StructuralComponents – a software system for conceptual structural design

Bastiaan van de WEERD[1], Anke ROLVINK[1] and Jeroen COENDERS[1]

[1] BEMNext Laboratory, Faculty of Civil Engineering and Geosciences, Delft University of Technology and Arup Amsterdam, the Netherlands, *b.m.vandeweerd@tudelft.nl*, *a.rolvink@tudelft.nl*, *j.l.coenders@tudelft.nl*

Summary

Conceptual design is the starting point of the design process. The conceptual design stage comprises the formation of several ideas or design concepts to meet the imposed constraints. StructuralComponents is a software application that attempts to provide the designing engineer with a suitable set of tools for this stage. It facilitates the exploration of the early design space with the proper balance between accuracy and the ability to explore.

Using parametric-associative modelling, a structural design concept can be composed of parameters and object relations. Built-in design and analysis knowledge returns immediate output to guide the design process. The major development is a redevelopment of the software architecture as a client-server system in support of emphasising design exploration and a system of design plurality and directed optimisation.

Keywords: *Structural engineering; computation; early-stage design tools; parametric-associative modelling; real-time analysis; design argumentation and justification; client-server architecture.*

1. Introduction

The main goal of StructuralComponents is to capture a record of the conceptual design process in structural engineering. It is a software tool that uses parametric modelling to let the user define the logic of a structural design. It simultaneously analyses the performance of the resulting structure to give immediate feedback. Lastly, it allows its strictly-defined design logic to be presented in a number of distinct ways. It is a software tool that accommodates the early-stage design process through the distinct phases of creation, analysis, and presentation. As such, StructuralComponents is a software tool that tries to augment and complement the intelligence and creativity of the engineer.

The development of StructuralComponents has been driven by the perceived unsuitability of available engineering software to the conceptual design stage. After an investigation of the structural design process, it has been concluded that the tool needs to adhere more closely to the characteristic early-stage design cycle. To this end, further emphasis is needed on alternatives generation and exploration. Goals have been set for partly-automated alternatives generation, faster

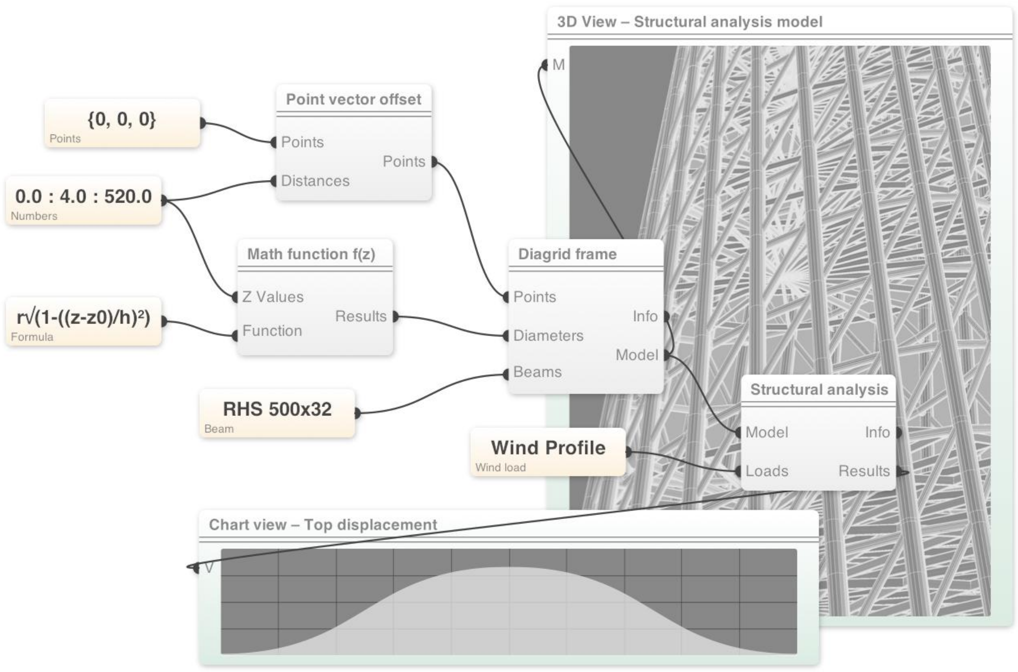

Figure 1: Parametric tool for conceptual structural design stages

and thus more immediate evaluation, and capturing these through richer documentation. The implemented software application attempts to address these goals.

