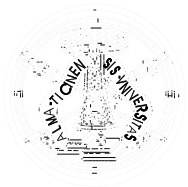
Istituto Universitario
di Studi Superiori



Università degli
Studi di Pavia

EUROPEAN SCHOOL FOR ADVANCED STUDIES IN
REDUCTION OF SEISMIC RISK

ROSE SCHOOL

THE EFFECTIVE USE OF COMPUTERS IN ENGINEERING DESIGN

A Dissertation Submitted in Partial
Fulfilment of the Requirements for the Master Degree in

EARTHQUAKE ENGINEERING

by

DAVID DE KONING

Supervisor: Dr RUI PINHO

August, 2007

The dissertation entitled “The Effective Use of Computers in Engineering Design”, by David de Koning, has been approved in partial fulfilment of the requirements for the Master Degree in Earthquake Engineering.

Rui Pinho _____

Nigel Priestley _____

ABSTRACT

Computers are currently used in all engineering design offices, but have not fundamentally changed the way structures are designed. This dissertation investigates the attributes of good design and introduces ideas and concepts from computer science that can help engineers use computers as effective design aids. These are combined in an example of a suspension bridge design tool, which highlights the shift in decision-making from the design engineer to the software developer.

Keywords: computer-aided design; engineering design; model-view-controller; inheritance; suspension bridge

ACKNOWLEDGEMENTS

Thanks to: Dr. Priestley and Dr. Pinho for their enthusiastic backing of my somewhat esoteric research proposals; my parents for their constant encouragement and for not charging me rent while I was writing; Don Nicol for providing a quiet place to read and research; and most of all, my fiancée Elizabeth for patiently bearing with my slow progress and motivating me finish.

TABLE OF CONTENTS

LIST OF FIGURES

Page

LIST OF TABLES

Page

1. INTRODUCTION

"Old ideas can sometimes use new buildings. New ideas must use old buildings." [Jacobs, 1961] Jane Jacobs, the celebrated urban planner, explains that established businesses with a proven track record, can afford to take the risks associated with expensive new buildings. Start-up companies, with new, exciting and unproven ideas and processes, must inhabit older buildings, where the cost of operating and the cost of failure are lower.

To better understand how computers have been (or should be) adopted by civil engineers, consider a slightly modified version of the above quote: "Old design methods can sometimes use new technology. New design methods must use old technology."

When a design firm has a design method in place, they may invest in a new technology to improve that process, confident that the method will work and that the investment will pay off. On the other hand, a new idea, a new way of organizing the work or a new way of doing design will generally be tested alongside the proven method. If it is better, it will gradually be adopted as the technology allows.

The question of computerization is complex because computers on the one hand are a new technology (an expensive tool) and on the other hand are so radically different from any tool used previously that they open up a wide range of new design methods and processes.

When computers first appeared, their status as new technology dominated their adoption. Existing design methods received incremental improvements, and the expense of computerization was recovered. As John Voeller says "computers have changed the landscape of tools and capabilities available to engineers, yet they have also allowed many old ideas to be preserved without rethinking basic principles fully." [Voeller] Or take Sutherland's words: "The gain in our capacity to check quickly is wholly desirable, but it is less clear that

the thinkning of the designer is advancing as fast as the tools he uses." [1999] However, 'buildings' age and computers age in two significant ways. In the first place, the initial expense of computers is paid off and their ongoing expense is accepted as normal. Secondly, engineers become familiar with the computers and their own peculiar terminology and concepts.

This dissertation returns to the basic principles of good design and introduces ideas from computer science that can be used to build new design tools from the basic principles.

In Section 2, Key Attributes of Good Design", the important attributes of good design are explained. Since I have no design experience, I have drawn on the writings of other engineers.

Section 3, "Evaluating Design Tools", restates these attributes as a series of questions that will help evaluate the effectiveness of particular design tools.

Section 4, "New Thought Patterns", introduces concepts and ideas from computer science and states how they might contribute to the attributes of Section 1.

Section 5, "Sketch of a New Design Tool", applies the principles of Section 2 and the ideas of Section 4 to the problem of suspension bridge design. This is an illustrative example intended to give the ideas of the preceeding sections a more concrete form.

What shape computer-aided design will take is an open question. The aim of this dissertation is to help computers grow old, by making their concepts familiar. The proper place of computers in engineering design is a complex issue and the discussion will continue for quite some time.

2. KEY ASPECTS OF GOOD DESIGN

The question of what exactly engineering design is, is one of the least tractable of all the engineers' problems. Every author has his own definition of 'design' and his own approach to understanding the process. The common thread is that a good design process puts the designer's thoughts in order. Four attributes that contribute to an organized thought process are a strong background knowledge, relevance & focus, transparency, and simple & flexible analyses.

