Artificial Intelligence for Engineering Design, Analysis and Manufacturing (2002), *16*, 203–218. Printed in the USA.

DOI: 10.1017.S0890060402163062

Structural monitoring using engineer–computer interaction

RUTH STALKER[1] AND IAN F.C. SMITH[2]

[1]Computing Department, Lancaster University, Lancaster LA1 4YR, United Kingdom
[2]Institute of Structural Engineering and Mechanics, Swiss Federal Institute of Technology (EPFL), Lausanne, Switzerland

(RECEIVED September 10, 2001; ACCEPTED April 12, 2002)

Abstract

Engineer–computer interaction (ECI) is a new subdomain of human–computer interaction that is specifically tailored to engineers' needs. ECI uses an information classification schema, provides a modular approach to task decomposition, and integrates standard engineering characteristics and working procedures into software. A software tool kit that interprets monitoring data taken from bridges was developed according to ECI guidelines. This tool kit was given to engineers for testing and evaluation. An empirical evaluation using questionnaires was performed. The results show that this ECI software corresponds to engineers' needs and the ECI approach has potential applications to other engineering tasks.

Keywords: Structural Monitoring; Engineer–Computer Interaction; Modular Approach; Task Decomposition

1. INTRODUCTION

The first application of information technology (IT) to structural engineering was an analysis program for plane frames, which was proposed in 1956 at Manchester University (Manchester, UK). This became the starting point for much research into the use of computers for structural engineering because it illustrated the utility of IT for analyzing large structures (Grierson, 1996). Although computers are ubiquitous in structural engineering, engineers remain frustrated with the inadequacy of computer support. For example, 15 commercially available software statistical packages were reviewed by Burr (1999) with the conclusion that none were suitable for engineers. This was because they did not offer the functions that engineers need to perform certain tasks.

Engineers must perform tasks using incomplete knowledge, problem-specific characteristics, and context dependency (Salvaneschi et al., 1996). Thus, tasks are difficult to model completely, although there have been attempts (Fenves, 1989). Instead of modeling everything (including engineering expertise), a more practical approach is to enable engineers to interact with the computer to add or delete information as desired in order to make software calculations more compatible with reality. Engineers are legally responsible for their decisions; therefore, they need software that provides realistic solutions and in which they have confidence (Smith, 1996). Thus, the interaction between the computer and user becomes just as important as the algorithms (Wegner, 1997). Even though human–computer interaction (HCI) is a growing domain and its importance has been recognized in structural engineering, it remains a secondary consideration (Anumba, 1994). Reciprocally, HCI does not address engineers as a specific group of users with their own particular needs.

In this article, engineer–computer interaction (ECI) and its application to the domain of structural monitoring are described. ECI is a subdomain of HCI that is tailored particularly to the needs of structural engineers. It supports the design and development of software for engineers as described in Section 2. A description of the testing and evaluation of ECI, using software developed for structural monitoring and diagnosis, is given in section 3. Section 4 contains the results of evaluating the software by questionnaire, which show that engineers are provided with more appropriate decision support. Finally, conclusions and future work are given in Section 5.

Reprint requests to: Dr. Ruth Stalker, Computing Department, Faculty of Applied Sciences, Engineering Building, Lancaster University, Lancaster, LA1 4YR, UK. E-mail: ruth@comp.lancs.ac.uk

2. ECI Analysis

ECI is defined in Stalker (2000) as "a sub-domain of human–computer interaction for the design, evaluation and implementation of interactive decision-support systems for engineering tasks." It is composed of the following three aspects:

1. Organizational schema: A function, behavior, and structure schema of engineering information that represents important stages in a structure's life cycle.
2. Task decomposition (TD): ECI TD identifies subtasks that have been specifically chosen to incorporate iteration, multiple solutions, comparison, and viewpoints into the information transformations. These transformations occur during the tasks that are identified in the organizational schema.
3. Engineer identikit: TD is supported by a generic representation of engineers. This representation enables easy assembly of a graphical user interface (GUI) through implementation of appropriate features.

2.1. Organizational schema

Gero (1990) proposed a schema in order to consider a designed artifact in terms of function (the semantics of a design), behavior, and structure (the syntax of a design). Here the schema is augmented in order to represent temporal aspects. The subscripts t, 0, and t^0 indicate time before the artifact physically exists. Bridges are used as artifact examples in this paper. Thus, the subscripts t, 0, and t^0 indicate time before the bridge exists physically. This augmented schema is shown in grey in Figure 1. Tasks such as design, analysis, formulation, synthesis, and construction transform information from one category to another. In ECI the schema is extended in order to represent the whole life cycle of a bridge and tasks such as monitoring, model correction, intervention, and dismantling are added. These tasks use the subscript t^n to indicates the many iterations the bridge design may have gone through before it is built (extensions are shown in black in Fig. 1).

Function F_0 is a set that refers to structural requirements and reflects objectives such as strength, serviceability, security, and durability of structures. Such objectives cannot be directly transformed into a set representing structural description possibilities (S_{t^0}) without first anticipating desired or expected behavior. Therefore, functional objectives are formulated in terms of expected behavior (B_0). The task of synthesis uses the expected behavior to provide a set of structural descriptions (S_{t^0}). Thus, many structural descriptions may be formulated and iteratively refined. The transformation, or synthesis, of expected behavior to a structural description is a difficult task. The structural description is a geometrical description of the artifact with the topological configuration of types of elements, such as a beam or trusses, and it contains material properties and environmental effects, such as loading. The iterative process of F_0 to B_0 to S_{t^0} to B_{t^0} and a comparison of behaviors B_0 and B_{t^0} is performed through the tasks of formulation, synthesis, analysis, and traditional evaluations until a suitable structural design description is decided upon. The task of construction uses the selected structural design description to create an actual physical structure (S_{t^n}).