1.1 Background

The general concept of StructuralComponents has been described extensively in earlier work of the authors [1]. Moreover, many of its concepts are linked to the work of Coenders [2]. Consequently, this paper will cover many of these topics briefly, instead focussing on the latest developments and how they relate back to the starting principles. These developments are the increased emphasis on the synthesis and generation of design alternatives, and the client-server software and open data architecture overhaul.

1.2 Conceptual design and tools

Prompting the above research is the perception of the authors of the insufficient suitability of the structural engineering software in common use to the early conceptual design stage. This stage, more than those following later, requires creativity and generation under limited constraints, concepts with which computers have relative difficulty. In exploring possible design solutions, this stage requires only low precision, but high flexibility. Later stages determine the performance of a specific solution, while this stage seeks a solution for a desired performance.

Design will be understood in the sense of Systems design, as the process of trying to find the best solution satisfying a problem. A solution will be a set of decisions or choices. A problem will be a set of requirements or constraints that can be satisfied by multiple solutions, of which satisfaction is easily verifiable. Importantly, some solution will have greater net value beyond the point of satisfaction than others. A set of decisions will be the selected choices to a given set of options. As decisions are made, the scope of subsequent decisions will be limited while the initial set of requirements will expand.

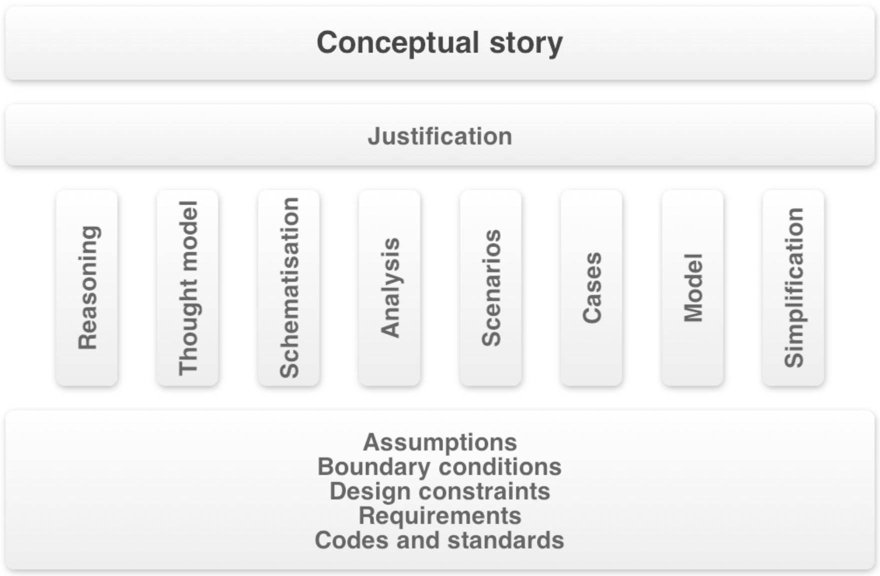

Figure 2: Conceptual story and design justification in StructuralComponents

Understood in terms of structural engineering, decisions relate to concepts such as structural layout, topology and member sizing. The initial set of requirements cover concepts such as loads, architectural shape or floor area ratio. Further requirements will be introduced during the design process, such as material stresses or geometry clashes. The abstract design problem in structural engineering falls in the category of problems for which no known efficient way to find a solution exists [3], although the complex and creative human decision-making process has proven very valuable.

StructuralComponents revolves around the concept of the design story. This is the overarching narrative of a design solution on a high level, covering all the major design decisions but leaving many of the low-level details to be resolved in later stages. Of the story of a design solution that satisfies a problem it can be assumed that its decisions will minimally clash with detail decisions later on in the process, such that early decisions leave enough room for sensible detail solutions. The narrative of the design story will be based on a range of assumptions, logic, calculations and experiments that will be referred to as the design justification. Forming the design story and its supporting justification completes de conceptual design stage.