2.1 Strong Background Knowledge

Often referred to as "experience", a good designer is very familiar with the components of his work. Stephenson and Callander state that design "requires a wide range of knowledge and skills." [1974] Englekirk calls this a "fundamental knowledge base". Engineering design must be carried out by trained, experienced designers. This base is composed of "a basic understanding of engineering principles and system behaviour". [Englekirk, 1994] Roos describes this with the following list: "technical knowledge, judgement, intuition, experience". [1966] With a well-developed knowledge base, a designer can then proceed by simply guessing at solutions, which Cross suggests as one of the simplest and best design processes [1935].

He goes on to say that the "faculty of guessing at solutions is capable of great development. One of the chief values of formal analyses is to aid in its development." [1935] Paradoxically, he also states that analyses can't be properly understood without this background knowledge: "The interpretation of stress analysis makes absolutely necessary a clear idea of the action of the structure up to the stage at which rupture is conceivable." [1953b] Perhaps, it is best to start with Stephenson's more empirical suggestion: "To see things bend and break and observe their fracture is more revealing then days of lectures or reams of text." [1974]

A good designer will understand how a structure behaves when loaded on both the component and the system level. Since no one is born with the structural sense already developed, a good designer will use a method that builds up his structural sense as he works.

2.2 Relevance and Focus

A successful designer will know what to consider and what to ignore, and will keep the irrelevant details from distracting him. Roos' perspective on the design process almost entirely in these terms: "An engineer makes decisions based on the relevant problem information. As such, an engineer may be considered a highly complex information processor who is faced with the problem of effectively and efficiently acquiring, organizing, reducing, evaluation and updating data and information." [1966] Englekirk points out the necessity of "a systematic approach to problem solving", that "focus is essential to problem solving." and that "all too often , designers utilize overly zealous analytical efforts and this usually causes them to lose sight of the objective" [1994]

The ideas of focus and strong background knowledge are connected by Shedd & Vawter, when they discuss analysis: "the successful designer must have so thorough an understanding of the fundamental principles of structural analysis and such facility in their application that he makes these essential computations as a matter of course, leaving his mind free to focus on those aspects of his design in which technical skill...are all important." [1941]

Though this is not emphasised in engineering schools, it makes the difference between a quick, clear design and a complex design, slowed by all the irrelevant details.

2.3 Transparency

A design must be transparent with respect to several parts of the process: assumptions and analysis procedures; analysis verification and communication.

Transparency means that a designer will know what he is doing. Cross reminds us that "judicious engineers decline to use theoretical formulas whose derivation they have no time to unravel." [1935] In 1916, Strassner wrote that "the methods of calculation based on simple geometrical designs...open the way to a natural, perspicuous method of survey" [1964]. Englekirk bemoans the lack of transparency in design codes, which "give the appearance of precise technology" and hide "many of the key assumptions upon which the relation[s are]

based." [1994] van Rooij and Homburg note that greater transparency, among other things, "very much improves the quality of the design." [2002]

Transparency also facilitates clarity in communication. As Stephenson and Callander say: "Although some of this work can be a mental exercise, it is important that the engineer should be able to record his ideas and decisions, or pass them on to others, clearly and unambiguously." [1974] With regard to checking analysis results, Cross says that it "is highly desirable that the solution of a problem in stress analysis be so stated that anyone familiar with the subject may see whether the forces balance throughout the structure and whether the deformations are consistent with continuity in the structure." [1935]

Transparency gives a designer a greater confidence in his work and allows his peers and colleagues to better verify and implement the design.

2.4 Simple and Flexible Analysis

While computers have opened wide the ranges of available analyses, flexibility and simplicity remain the most important criteria. Complexity does not imply power.

"Simplicity is more important than speed." [Cross, 1935] A good design method is simply "a clearer, more definite, more flexible restatement and correlation of fundamental procedures", and is "simple in its steps" [Cross, 1935] Straub refers to Strassner's method for analysis of non-articulated vaults as "an extremely simple statical calculation." [Straub, 1964]

Cross says that "it is desirable—one would feel tempted to say that it is necessary, if we were not so far from realizing it—that the method of analysis shall be flexible." [1935] By this he means that material uncertainties and variability are accounted for and the effect of non-uniformities can be easily estimated. He mentions that in some cases "it seems quite impossible to fit even approximately the method of analysis to the actual structure." [1935]

Now that computers are widely used, simple and flexible analyses will be seen only when simplicity and flexibility are explicitly and deliberately sought.

All four of theses aspects—strong background knowledge, relevance & focus, transparency and simple & flexible analyses—relate to how a designer thinks. Above all, a good design method helps the designer think clearly.

3. EVALUATING DESIGN TOOLS

A good designer is marked by a strong background knowledge and clear focus; by a transparent design process; by flexible and simple analyses. It is then highly desirable to evaluate design tools (software products, in many cases) in terms of how they take advantage of the designer's skill and create a fruitful working environment. The following questions, organized under each heading, should help designers think through such an evaluation.

3.1 Background Knowledge

- is the tool more useful to senior or junior engineers?
- does it help build your structural sense?
- does it allow a review of the design process once it's over? is this a simple task?
- are you able to satisfy yourself that the results are reasonable independently of the tool's output?