The monitoring transformation maps the physical structure to the measured behavior (B_{t^n}). A comparison of the measured (B_{t^n}) and predicted (B_{t^0}) behaviors should lead to improved structural representations of both the physical structure $(S'_{t^{n+1}})$ and the analytical representations (B_{t^0}).

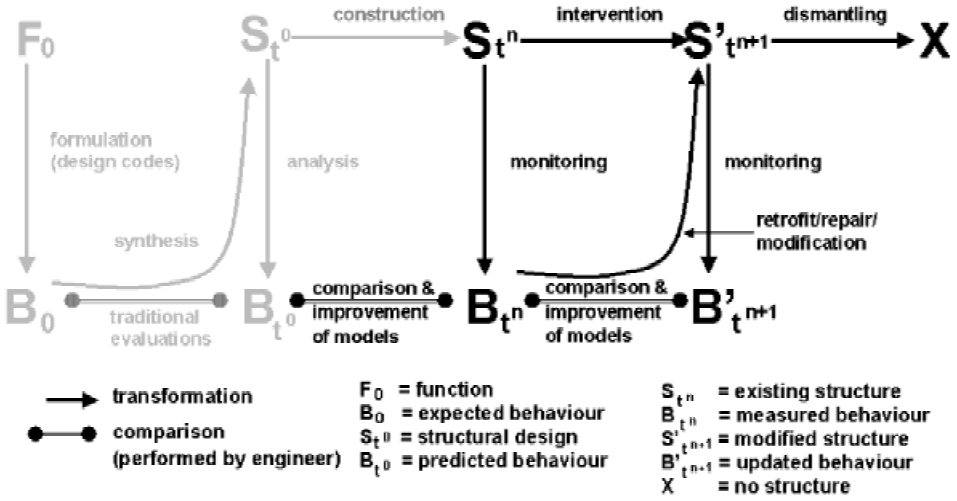

Fig. 1. Engineering information is classified into categories in terms of function, behavior, and structure. Engineering tasks, such as synthesis, analysis, and construction, transform this information from one category to another. Monitoring, modification, and prediction are added to the schema. Transformations are iterative (not shown).

https://doi.org/10.1017/S0890060402163062 Published online by Cambridge University Press

The updated behavior ($B'_{t^{n+1}}$) and modified structure ($S'_{t^{n+1}}$) are later stages in the structural life cycle. The *X* illustrates the end of the structure's life.

The use of the schema enables a classification of information and task to be more simply translated into a software structure to be implemented in a computer. Each task is an iterative procedure that must be decomposed into manageable subtasks for ease of use and to fully exploit available information.

2.2. Description of TD

Each task transformation (as defined in Section 2.1) is divided into five modules (Fig. 2). These modules were specifically chosen in order to encourage the employment of multiple solutions, comparisons, and viewpoints. These aspects have been used in engineering tasks such as design and analysis (Stalker, 2000).

2.2.1. Data management (DM) module

The purpose of this module is to examine the validity of software input. It is important that the information input (in any form, whether a database, matrices, or topology declarations) is realistic. The DM module enables the semiautomation of validating this input heuristically, statistically, and visually and by utilizing plug-ins. A heuristic approach is computationally represented through constraints by the specification of boundaries. Input values must lie within these boundaries in order for them to be acceptable. The statistical approach is a more historical approach, where the mean and standard deviations of input sets are calculated using previous input sets. A normal distribution of errors is assumed. Using simple statistics, confidence levels are set in order to judge the validity of each input set. Examples of such statistical approaches can be found in Stroud (1995). Input is visualized using an annotated visualization in the visual approach. Data points are labeled with pertinent information. Any doubtful values are highlighted by the system. This enables the engineer to interactively accept and reject values simply by observing the input. Such a presentation with highlighted aberrant input is a form of active decision support (Smith, 1996). Plug-ins are software developed for the validation of computer-aided design input and finite-element analysis input (CADFIX, 2000). Therefore, one method would be to reuse these software packages.

The engineer decides how to use the input as admission, omission, or rejection. With an admission the engineer accepts the input as is, including aberrant values. In an omission the engineer deletes aberrant values or asks the DM module to do so automatically. This leads to an incomplete input, but there are still enough data points for the set to be useful. For a rejection the engineer is notified that the quality of the data set is poor. This normally means that there are too many aberrant values or the set does not have enough values to be of use. This may be either the original input or the resulting input after having passed through the omission stage. During model formulation and selection, decisions related to model sensitivity may affect DM methods.

2.2.2. Model selection (MS) module

A space of solutions is more useful than just one solution. Therefore, the MS module contains a choice of models that represent various combinations of behavioral assumptions for the engineering task to be performed.

This approach uses model-based reasoning (MBR), which uses models abstracted from reality, formulated by engi-

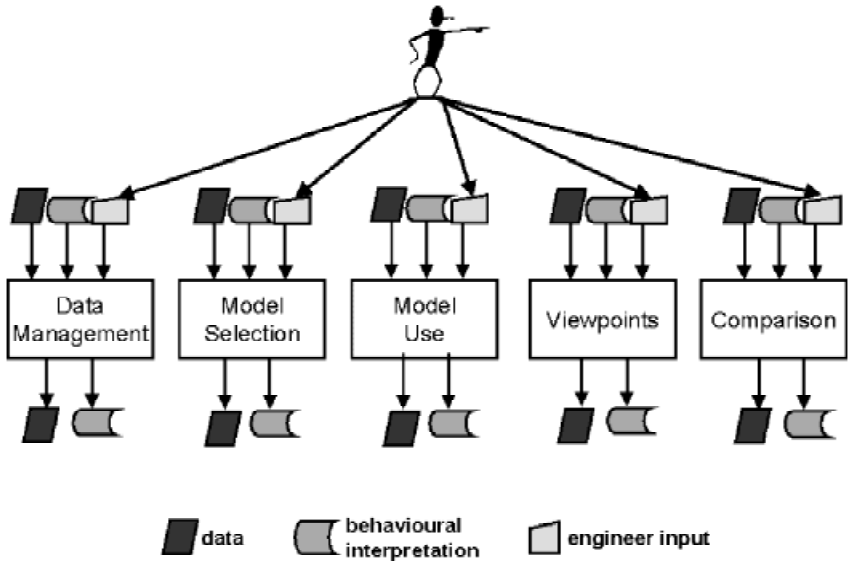

Fig. 2. Task decomposition is represented by five modules. Each module can receive input from the engineer, and they do not have to be used chronologically.

