Available online at www.sciencedirect.com
ScienceDirect
Advanced Engineering Informatics 22 (2008) 71–89
ADVANCED ENGINEERING INFORMATICS
www.elsevier.com/locate/aei

Constructing design representations using a sortal approach

Rudi Stouffs *

Faculty of Architecture, Delft University of Technology, P.O. Box 5043, 2600 GA Delft, The Netherlands

Received 29 November 2006; received in revised form 5 July 2007; accepted 17 August 2007

Abstract

Supporting the early phases of design requires, among others, support for the specification and use of multiple and evolving representations, and for the exchange of information between these representations. We consider a complex adaptive system as a model for the development of design representations, and present a semi-constructive algebraic formalism for design representations, termed *sorts*, as a candidate for supporting this approach. We analyze *sorts* with respect to the requirements of a complex adaptive system and compare it to other representational formalisms that consider a constructive approach to representations. We demonstrate the advantages of *sorts* in various examples, illustrate its use to support the specification of design queries and the recognition of emergent information, and consider *sorts* in relationship to integrated product models.
© 2007 Elsevier Ltd. All rights reserved.

Keywords: Design representations; Design communication; Complex adaptive system; Design queries; Emergence

1. Introduction

Building design is a multi-disciplinary process, involving participants, knowledge and information from various domains. Building design problems, therefore, require a multiplicity of views, each distinguished by particular interests and emphases. Each actor in the design process takes his or her own professionally oriented view—derived from an understanding of current problem solution techniques in the respective domain. Each view, in turn, requires a different representation of the same (abstract) entity; a building may be considered in its entirety, as a shape, a collection of parts or some grouping of properties. As such, different views—or different representations—may derive from different design stages, and may also support different persons or applications within the same design stage. Even within the same task, or by the same person, various representations may serve different purposes defined within the problem context and the selected approach. This is certainly true in architecture, where the design process, by its exploratory and dynamic nature, invites a variety of approaches and representations (see, for example, [1]). Any man-machine system to aid the designer must recognize his reliance on multiple representations [2].

There has been concerted effort in developing integrated product models that span multiple disciplines, multiple methodologies and support different views (e.g., [3]). Various modeling schemes for defining product models and ontologies also exist (e.g., [4,5]). These allow for the development of representations in support of different disciplines or methodologies, and enable information exchange between representations and collaboration across disciplines. Such efforts tend to characterize an a priori top-down approach: an attempt is made at establishing an agreement on concepts and relationships, which offer a complete and uniform description of the project data, mainly independent of any project specifics [6]. We are concerned how data can be effectively structured a posteriori. Research in cognitive science and design cognition has shown that expertise in both problem solving and design often relates to having access to more and better representations [7,8]. This is especially true in architecture, Akin refers to architecture in this respect as a “representation

* Tel.: +31 15 278 1295; fax: +31 15 278 4178.
E-mail address: r.m.f.stouffs@tudelft.nl

saturated problem domain" [9]. Importantly, the outcome of the design process relates to the representation that is used. Furthermore, architectural design differs from other forms of problem solving in that the problems in architecture are generally ill-structured; defining the problem space is an intricate part of the design activity [8]. As the problem shifts during the design process, so should the representation adapt. Design representations may be as much an outcome of as a means to the design process.

To clarify our aims we present an example. Within the conceptual phase of design, architects and designers study relevant precedents and collect and look at design and other documents as sources of knowledge and inspiration [10]. Generally, electronic design document libraries or image archives serve to collect this information, separately from the CAD or modeling environment that is used to shape the design. In this paper, we present a formalism for constructing design representations that can assist in specifying and maintaining relationships between design-related documents and the elements within a CAD model. Specifying such relationships helps to organize the information contained within these documents in relation to the CAD model. Consider a representational structure that reflects on (part of) the CAD model, for example composed of element IDs and descriptions. A corresponding data construct can easily be generated, automatically, from the CAD data. This representational structure can then be extended to allow for document references to be associated with the CAD elements. Using a graphical interface, the user can specify both the references and their associations to CAD elements. When the CAD model is subsequently changed, the data reflecting on the CAD model can be regenerated, while the associated data can be retrieved from the original representational structure using an automatic conversion based on the matching of both representational structures. The formalism here considered supports such matching of representational structures, by means of a subsumption relationship over representational structures and a behavioral specification for data constructs. Merging both data constructs re-associates the document references to the CAD elements, on condition that the respective element IDs have not changed.

Let us denote the representational structures constructed by means of this formalism as *sortal* representations. Consider the construction of sortal representations as compositions of other sortal representations through the use of two compositional operators, an attribute operator specifying an object–attribute relationship between two sortal representations and an addition operator specifying a disjunctive relationship between sortal representations. For instance, the representational structure that reflects on part of the CAD model can be expressed as a composition of element IDs and element descriptions under the attribute relationship (denoted '$\wedge$'):

$$element\_ids \wedge element\_descriptions$$

The user can associate document references to CAD elements through a sortal representation that is similarly composed of element IDs and document references using the attribute relationship:

$$element\_ids \wedge document\_references$$

Both sortal structures can subsequently be added, yielding the following sortal representation ('$\wedge$' distributes over '$+$'):

$$element\_ids \wedge (element\_descriptions + document\_references)$$

The resulting sortal structure can be matched back to the sortal representation *element_ids* $\wedge$ *document_references* in order to extract the associations from the CAD-derived data. These can be updated, if necessary, through the user interface. At the same time, the CAD-derived data can be regenerated from the CAD model. Repeating the sortal addition of both structures yields once again the entire sortal structure, relating design documents to CAD elements. Fig. 1 offers a schematic overview of the entire process.

A similar application can be considered for associating cost data to CAD elements and calculating production or construction costs. A corresponding user interface can allow the user to associate component or material types

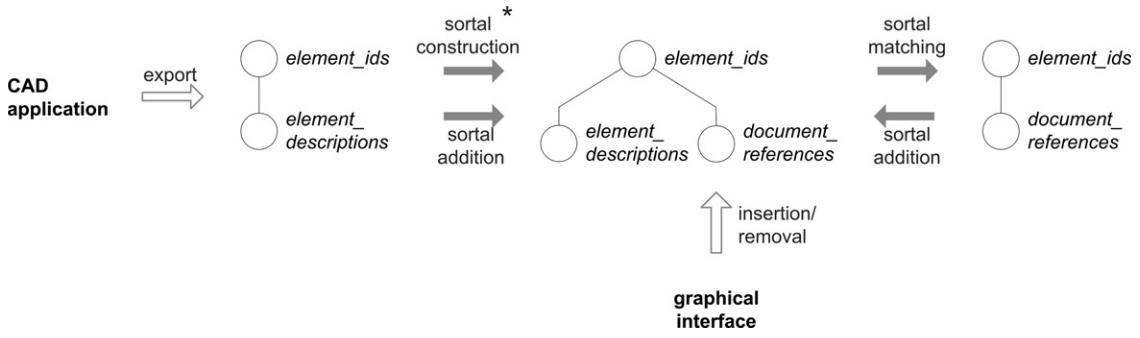

Fig. 1. Schematic overview of the process of relating design documents to CAD elements using *sorts*. Filled arrows denote automatic sortal conversions; * sortal construction applies only once, initially.