3.2 Focus & Relevance

- how much time is spent learning how to use the tool?
- what proportion of time is spent interacting with the tool, as opposed to advancing the design?
- is a final design produced more quickly?
- is the final design improved?
- does the tool give you greater or lesser confidence in the final design?
- what information is absolutely required? do you consider this information to be relevant?
- does it hide and show different information for different people?
- what information is emphasised?

- how is the information organized?
- does it required the designer to change the way he organizes his thoughts?
- how much space and time is devoted to the design (versus dealing with the tool)?

3.3 Transparency

- do you have any idea what the tool actually does?
- are assumptions listed?
- are assumptions presented clearly with analysis results?
- can the design data be accessed without the tool?
- can you check the analysis results visually?
- can someone unfamiliar with this particular design check it quickly?
- is useful information buried in a flood of data?

3.4 Flexibility and Simplicity

- can uncertainty be considered?
- how much information is required for the tool to work?
- how much definite data does it take to set up a model?
- can you change what the tool shows you?
- can you create a new presentation of the tool's output?
- does it increase the maximum complexity of systems that can be considered?
- does your design process need to be changed to accommodate the tool?

4. NEW THOUGHT PATTERNS

"What convenience there is in the procedure is inherent not in the method itself but in the mode of thought of the engineer who is to use it." [Cross, 1935] If a good design method must help designers think clearly, computer-aided design must primarily be an aid to clear thinking. What new modes of thought does the computer contribute to achieving this? A full exploitation of the computer's potential requires a rethinking of the organization of engineering problems.

The following ideas will give fresh understanding to problems and procedures that engineers have known for ages.

4.1 Abstraction

The Oxford English Dictionary defines abstraction as "the process of considering something independently of its associations, attributes or concrete accompaniments." John Locke described abstraction as separating ideas "from all other ideas that accompany them in their real existence". [John Locke, An Essay Concerning Human Understanding (1690)]

At its heart, much computer work is a grand and complex exercise in abstraction. Although a computer is made of up transistors and wires, programmers never explicitly control a specific, physical device. The functions that the computer present are abstracted, first to machine code, then assembly language, then to the more complex languages such as C, Java or Lisp. When writing a program, much of the work is devoted to abstracting the tasks that the program will perform and abstracting the information it will use.

Though humans have been developing and using abstractions for millennia, the field of computer science has formalized and thoroughly investigated the concept of abstraction. The first two chapters of "Structure and Interpretation of Computer Programs", a landmark

computer science textbook, are devoted to the topic of building abstractions [Abelson and Sussman].

Abstractions are critical in good design. An engineer's background knowledge is made up of many abstractions that allow him to recognize and organized the available data. Abstractions allows a designer to consider only the relevant information, helping keep focus. This also increases the transparency, since the abstractions organize the design, allowing others to understand and check it more quickly. They also simplify a design, allowing more complex structures to be designed.

The idea of abstraction is already well established in the minds of civil engineers. Many abstractions are so familiar to us that we no longer consider them as such. The idea of strain hides all the complexity of inter-molecular forces; beam theory reduces a three-dimensional structure to one dimension; calculating a line of thrust through an arch allows an analyst to ignore the mixture of compressive forces and bending moments in a curved beam; the linear and linear-perfectly plastic material models are familiar even to first-year students; in earthquake engineering, the response spectrum reduces the dynamic behaviour of complex structures to a simple graph.

Another set of abstractions are analysis procedures. Hardy Cross' Moment-Distribution can be applied to a wide range of moment-frames [1932]; the Pelikan-Esslinger method proscribes a set of calculations for the dimensioning of orthotropic steel decks [1963]; Nigel Priestley's Direct Displacement-Based Design is a more modern example [2003]. In each case, the method focuses on a particular aspect of design (moment-resisting frames, orthotropic steel bridge decks and dynamic behaviour, respectively) and sets out a step-by-step process for determining a safe design.

With computers, these two types of abstractions are recognized formally (that is to say, the abstractions are abstracted). Abstractions of a procedure or process give rise to algorithms and programs, while abstractions of information yield data structures. All the above abstractions can be easily stored in a computer. The design methods may require human interaction, but all the rote computation can be automated.

The power that computers give to abstractions is a double-edged sword. It allows more complex abstractions to be used: non-linear, plastic material behaviour and time-history

analyses, for example. It can also abstract the analysis completely away from the designer. Many commercial software packages hide the data and procedural abstractions, leaving the engineer to enter data in the appropriate format and then interpret the output as best he/she can. Simplicity and focus have been obtained at the expense of flexibility and of the engineer's background knowledge.

For an engineer to remain in control of the design process, he must retain control over his abstractions. Computers have placed a wide array of powerful abstractions at his disposal, so powerful that unless he fights to stay in control, the design process itself will be abstracted away from the traditional design firm. In some sense, the programmers are today's design engineers.