https://doi.org/10.1017/S0890060402163062 Published online by Cambridge University Press

neers or taken from engineering literature, as opposed to heuristics given by experts (rules). Where possible, engineering models are founded on sound physical principles. In this way various representations of structure lead to the calculation of different structural behaviors that are then employed by engineers for subsequent decision making. During model formulation and selection, decisions related to model sensitivity affect DM methods. The question of when to use a certain model is not addressed because the assumption of the MS model is that engineers know which models they want to use.

Table 1 is a list of possible modeling assumptions for the Lutrive Bridge in Switzerland. These assumptions are organized into the following four categories:

1. Fundamental indicates whether a given assumption is fundamental to the creation of a minimal model.
2. Accuracy indicates whether a given assumption will affect the accuracy of the model.
3. Verified indicates whether a given assumption is made after verifying or analyzing structural behavior.
4. Hypothesized indicates whether a given assumption is assumed due to a lack of knowledge.

These assumptions were used for structural monitoring purposes by Raphael and Smith (1998) and were originally taken from design models and used for analysis (Robert-Nicoud et al., 2000). It can be seen by looking at the X's in each box in Table 1 how models are constructed. For example, the assumption of pure beam bending is fundamental to the creation of a minimal model and can be verified by the engineer. These models are represented in a GUI by engineering plans and symbols.

2.2.3. Model use

In order to support current practice, the model-use module makes parameters accessible so that engineers can modify them. This is because the behavior of a structure, which is represented by a model, may change significantly when model parameters are changed. Table 2 contains model parameters that can be changed. The same assumption classification is used as before. Hence, model parameters are categorized according to whether they are fundamental to model creation, affect model accuracy, can be verified, and are hypothesized. As model parameters become available for engineers to modify, MBR becomes a semiautomatic and explicit approach and thus is more accessible to engineers than heuristic rules.

2.2.4. Viewpoints

Viewpoints is a module that gives the engineer the possibility of selecting partial models or subsets of data in order to consider models and data from alternative views. It enables the exploration and exploitation of both data and model use. This module depends on the following factors:

- *Focus:* Through viewpoint fixation, task objectives become the focus of the task. For example, task objectives can be the realization of elegance, efficiency, economy, and utility (Billington, 1995; Shea, 1997). Without focus, objectives may be overlooked.
- *Exploration:* Through the consideration of the space of solutions from a specific point of view, the space is altered. This can be done by selecting part of a structure or part of a data set. This is viewpoint exploration. Alternatively, a new space is created for investigation purposes (Navinchandra, 1991). This may be performed through the addition or deletion of data or by looking at particular structural aspects.
- *Exploitation:* Through viewpoint exploitation, the space of solutions is examined in order to make the best possible use of the available information (Smithers, 1998).

Table 1. *Modeling assumptions for Lutrive Bridge*

Assumption	Fundamental	Accuracy	Verified	Hypothesized
Pure beam bending	X		X	
Continuous oversupports	X		X	
Linear-elastic behavior	X		X	
Twin column support		X		
No relaxation in prestressing force	X			X
Rigid beam and column connection		X		X
Cracks at the supports	X		X	
Rotational springs within span	X		X	
Deep beam hypothesis		X		
Load carrying capacity of deck		X		
Pin roller at supports		X		X
Point loads	X			
Constant temperature gradient	X			
Linear temperature variation				X

Data adapted from Robert-Nicoud et al. (2000).

https://doi.org/10.1017/S0890060402163062 Published online by Cambridge University Press

Table 2. *Explicit model parameters*

Assumption	Fundamental	Accuracy	Verified	Hypothesized
Varying moment of inertia		X		
Column stiffness during bending		X		
Plan curvature of beam		X		
Young's modulus				X
Hinges at midspans		X		X
Nonlinear material		X		
Geometric nonlinearity		X		
Support settlement	X	X		
Weight or position of point loads	X			
Distributed wheel loads			X	
End conditions		X	X	X
Stiffness coefficients				X

The parameters are represented so that engineers may change these parameters and experiment with different behaviors. Data adapted from Robert-Nicoud et al. (2000).

2.2.5. Comparison

Comparison, the last module in the TD, is the simplest way to evaluate the validity of a solution. It is a form of passive decision support (French, 1986). Comparisons may be performed between tasks or between iterations of one task. In the case of task comparison, results from another task can be read in and compared. An engineer may want to compare predicted behavior to measured behavior (B_{t^0} and B_{t^n}) in order to measure the disparity between the two behaviors so that models of predicated behavior can be improved and used more accurately in the future. A comparison between iterations of one task may give multiple solutions that are produced, for example, by changing parameters in the models. These multiple interpretations are considered in order to exploit the available information. Each time a new interpretation is created, the comparison module stores it with all the relevant information, such as the input and the model with its parameters. This is kept until an engineer deletes it.

The validity of a solution is judged using the same methods as the DM module for judging the validity of system input. These methods are heuristic, statistic best fit, visual, and plug-ins. Other forms of comparison may be useful (e.g., nondimensional). However, these methods are left to the engineer's choice. In order to access the TD approach, it is necessary to have an easy-to-use GUI in the form of the engineer identikit.