Conceptual design tools aim to support the designing engineer in forming the story. Any design tool shouldn't hope to actually design by itself, but rather to aid the designer precisely in the qualities that might promote his success. In order to do so, these tools must be highly interactive and yield immediate and obvious insights and precise to a certain degree. They must provide the engineer with a confidence and control.

1.3 Formal structural design process

Design is described as the process of seeking the best set of decisions to solve a set of requirements. This process is iterative, with the same overall theme recurring repeatedly for smaller sub-problems. Such an occurrence is understood as a design cycle, which consists of the interwoven phases of proposal, testing and modification [4]. That is, proposal or generation of a solution, its testing or evaluation in view of the problem, and the modification of failing assumptions. As design iterations progress, the emphasis of the cycles shifts from generation initially to evaluation eventually.

Various character characteristics of the expert design process have been found. Broadly speaking, these characteristics involve taking time to understand the problem and frequent testing, creating alternative concepts using both top-down and opportunistic decision-making, considering components' interrelations and reviewing and reflection in terms of requirement [5]. Most of these characteristics assume a high degree of repetition, reinforcing iteration as a key aspect of design.

Towards successfully tackling the design process, a number of qualities and tactics have been found [6] to promote success in engineering design. These include—in addition to solid knowledge of the field—spatial imagination and routine sketching. Successful design often summarises without suppressing solution ideas, in order to explore the possibilities. Alternatives are sought for isolated requirements, will fully disregarding others, only to merge new discoveries later on. Most successful design involves frequent evaluation and balanced solutions reduction.

The various design constraints will either be a base aspect of the problem or a function of multiple aspects. Various aspects are often highly interdependent as they typically share resources in some form. The design process is assumed to converge to a single design that satisfies its interdependencies [7], though is impeded by the individual interests of its aspects.

2. Implementation

StructuralComponents is the software application that implements the aforementioned concepts. It makes use of parametric-associative modelling to define design logic and structural finite element analysis for design evaluation. Modelling happens on a client application, separated from the server application that handles model execution and analysis, both of which run on custom engines. For communication between the two, the server application provides a RESTful web service API.

Parametric-associative modelling in StructuralComponents largely follows the paradigm seen in GenerativeComponents [8] and Grasshopper [9]. Two previous versions were developed as plugins to these applications, respectively. The latest version introduces a custom engine that supports—in addition to parameters, components and connections between them—encapsulation of models into a single component in order to increase parametric legibility. The standard library includes mathematical, geometrical and structural components and parameters, from numbers and vectors to points and curves to beams and loads.

The latest version of the package includes a new finite-element structural analysis engine which is based on the Cholesky decomposition algorithm. Only static non-linear analysis has been implemented at this time. Element types include axial and bending elements, both one and two-dimensional. Nodes can be constrained on lines or planes, both rigidly and elastically. After assembly, stiffness matrices are solved using the CHOLMOD library [10].

Client applications are free to access and represent the design and performance data in any way. Implementations have been developed for iOS [11], and HTML in web browsers. The first allows both viewing and editing using an interactive graph-like interface, while the latter only allows viewing. Both implementations use the SVG-based D3.js framework [12] for data visualisation.

3. Emphasised Synthesis

One of the major areas of focus for recent-most development and an important reason behind the move towards a standalone package was to increase the emphasis on design generation, or synthesis. This emphasis is implemented foremost using a method of design plurality where the system generates and evaluates slight variations along with the decisions of the user. Building on that is a system of specification by bandwidth as opposed to exact values.

Design plurality refers to the implemented system of automatically varying user-specified parameters, every resulting combination of which constitutes an alternative design that is additionally evaluated. The client application then displays the results in a way that instantly illustrates the effect of adjusting the varied parameter up or down. Consequently, the user can decide whether making the adjustment would be worthwhile. Due to the overall speed of the system the user will quickly grasp parameter sensitivity and continuously make adjustments, constituting a form of user-guided optimisation.

Building on the system of design plurality is a system of design by bandwidth. Instead of specifying a parameter's exact value, the user can specify it as lying within a certain bandwidth. Bandwidth is represented by a probability distribution function, which can be uniform but also a bell curve. The result is a design where each derived parameter evaluates to a distribution function of the preceding bandwidth-specified parameters. The bandwidth-specified parameters can be set so that each value within the distribution function derived parameter satisfies design requirements. It is then left to a later stage to specify the parameters exactly.