Some firms have responded by developing in-house software, such as Black & Veatch's PowrTrak and Buckland and Taylor's SABER and CAMIL [Increasing the Load Capacity of Suspension Bridges J. Bridge Engrg., Volume 8, Issue 5, pp. 288-296 (September/October 2003)]. In an effort to encourage this kind of work, the following sections present some abstraction from computer science. Hopefully, as engineers become more familiar with computers, they will write their own programs more often: retaining control of the abstractions they use and therefore the design process.

4.2 Model-View-Controller

The model-view-controller (MVC) abstraction is very basic, relating to how information is stored, viewed and modified. It stands in sharp contrast to pre-computer methods of storing information.

As has been mentioned, the great advantage of computers is that we can store information abstractly, devoid of context and, for practical purposes, devoid of physical limitations. Information can be transferred halfway around the world without moving anything more than a few electrons. In our amazement, it is easy to forget how unnatural this is. We are used to physical representations of information: ink on paper, a plastic spedometer or the red line of a thermometer. We do not deal with raw information, we deal with pictures of information.

The MVC pattern breaks down how we store, observe and change information. The model is the organization of the raw data. What data should be stored? What information is relevant

to the problem at hand? What data structures are best suited for their storage? How we observe and interpret the data is not relevant to the data model.

The view takes the data from the model and presents it for observation and interpretation. A view could be a drawing, a report or a summary of analysis results. No information is stored in the view itself–all the view does is specify how the data in the model is presented.

The controller allows the data model to be modified. If we only have a model and some views, short of modifying the data directly, the data cannot change. The controller watches for modifications to a view and updates the model. Once the model is updated, the changes will be reflected in the view.

To understand the distinctions, consider the case of an engineering drawing. A model is the CAD file stored on a computer and a view is the plot. The engineer can then mark corrections and changes on the paper and the draftsman, functioning as the controller, can take the marked up paper (the view) and update the model. The view is then updated by printing a new sheet and discarding the old one.

The MVC pattern can also be applied to understand how the draftsman interacts with the CAD software. Here the model is still the raw data sitting on the disk, but the view consists of the drawing and the controls that are presented on the screen. The controller is the part of the software that interprets the draftsman's commands and modifies the data on the disk accordingly.

In a paper-based drafting process, there is no distinction between the model and the view: the data storage and the data representation are the same thing. Computer-aided drafting is a situation where new technology is in place, but is used to implement old thought patterns. Despite the presence of a separate model and view, we are not very far from the traditional paper workflow. While revisions are much easier and the drawings can be transferred electronically, they are still merely drawings. The data model is a drawing model, not a structural model. Instead of storing the data in structural abstractions (such as beams, arches, etc...), drawing abstractions (lines, arcs & shapes) are used.

A more advanced data model can store structural elements as structural elements, and a drawing view could generate plots, not from a model of a drawing, but from the model of a

structure. When the model represents the object under consideration directly, all the relevant views can then be generated: drawings, analyses and their results, material sheets, etc...

This type of thinking has been put into practice by at least one engineering firm. Black & Veatch has developed a system based on "data-centered thinking", a term coined by John Voeller synonymous with the MVC pattern. [Voeller, 1996]

The difficulty in the adoption of the MVC pattern is not technological but psychological. Engineers think in terms of pictures: free body diagrams and building elevations. The first step to adoption is to find a data model that is adequate to represent structures. Here the balance between simplicity and the opportunity for creative thought must be kept in mind. This is no easy task. It is likely that a different data model is required for every different structural task. Defining a simple, flexible and comprehensive data model will be one of the primary challenges facing engineers who want to put computers to work for them.

4.3 Prototyping and Inheritance

These two ideas are related to the hierarchy of abstractions. Prototyping occurs when a data structure is designated as a prototype and then copied as needed. The new copy can then be modified without affecting the original. For instance, if a developer is building a subdivision of identical houses (which does occur, sadly), then a single design can be reproduced, and each house modified slightly to account for the differences in ground condition or the buyer's preferences.

Inheritance is a similar concept, but involves types of objects, instead of the objects themselves. Drawings and reports, though very different, both have creation dates, modification histories and authors. These common attributes are associated with documents, and drawings and reports are types of documents. Drawings and reports are special types of documents and therefore inherit the attributes associated with a document. Note that we are discussing presence of certain attributes, not their values. A drawing, by virtue of being a document, has all the attributes of a document, but the values of those attributes are unique to that drawing. The benefit of inheritance is that when you want to deal with all the documents in one place, you can consider all the documents as mere documents and ignore all the differences between the different document types.

Both of these concepts are better explained in other places. Any basic textbook of object-oriented programming will cover inheritance in great detail and most references of the ECMAScript language will discuss prototyping.

The idea of inheritance and prototypes give structure to abstractions by defining the relationship between similar abstractions. Recognizing this may help senior engineers to better present their accumulated knowledge to junior staff. Organizing like this can also improve consistency across a large and complex project.

At first glance, it may seem that a judicious application of these ideas will result in less work for the designer. This is not necessarily the case and the main advantage is in organizing the designer's thoughts, helping him keep his focus on the relevant information.