2.3. Engineer identikit

The engineer identikit is the part of ECI concerned with GUI development specific to engineers' needs. It takes a user-centered design-based approach (Preece et al., 1997; Dix et al., 1998) to produce a GUI that is comfortable and intuitive for an engineer to use.

Engineers are trained to perform tasks and meet specific objectives. They often use formalized procedures for confronting problems. They use explicit knowledge, which takes a hierarchical form, in order to classify information during engineering tasks. Engineers make trade-offs between competing objectives such as time, cost, and quality. During discussions they describe physical behavior or properties of artifacts using mathematical formulae. They often employ a specialized graphical language that uses predefined combinations of symbols and graphical representations, thereby communicating without ambiguity. Finally, engineers usually have specific tasks to perform and want to do them in different ways. This engineer identikit (Fig. 3) should be looked upon as a GUI tool box builder where engineer aspects are used in software systems as desired. Part or all of it may be used.

2.3.1. Computer literacy

Engineers are generally computer literate. By 1975 computing was on the syllabus for engineers at universities and books that used computing examples, such as McCormack (1975), were recommended reading. Thus, manual calculation for engineering tasks is no longer a feasible alternative. The ECI approach is grounded in the belief that good solutions are the combined effort of engineer experience and computer support.

2.3.2. Classification

Structural engineering uses a more rigorous terminology and information classification than domains such as computer science. Classification may be based on material properties, which in turn determine material behavior, acceptable loads on materials, and the tests that should be performed. This information is imperative because it dictates how engineers should use materials. It is also useful for talking to other engineers and is therefore a practical way to represent information in a computer.

Figure 4 contains a hierarchical classification of a bridge. This is a typical decomposition strategy for representing bridges during design, and such decompositions should be incorporated into software systems (Boulanger, 1997).

https://doi.org/10.1017/S0890060402163062 Published online by Cambridge University Press

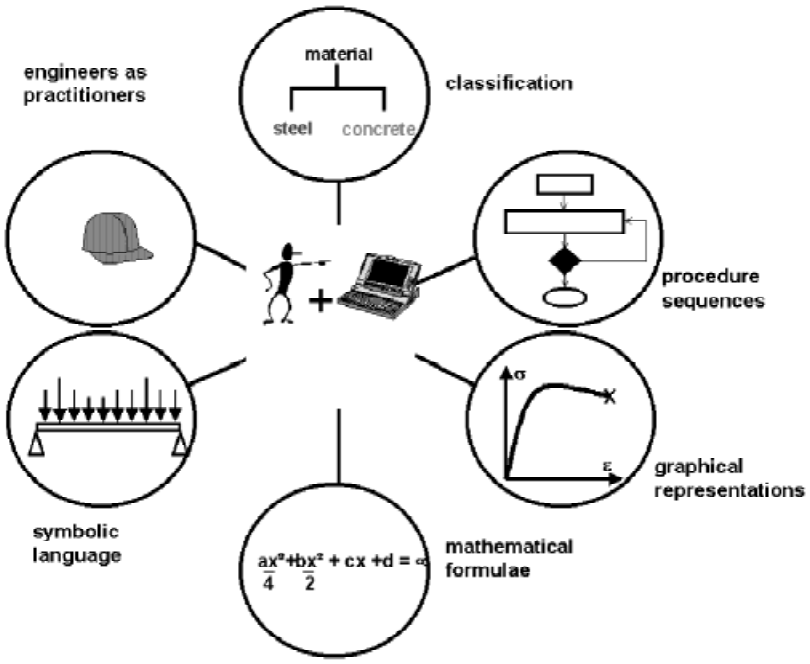

Fig. 3. Engineers use formulae and graphical representations to work with knowledge. These characteristics are reproduced in ECI software.

2.3.3. Task objectives

Engineers are trained to perform a finite number of tasks and to realize objectives in a certain manner. Figure 5 contains two examples of computer generated roof trusses. Both trusses are functionally sound, but neither satisfy the engineers' objectives that are dictated by society and culture. The left-hand truss is much heavier than the right-hand one, which is asymmetrical and has more cross sections and joints. From an aesthetic viewpoint, the left-hand truss is more pleasing.

Billington (1995) classified task objectives into efficiency, economy, and elegance. Shea (1997) added utility. These objectives are weighted differently in each project and may even be dictated by politics (Boulanger & Smith, 1994). Thus, it is important that engineers have the opportunity to interact with software in order to choose the task focus and to prioritize the task objectives. In this way, it is the engineers who steer the calculation process to achieve the desired results instead of the computer providing engineers with results they will not use.

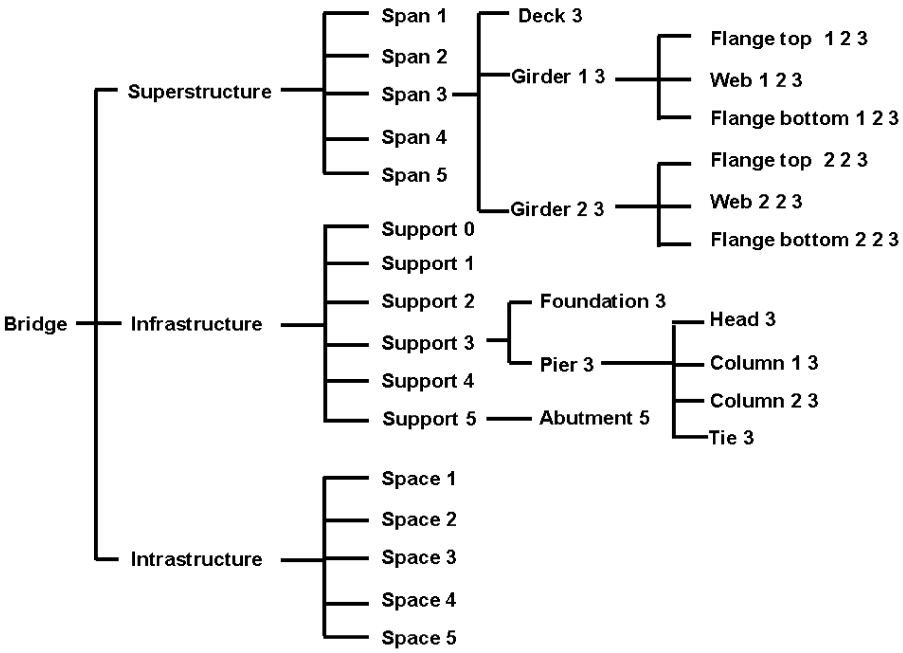

Fig. 4. A hierarchical classification of a bridge. Adapted from Boulanger (1997).