4. Software Architecture

In its current implementation, StructuralComponents is a web application. That is, it is an application that runs on server-side machines and has only its input and output data accessed by clients. It defines the format in which its input data must be submitted, and from which output data can be interpreted. In this way, the application is client-agnostic, leaving the exact ways in which its input data is formed and output data is used open.

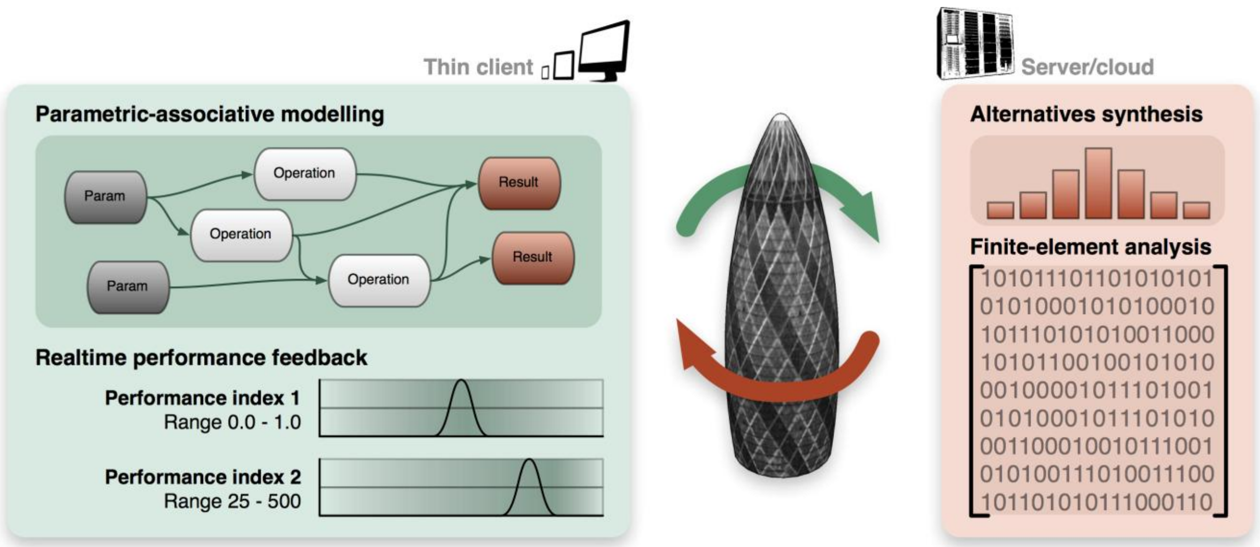

Figure 3: Overview of the client-server architecture

This client-server architecture has many benefits. Execution of critical code is confined to the server, while the client-side code is only involved in creating and interpreting input and output data. Foremost, this separates demanding processor-bound code and storage from the lightweight (likely graphical) user interface. The server application can be modified at any time without involving users, making it possible to continuously update or scale up and down according to demand. Features can be added to the data format specification or deprecated in a balanced manner. Client applications can be developed as needed and always remain compatible.

The StructuralComponents client application is largely written for UNIX in C, for Linux or OS X. A number of temporary development shortcuts currently confine it to the latter, though mostly in the networking code. The application will eventually hook into Apache as a common handler [13]. Clients have been developed for iOS, in Cocoa and Objective-C—targeting the iPad—and the web, in HTML—targeting any modern browser.

Communication uses an application programming interface over HTTP, designed in the REST architectural style [14]. REST stands for Representational State Transfer, which was developed as an abstraction of the HTML-based architecture of the web. It is based on uniquely identified resources, which are represented in a certain way and manipulated through these representations without any assumption about server state other than what the server explicitly reveals.

StructuralComponents specifies its data format in both the JSON and XML languages. These formats are the representations of its resources, such as model definitions, components, and parameters, among others, which are all uniquely identified. Clients may request a representation of a list of resources, and upon receival request any of their specific representation. It may then locally make a modification to the representation and send it back as such, upon which the server may carry out the modification to the actual resource.