5. SKETCH OF A NEW DESIGN TOOL

To illustrate all these ideas, this section presents a sketch of a software tool that puts these ideas into practice. It helps with the design of suspension bridges, and demonstrates how a simple tool can help the engineer understand and direct the design process. We will start by applying the ideas of section 3, then we will evaluate the tool based on the key aspects of design described in section 1.

5.1 A New Idea

Let us start by breaking down the design process along the lines of the Model-View-Controller. We need to decide how to model a suspension bridge and what information the model requires. We then will present different views of the data, different ways of looking at a design. Finally, we will present some interface elements for modifying the design.

5.2 Data Model

The data model is a statement of what information is relevant for design. For the purpose of this exercise, a suspension bridge is modelled by the coupled differential equations of the main cable and the deck. This model makes several reasonable assumptions: that the vertical hangers are inextensible and closely spaced, that the main cables follow a parabolic profile, and that the full dead load is carried entirely by the cables. [Gauvreau, 2002]

$$E_G I_G v^{IV} - (H_D + H_L) v^{II} = W_L - \frac{H_L W_D}{H_D} \tag{5.1}$$

$$\frac{H_L l_c}{A_C E_C} - \frac{W_D}{H_D} \int v dx = -\frac{2H_L}{k} \tag{5.2}$$

The information required to solve these equations are as follows:

Table 5.1. Data Required for Analysis

W_L	live load
W_D	dead load
H_L	horizontal component of cable force due to live load
H_D	horizontal component of cable force due to dead load
E_GI_G	bending stiffness of the deck
E_CA_C	longitudinal stiffness of the main cable
k	effective lateral stiffness of the top of the tower
l_c	Initial arc length of the cable

We will also store the results:

Table 5.2. Analysis Results

V	live load deflection of the deck
M_G	moment in the deck
V_G	shear in the deck
R	reactions at the abutments and the tops of the towers

In addition to these, the cable geometry is required to find H_D and H_L. These pieces of information comprise a complete data model. The deck is represented by a single number: it's stiffness. The towers are represented by the location and lateral stiffness of their tips. The main cables are represented by a parabola and a longitudinal stiffness. The loads are stored as a set of uniform and point loads.

We also need information to decide the bridge location and geometry. We need a bedrock profile, to decide where to place the towers and approaches. To decide the deck geometry, we also need to know water levels and the required shipping clearance.

Though it is not strictly necessary at this point, we will also consider the deck geometry, that is the elevation profile of the deck. This is not considered in the analysis, but it is such a basic attribute of the bridge that it is best to include it from the very beginning. Note this inclusion

is based on background knowledge: we decide what is relevant and we give the design process a focus from the moment we start choosing a data model.

Once this data is in place, it is not difficult to solve for the live load deflections of the girder. A variety of numerical methods can be applied and this problem does not concern us here.

It is helpful to organize the data model in terms of design, not analysis. The data model should not reflect the information required for the particular analysis, but rather the decisions that the engineer makes. The values required for analysis can be computed from the decisions. The following table shows which analysis inputs depend on which design choices.

Table 5.3. Relations between Analysis and Design Data

W_L	live load
W_D	size of cables, deck design
H_L	geometry, live load
H_D	geometry, dead load
$E_G I_G$	deck design
$E_C A_C$	size of cables
k	tower design, cable geometry, dead load

This gives our final data model:

Table 5.4. Data Model

Geometry Constraints:	bedrock profile, water level, required shipping clearance
Geometry:	tower position, tower height, span/sag ratio of cables, deck midpoint elevation*, abutment elevation, abutment positions
Cables	main cable size
Deck	bending stiffness, dead weight
Towers	effective lateral stiffness

* we can constrain two of the tower height, span/sag ratio of the cables and the deck midpoint elevation

It is worth noting in passing that all this applies to a single design alternative. To be truly useful, we require multiple copies of this data model (one per design alternative). This

implies a data model for design alternatives and views for considering the alternatives together.

5.3 Views

The next step is to determine how to present this data. A good representation will increase the transparency and help the designer see how to modify the design.

5.3.1 *Elevation - Geometry*

The first representation that comes to mind is a simple drawing of the bridge. Since the basic model only considers vertical loading and displacement, an elevation is the simplest way to have an overview.

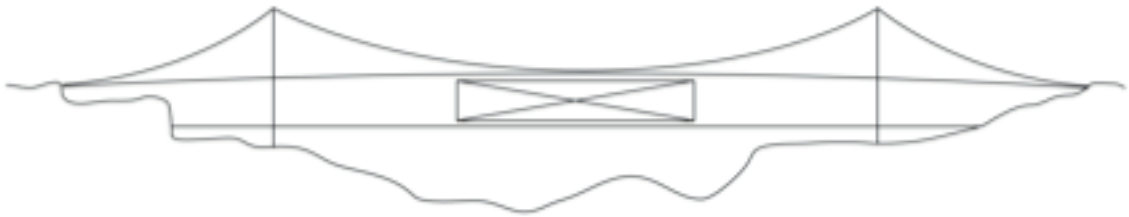


Figure 5.1. Bridge Geometry

Note that not all the data is displayed—this view focuses our attention on the geometry of the bridge. As well, all the data required to generate the drawing is taken from the data model. The towers and the deck are represented by thin lines, since no more detailed information is available. This drawing exposes the geometrical assumptions of the analytical model as well: the drawing shows the parabola assumed in the analysis, even though it is not explicitly stored. Also, even though we only specify the top point of the tower, the drawing shows the whole tower, allowing us to evaluate the proportions of the structure by eye.