https://doi.org/10.1017/S0890060402163062 Published online by Cambridge University Press

Fig. 5. Two computer generated trusses illustrate the task objectives of elegance and utility. Adapted from Shea (1997).

2.3.4. Procedures

Engineers often follow predefined procedures when performing tasks, for example, the tensile test (Megson, 1987). These procedures are found in the codes of practice, which gives guidelines for tasks related to structures throughout their life cycles. These rules are codes and procedures for design, evaluation, and testing in order to ensure structures meet with criteria such as safety and serviceability. Procedures are rigorously structured. This approach inspired software engineers to be equally rigorous in designing software (Pressman, 1994).

2.3.5. Graphical representations

Graphical representations are used in scientific fields such as physics, mathematics, and engineering. An image may contain much information, and in structural engineering, certain graphical representations are familiar to engineers. They often sketch the bending moment or deflection of a beam using a 2-dimensional graph. Figure 6 contains the graphical representation of the direct relationship between stress and strain for ductile material. Therefore, it is advantageous to include such representations in the engineer identikit. Engineers can identify the meaning behind these graphical representations and can see if the calculations are giving the desired results.

2.3.6. Mathematical formulae

When discussing a project, engineers describe the physical behavior and properties of artifacts through mathematical formulae. For example, the deflection of a simply supported symmetrical beam of length l that bends under a uniform load q while its plane cross sections remain plane and normal to its longitudinal fibers can be described mathematically as a quartic relationship:

$$\frac{qx}{24EI}(x^3 - 2lx^2 + l^3), \tag{1}$$

where q is the uniform load per unit length, E is Young's modulus, I is the moment of inertia, and x is the distance from the left-hand support. Most other HCI user groups prefer mathematical formulae and algorithmic details to be hidden and results to be represented graphically, whereas engineers prefer to manipulate formulae symbolically and quantitatively.

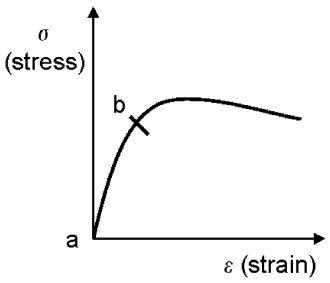

Fig. 6. Graphical representations are useful and recognizable. Adapted from Megson (1987).

2.3.7. Symbols

Alongside formulae and graphical representations are the combinations of predefined symbols engineers employ in order to attempt to communicate without ambiguity. These symbols are precise. Figure 7 illustrates two similar structures with differing supports. The structure on the left has two triangles, each triangle represents a support and one support has two circles. The structure on the right does not have any circles or triangles. The omission of circles and triangles indicates that the supports are fixed. Not only are fixed supports more expensive than hinged supports, they change the behavior of the proposed structure. Thus, removing the circles and triangles will produce a different structure. Such an error is semantic and not merely syntactic. Symbols are precise. The incorporation of such symbols into GUIs builds on existing engineering work procedures and provides interface transparency.

2.3.8. Engineers as practitioners

Engineers are practitioners. Their characteristics are similar to other practitioners, such as in business, law, and medicine. Society trusts them to perform in a responsible manner. Although laws may fix limits on their activities, they have much freedom to identify creative solutions. Engineers read plans, have a legal responsibility, make tradeoffs, and need flexibility.

Plans. Due to the size and complexity of structures, many plans (geometric descriptions on paper) are needed to provide accurate geometrical representations. For example, during construction of a midsize bridge, approximately 20 kg of plans are produced. These plans should be used in GUIs.

https://doi.org/10.1017/S0890060402163062 Published online by Cambridge University Press

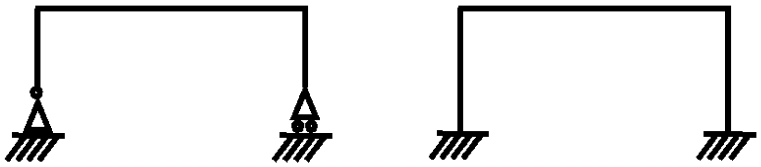

Fig. 7. The omission of triangles and circles changes the structural behavior.

Legal responsibility. Civil engineers have been responsible for their work since Egyptian times. Legal responsibility is widely acknowledged by engineers. With such responsibility, it is important that any tools engineers use, such as computers, must support them in an understandable way and make provisions for them to modify calculations should the need arise. Engineers are often required to justify their decisions during technical litigation processes. This is not the case for many other groups of users.

Trade-offs. Engineers make trade-offs between competing objectives such as time, cost, and quality. These objectives are dictated by the economic and social constraints within which the engineer must work. Different design objectives of elegance, economy, efficiency, and utility produce very different designs. Also, not all domains are influenced so pointedly by additional factors such as the availability of resources and political issues (Boulanger & Smith, 1994). Such factors cannot be easily modeled in a computer. Engineering experience can translate factors into a value with which a computer can work. Therefore, the engineer must be given the opportunity to interact with software in order to choose task focuses and prioritize task objectives. Good interaction between the engineer and the computer allows the engineer to steer the calculation process in order to achieve the most reasonable results within given constraints.