As implemented, StructuralComponents uses the power of heavy server-side machines for computation-intensive tasks while leaving room for the thin client to be designed around the user experience. Its network architecture and open data structure allows access from anywhere, manipulating or presenting data in any manner. An interactive modelling client may access the data for manipulation, while for example a documentation client may only access parts of the data and represent them in a natural language. In short, it provides universal access to design logic.

5. Conclusion and discussion

The described system attempts to provide a digital toolset suitable for the early-stage structural design process. It models design decisions and logic through the objects and relations of a parametric model. While it leaves the design decisions to the designing engineer, it uses powerful computational methods to give context to those decisions. It shows parameter sensitivity and alternative design paths that may not be immediately obvious.

The authors propose that digital tools for the conceptual design stage should principally strive to aid the engineer in exploration of the design space and documentation of the design path. Towards the former, it has been the underlying assumption of this research that parametric-associative modelling is a suitable approach. It has subsequently attempted to enhance the approach with the flexible yet powerful client-server software architecture, enabling the described concepts of design plurality and design by bandwidth.

The authors regard the same parametric-associative modelling approach as a suitable tool to documentation the design path. The described client-server architecture has been an attempt to define a data model that captures design. Its approach is to centrally define the set of decisions that define a design solution in a well-defined data structure and make it accessible for clients to derive an interpretation suitable to the individual context.

While the authors acknowledge that no tool to aid design can be definitive, StructuralComponents shows a number of concepts that they believe to be instrumental in the digitally-enhanced approach.

6. Recommendations

The implementation remains a collaborative effort between the BEMNext laboratory and Arup and its development will be continued academically and within the engineering practice. It is not immediately usable in its current state, as the focus has been on developing a conceptual prototype. Various aspects essential to any software package are completely absent, such as data persistence, security and a proper authentication and authorisation model.

As a conceptual tool, the number and variety of design objects currently available in the toolbox is limited. Enough have been added to test the concepts, but more will be required to put the system into practical use.

The data structure and specification are completely proprietary to the system, as no suitable existing alternative was found. Going forward, it will be necessary to engage in the standardisation of formats describing relational structural design.

While an attempt has been made to increase the user-friendliness of the system, it still requires a certain degree of advanced knowledge of the underlying software. While additional effort can alleviate these requirements to an extent, it will ultimately be necessary for the structural engineer to acquire some knowledge of these technologies and practices

7. References

[1] Rolvink A, van de Weerd B, and Coenders J, “StructuralComponents”, *Proceedings of the IASS-IABSE symposium,* Taller, Longer, Lighter, 2011, London

[2] Coenders J, *NetworkedDesign: next generation infrastructure for computational design,* PhD thesis, Delft University of Technology, Delft, 2011

[3] Chapman WL, Rozenblit J, Bahill AT, *“The system design problem is NP-complete”,* International conference on robotics, 1994

[4] Smith RP, and Tjandra P, *“Experimental observation of iteration in engineering design”,* University of Washington, Seattle, Research in Engineering Design 10, no. 2, 1998

[5] Sims-Knight JE, Upchurch RL, and Fortier P, “A simulation task to assess students’ design process skills”, *35th ASEE/IEEE Frontiers in Education Conference,* University of Massachusetts, 2005, pp. F4G1-F4G6

[6] Fricke G, *Succesful individual approaches in design engineering”,* Research in engineering design, vol. 8, pp. 151-165

[7] Klein M, Samaya H, et al. *“The dynamics of collaborative design: Insights from complex systems and negotiation research”,* Massachusetts Institute of Technology, Complex Engineered Systems, Springer, 2006

[8] Bentley, GenerativeComponents, http://www.bentley.com, 2012

[9] McNeel, Grasshopper, http://www.grasshopper3d.com, 2012

[10] Davis TA, Hager WW, "Dynamic supernodes in sparse Cholesky update/downdate and triangular solves", ACM Trans. Math. Software, vol. 35, no. 4, 2009

[11] Apple, iOS, http://www.apple.com/ios/, 2012

[12] Bostock M, D3, http://mbostock.github.com/d3/, 2012

[13] Apache, Apache HTTP server, http://httpd.apache.org/, 2012

[14] Fielding RT, "Architectural styles and the Design of Networked-based Software Architectures", University of California, Irvine, 2000