5.3.2 *Analytical Model*

Figure 5.2 shows the model used in the analysis. The main concern is to remove all extraneous details, and focus on the only data being used in the analysis.

Figure 5.2. Analytical Model

5.3.3 *Material Estimate*

From the data model, we can also give a rough estimate of the materials required. In this example, we are limited to an estimate for the main cables. When detailed models of the deck, towers and suspenders have been added, this additional data can also be fed into the material estimate.

5.3.4 *Loading*

A simple view that summarizes the loads applied to the bridge is very important for engineers to have a clear understanding of how the model is working. In Figure 5.3, a simplified sketch of the bridge gives a reference point for where the loads are applied. The dead loads are shown explicitly, even though they may be calculated from the material data. The live and dead loads are separated by the deck, but are shown to the same scale.

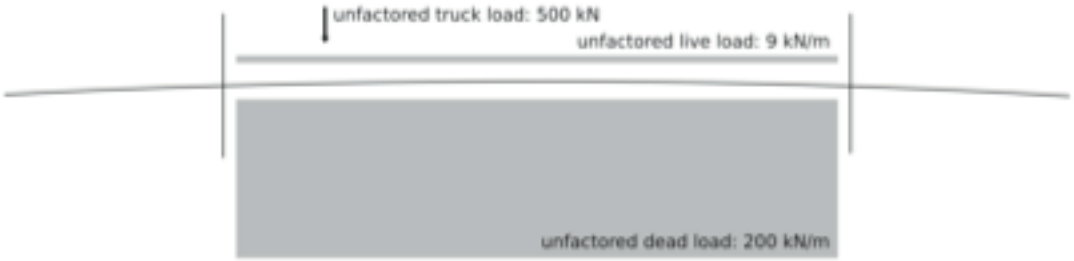


Figure 5.3. Bridge Loading

5.3.5 *Behaviour*

Besides the geometry and the loading, we would like to see the behaviour of the structure under load. We can simply label key points of the bridge—such as the midspan of the deck and the tops of the towers—with the deflection at those points.

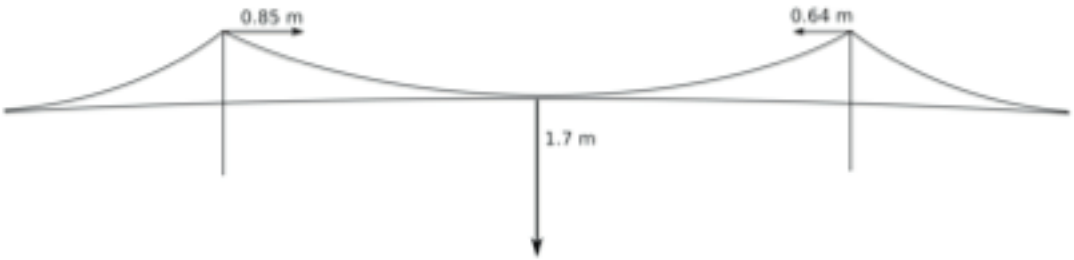


Figure 5.4. Bridge Deflections

Figure 5.5 shows the output from an existing computer analysis of a suspension bridge [Inány].