Flexibility. Finally, engineers have specific tasks to perform and the order in which steps of a task are executed may change from engineer to engineer. Therefore, flexibility of approach is often an important requirement within and between tasks. This flexibility should be reflected in software (Boulanger, 1997). Software should not impose a specific sequence of steps if it is to be adopted by engineers.

2.4. Application of ECI to engineering tasks

The application of ECI to a given task is approached in three steps, as shown in Figure 8.

1. The relevant part of the organizational schema is selected in order to have a foundation for the software.
2. Modules are chosen and used as necessary, depending on the task. Not all modules may be needed.
3. Desired engineering characteristics are chosen from the engineer identikit and added to the software in order to support the modules.

ECI is representative of a decade long trend to replace automated systems with a collaboration between the engineer and the computer. In this way, the engineer's experience and the computer's computational capacity are used together. Interactive systems are more useful than automated reasoning because they provide transparency and enable collaboration between engineer and computer. The following section illustrates the application of the ECI blueprint to the task of structural monitoring.

3. STRUCTURAL MONITORING TOOL KIT (SMTK)

3.1. SMTK organizational schema

Structural monitoring and the interpretation of data are the transformation link between the existing structure and measured behavior as shown in Figure 1 and described in Section 2. Monitoring data interpretation uses measured results and endeavors to find an explanation for structural behavior.

The use of the schema makes the difference between the expected behavior and measured behavior explicit. The assumptions made during the task of analysis, in particular, loading assumptions, are used to calculate the expected behavior, which may be very different than the measured behavior. These two factors are the main motivators for structural monitoring. The use of the organizational schema makes it easier to define software objectives.

The schema also enables the engineer to explicitly identify the types of information SMTK needs to interpret monitoring data successfully. This information is as follows:

- *Monitoring data:* This data may be read in either directly on-line while on-site or off-line, from a database. The IT represents raw structural behavior; therefore, it is the data that is to be interpreted. This information is treated by the DM module and is represented in the organizational schema as B_{t^n}.
- *Structural knowledge:* This knowledge describes the structure being monitored in terms of its dimensions or coordinates of length, breadth, and height. Plans of the structure that sketch these coordinates are scanned in

https://doi.org/10.1017/S0890060402163062 Published online by Cambridge University Press

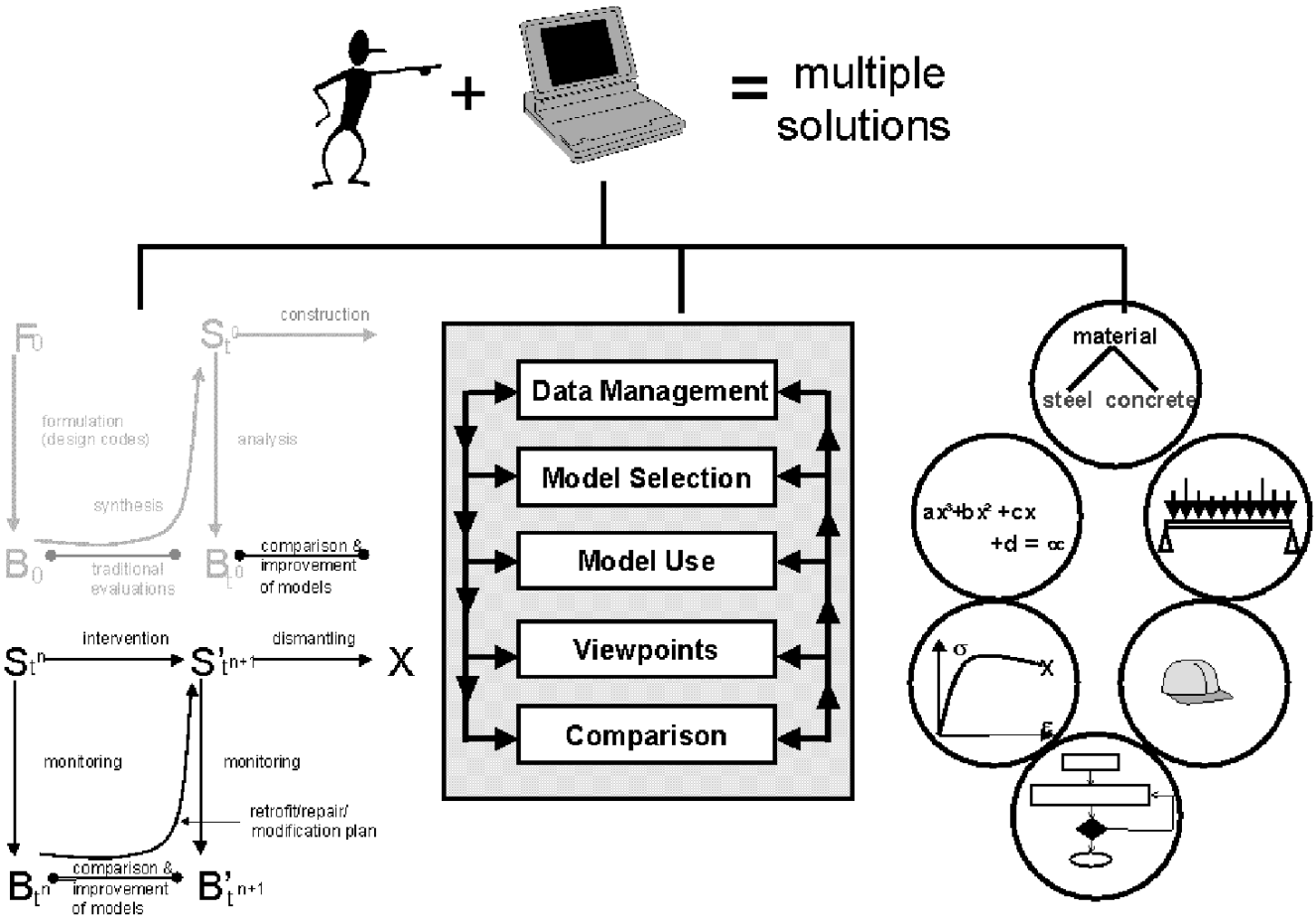

Fig. 8. ECI contains an organizational schema, task decomposition, and an engineer identikit.

at the same time so that the engineer has a visual representation of the structure. The structure's coordinates are used for calculation in the model-use and comparison modules. The plans and the coordinates are used together in the viewpoints module for visual representation and calculation. This knowledge makes up part of existing structure. More information for the existing structure is expressed as loading and environmental considerations.

