : 40.0; END_INSTANCE; State 5.1 (solved): UNSTANCE_OF spring; undeformed_length : 20.0; start : 10.0; length : 22.0; force : 40.0; END_INSTANCE; State 5.1 (solved): UNSTANCE_OF spring; undeformed_length : 20.0; start : 10.0; length : 22.0; force : 40.0; END_INSTANCE; State 5.1 (solved): State 5.1 (solved)</pre>							
	5							