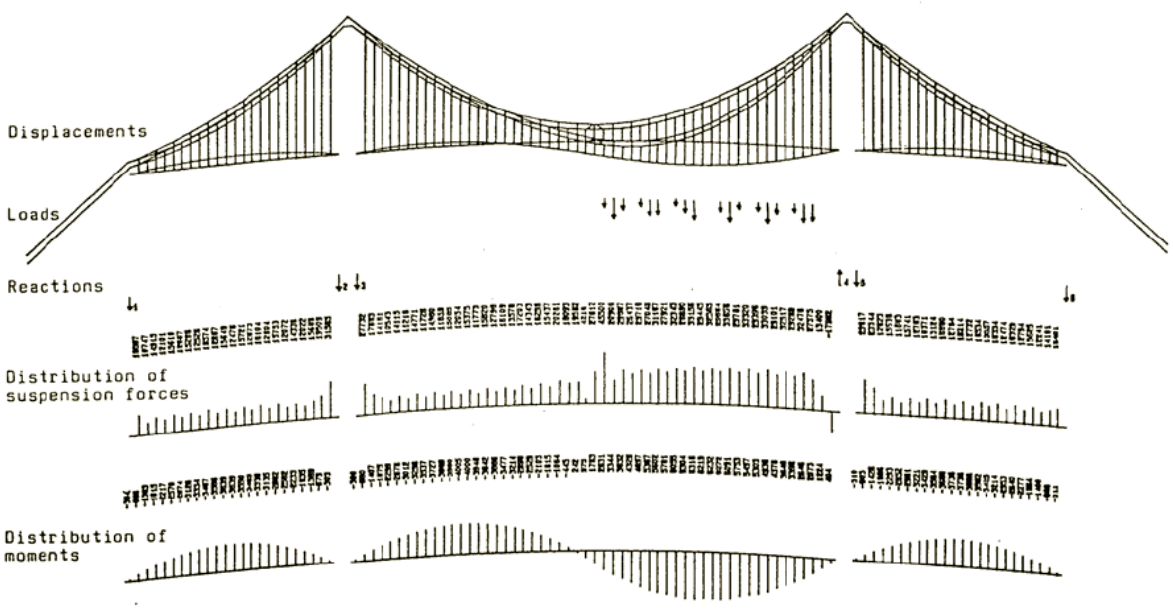


Figure 5.5. Analysis Results from an existing computer program

5.3.6 *Overview*

So far, each other these views have considered one bridge with one load case. As mentioned above, we would like to represent several design alternatives together? The tool must have a view neatly summarizing each one.

This can be accomplished with a printed page that combines all (or some) of the above views. For considering and comparing large amounts of data, the high resolution of the printed page (generally three or four times greater than that of a screen) allows more information to be presented. The flexibility of having several pages, which can be looked at one at a time or spread out side by side, also presents a great advantage over a computer's screen.

5.4 Control Views

In addition to thinking about what information we store and how we present it, we have to consider how to modify it. Here we demonstrate some additional views that allow the designer to modify the data model.

5.4.1 *Bridge Geometry*

The bridge geometry can be changed by clicking and dragging various control points, or by direct modification of the design data. Figure 5.6 shows some control points that can be moved around to easily find a geometry that is compatible with the site.

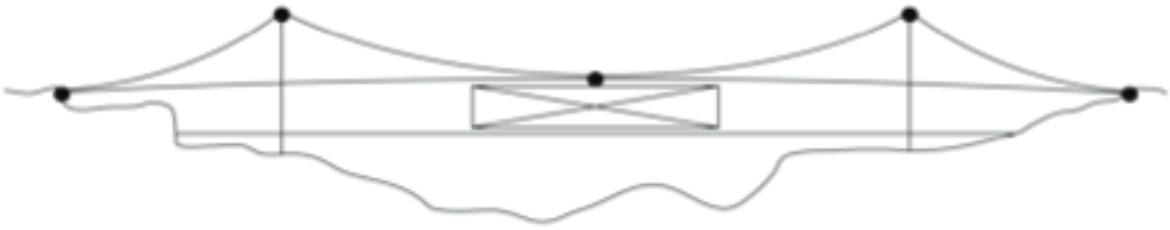


Figure 5.6. Control Points

Since a designer may want to specify the bridge's dimensions exactly, the following view may also be helpful.

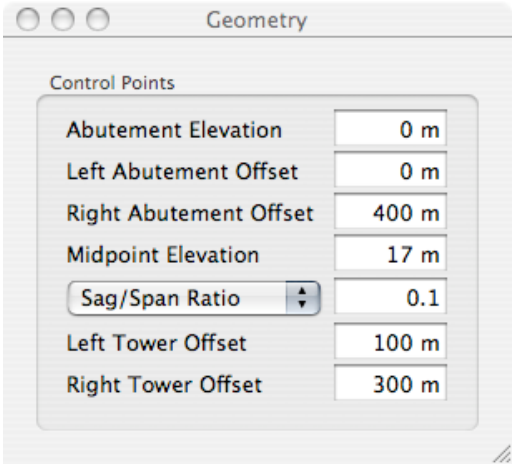


Figure 5.7. Geometry Control

5.4.2 *Member Properties*

The following view, Figure 5.8, allows the important section properties of the bridge to be specified. The design engineer can perform whatever kind of calculation or approximation he wants to produce these numbers.

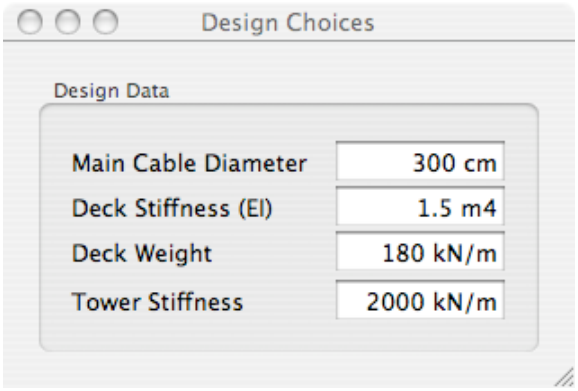


Figure 5.8. Design Choices Control

5.5 Specific, Detailed Data Models

As the design proceeds, data models, views and controllers will need to be developed for the deck, the towers, the anchorages and the rest of the structure. The overview model is important as a focal point for organizing the design engineer's efforts and dictating what features of the detailed models have an impact on the structure's global behaviour. These detailed models can be linked to the overall model by supplying the required data for the analysis (ECAC, EGIG, k, etc...). The detailed models may also be linked to more sophisticated models, such as a full FEM analysis of the completed design.