- *Monitoring equipment:* This information describes the position of monitoring apparatuses on or in the structure. The equipment description is used for calculation in the model-use module and for visualization purposes in the viewpoints modules.
- *Interpreted information:* Previous interpretations are read into SMTK.

3.2. TD for SMTK

The ECI TD of five modules was instantiated as SMTK modules. The modules are described in the following paragraphs.

3.2.1. DM module

This module enables the validation of the current data set. A given data set is one group of measurements. If a bridge is equipped with 30 deformation measuring instruments and each instrument provides a value at a given time, the result is one data set of 30 points. If the same action is performed eight times in 1 day, then eight data sets will be available. Each data set can be viewed as a snapshot of structural behavior at a specific time. Data quality is important, because some measurements may be false due to measuring apparatus problems such as a mechanical fault or incorrect use.

Values can be judged heuristically or statistically. The DM module heuristically judges the aberrant values (outliers). The history of each value is considered as shown in Figure 9. The box in the figure illustrates the constraints placed on acceptable values for this data point. Points outside the box are brought to the engineer's attention by SMTK because they have been judged as invalid. This box changes considering the history of the data point. If the data point is within a series that increases or decreases in value and the data point in question does not, SMTK high-

https://doi.org/10.1017/S0890060402163062 Published online by Cambridge University Press

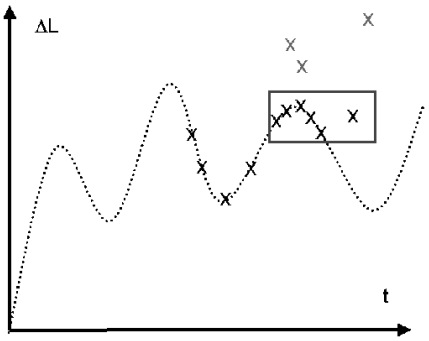

Fig. 9. Examples of judging data.

lights the point so that the engineer is aware of its possible aberrant value.

3.2.2. MS module

Each bending deformation measurement can be treated as a data point that is plotted against a curve representing the expected deformation (Vurpillot, 1999). In other words, the data is reduced using a model. These curves, however, are only approximations. Different assumptions lead to very different deflection curves. Thus, it is better to have a selection of deflection curves from which to choose.

The models contained in this MS module should be chosen, or created, by the engineers who are going to use SMTK with respect to the structure they are monitoring. The models are represented in the module by symbols and mathematical formulae from the ECI identikit.

3.2.3. Model use

The model-use module enables the engineer to explore a chosen model in detail. The choice of which parameters the system makes explicit should be made at the same time that the MS module is formulated. Possible parameters that may have an effect on the behavioral interpretations are stiffness coefficients (e.g., the moment of inertia) and types of loading.

3.2.4. Viewpoints

This module uses a plan of the structure and monitoring apparatus. The engineer may click on the structure and the monitoring apparatus in order to select areas of interest. For example, there may be several types of monitoring apparatus in one structure. The engineer may select just one type of monitoring data from one piece of apparatus in order to interpret these results alone. Alternatively, the engineer may want to look at specific parts of the structure and consider its behavior, independent of the global structural behavior.

3.2.5. Comparison

Once calculations have been performed, they may be stored for comparison with other interpretations of a particular data set. Comparison may be performed according to different criteria:

- *Time for comparison on snapshots of behavior at certain times:* This is a common approach with engineers who may want to compare a structure's behavior on a monthly or even yearly basis.
- *Multiple models:* Different models give different behavioral interpretations. Therefore, an engineer may want to try out several models in order to find a behavior that fits the data.
- *Multiple apparatuses:* Each apparatus may produce different data. The engineer may want to compare measurement results.

Once several functions for structural behavior are found, the comparison module allows the engineer to compare these functions statistically or visually: SMTK uses statistics to find which function fits the data best, or the engineer may be able to see which function is more representative of structural behavior.

Each curve represents the following: a function for curvature that is calculated in order to calculate the function for deformation, a function for deformation that describes the global deformation of the structure being monitored, a data set that is reduced in order to calculate the above functions, an indication of which model the deformation function represents, and the model parameters contained in the deformation function that the engineer specified during the data interpretation.

Other interpretations from data sets may be read into this module using this format so that comparison may take place. Many comparisons are possible (e.g., nondimensional comparison). Thus, it is for the engineer using the system to decide what types of comparison are appropriate.

3.3. SMTK engineer characteristics

The SMTK GUI was made up using the following characteristics from the engineer identikit as follows:

- *Task objectives:* The task objectives of structural monitoring are to find an interpretation of structural behavior.
- *Formal procedures:* The models in the following example (Section 3.4) are based Bernouilli beam theory (simple or pure beam bending).
- *Graphical representations:* A scatter plot illustrates a data set and a graph represents the calculated functions for global deflection. These are presented to the engineer in the DM module and the model-use module.
- *Mathematical formulae:* The functions that represent global deflections are described mathematically alongside the graphical representation in the model-use module. Behavioral assumptions that are made by different models are mathematically described in the MS module.
- *Plans:* The viewpoints window contains a plan view and a cross-sectional view of the structure to be analyzed. Engineers use these plans to make correlations between the monitoring data and the structure.

https://doi.org/10.1017/S0890060402163062 Published online by Cambridge University Press

- *Symbols:* The MS module uses symbols to represent structural behavior. This enables the engineer to understand the behavioral assumptions contained in the models.