5.6 Evaluating the Example

Now we will consider how this example helps with the key aspects of design.

5.6.1 *Strong Background Knowledge*

This tool assumes very little about the final design of the bridge. The design process and the final design still depend largely on the designer engineer's background knowledge. For junior engineers, the overview model and the more detailed models provide a structure for understanding the bridges behaviour and organizing all the details of the design. However, engineers whose background knowledge is organized along different lines will find this tool extremely frustrating.

5.6.2 *Relevant and Focus*

This tool provides a focal point for the design—the basic model—and dictates what is and is not relevant to the global behaviour. The most important thing to note is not what this particular example states is relevant, but that the relevance is decided by the designer of the tool. Very often, this is someone other than the design engineer.

5.6.3 *Transparency*

Explicitly defining the data model is fundamental to a transparent tool. The design engineer must know what data is being considered in the analysis. As well, organizing the data well keeps important information from being hidden in a sea of numbers.

The views must also be clear, informative and present nothing more than the data and model assumptions. A view's purpose is not to impress anyone, but to represent the data model in a way that helps the engineer arrive at a final design.

5.6.4 *Simplicity & Flexibility*

Although some significant calculation is required to solve this model, the model itself is simple and easy to understand. Since the design parameters are few and can be easily modified, their effect on the overall model can be quickly ascertained. The design engineer also has to flexibility (in theory) to choose what detailed models, if any, will be used.

5.7 Who Designs the Design Process?

Hardy Cross and Robert Englekirk both insist that engineers must take the initiative in applying a rational design method. However, from this little example, we can see that many decisions about the design process—what is relevant, how transparent it is, the level of simplicity—are made by the software developer, not by the design engineer. To enjoy the full benefits of computerization without compromising their control of the design process, engineers must either write their own software or work closely with the software developers.

6. CONCLUSION

A good design process requires the designer to have a strong background knowledge, keep focus and decide what is relevant, is transparent, and should use a simple and flexible analysis method. Design aids should be evaluated based on whether or not they help in these areas. Computers do not change these fundamental aspects, but do provide the opportunity to return to the fundamentals of civil engineering and rethink the structures we build and the procedures we employ to design and build them. With the computerization of design tasks, some of the responsibility for achieving a good design process shifts from the design engineers to the software developers. Engineers who wish to fully exploit the computer's potential in design must be actively involved in the design of software aids.

REFERENCES

ACI [1963] *Design manual for orthotropic steel plate deck bridges,* American Institute of Steel Construction, New York.

Abelson, H., Sussman, G.J. [1996] *Structure and Interpretation of Computer Programs, 2nd Edition*, The M.I.T. Press, Cambridge, Massachusetts

Cross, H. [1932] "Analysis of Continuous Frames by Distributing Fixed-End Moments," *Proc. A.S.C.E.*, May.

Cross, H. [1935] "Limitations and Application of Structural Analysis," *Engineering News Record*, October 17-24.

Englekirk, R. [1994] *Steel Structures: Controlling Behaviour Through Design*, John Wiley & Sons, Inc., New York.

Gauveau, P., [2002], course notes from *CIV356: Infrastructure Design*, University of Toronto, Department of Civil Engineering

Iványi, M., *Historic Danube Bridges in Budapest*, Budapest University of Technology and Economics, Hungary

Jacob, J., [1961] *The Death and Life of Great American Cities*, Random House, New York.

Locke, J., [1690] *An Essay Concerning Human Understanding*

Priestley, N., [2003] *Myths and Fallacies in Earthquake Engineering, Revisited*, IUSS Press, Pavia, Italy

Roos, D. [1966] *ICES System Design*, The M.I.T. Press, Cambridge, Massachusetts.

Shedd, T.C., Vawter, J. [1941] *Theory of Simple Structures*, John Wiley & Sons, Inc., New York.

Stephenson, J., Callander, R.A. [1974] *Engineering Design*, John Wiley & Sons Australasia Pty Ltd, Sydney.

Sutherland, R.J.M. [1999] "The birth of stress: a historical review" in *Structural and Civil Engineering Design* ed. Addis, W., Ashgate Variorum, Aldershot.

Straub, H. (trans. Rockwell, E.) [1964] *A History of Civil Engineering: An Outline from Ancient to Modern Times*, The M.I.T. Press, Cambridge, Massachusetts.

van Rooij, A., Homburg, E. [2002] *Building the Plant: A history of engineering contracting in the Netherlands*, Walburg Pers, Eindhoven.

References

Voeller, J, [1996] "Editorial: Data-Centered Thinking" *J. Comp. in Civ. Engrg.*, Volume 10, Issue 1, pp. 1-2.