3.4. SMTK for Versoix Bridge

SMTK was used to interpret monitoring data taken from the Versoix Bridge, which supports part of the dual carriageway (N1) close to Geneva Airport in Switzerland. It was built in the 1960s and is made up of two parallel bridges. Each bridge is made of two prestressed beams, which support a reinforced 30-cm concrete slab. In the interests of safety, the road was enlarged on both sides to create a hard shoulder. Concrete was added on both sides of both bridges in direct contact with the old concrete as shown on the right side of Figure 10. There were concerns about the interface between old and new concrete, concrete shrinkage and the spatial deflection of the bridge due to these extensions. Thus, during the extension process, the bridge was equipped with a network of SOFO fiber optic sensors in order to measure local deformations. A description of how the SOFO sensors were developed and how they work can be found in Inaudi (1997). Two beams of the Versoix Bridge were equipped with 96 fiber optics. The first beam (A) was equipped with five cells of sensors, and the second beam (B) was equipped with seven cells of sensors. Each cell contains eight sensors, as shown by the cross-section schema in the viewpoints window of SMTK in Figure 11. The data is read by the monitoring equipment and stored in a database called SOFO-DB (Inaudi, 1997). SMTK accesses the database and treats each data set as a separate file. SMTK was implemented in OpenGL and C/C++ and runs on a Silicon Graphics Indigo 2 (SGI). The software is presented module by module, and ECI engineer identikit characteristics are highlighted under each module heading.

3.4.1. DM module

A data set is one set of measurements taken at a specific moment in time on the bridge. Therefore, a given data set contains a maximum of 96 data points. If many readings were taken in 1 day, the data set is referred to by its date and time. This enables comparisons to be made at various times of day using the comparison module. The viewpoints window (Fig. 11) illustrates the layout of each cell of sensors. In order to calculate the radius of curvature successfully, the cell needs to produce a minimum of two data points. These values must have one from the top and one from the bottom. In a normal data set the sensors at the top of the beam give positive values and the sensors at the bottom give negative value. The data set is plotted on a graph in the module. The *y* axis measures positive and negative deformations. The *x* axis is the length of the beams. Each line represents 1 data point. It is annotated with the name of the

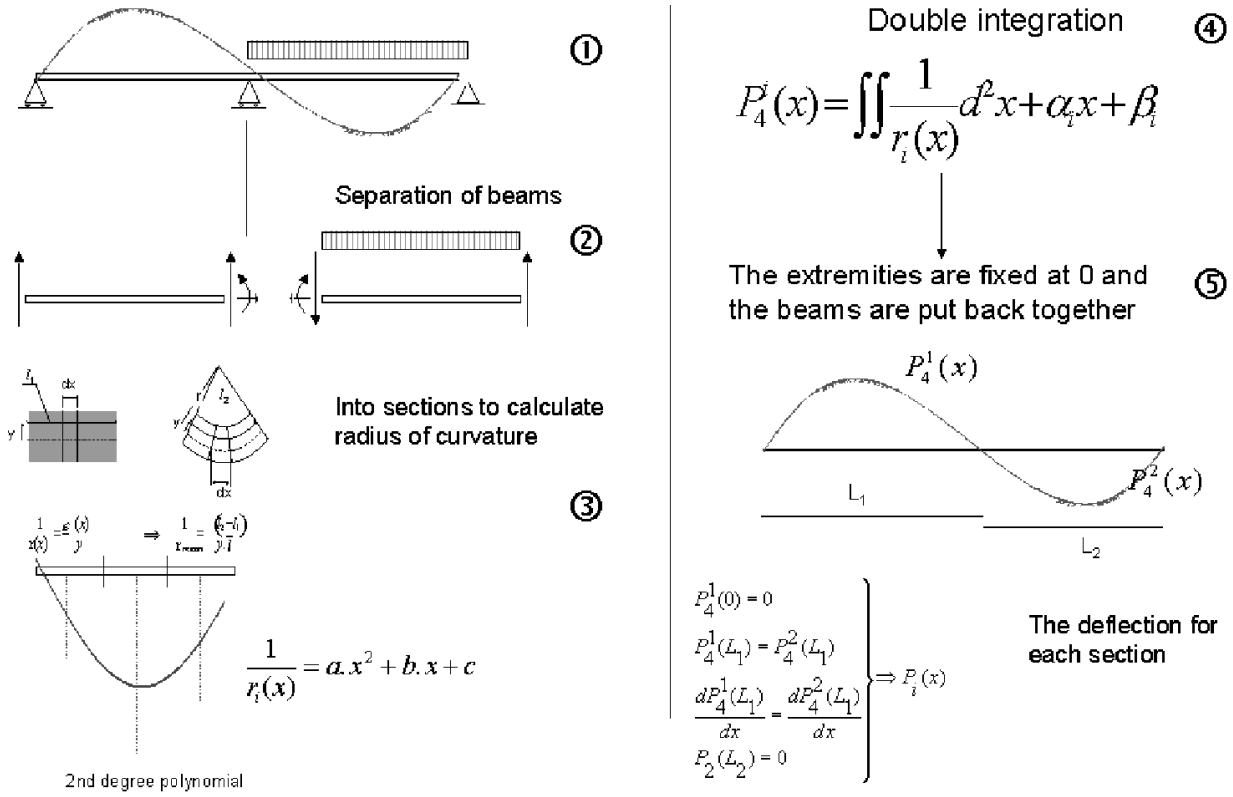

Fig. 10. A summary of the data reduction model used in SMTK for the Versoix Bridge. Vurpillot (1999).

https://doi.org/10.1017/S0890060402163062 Published online by Cambridge University Press

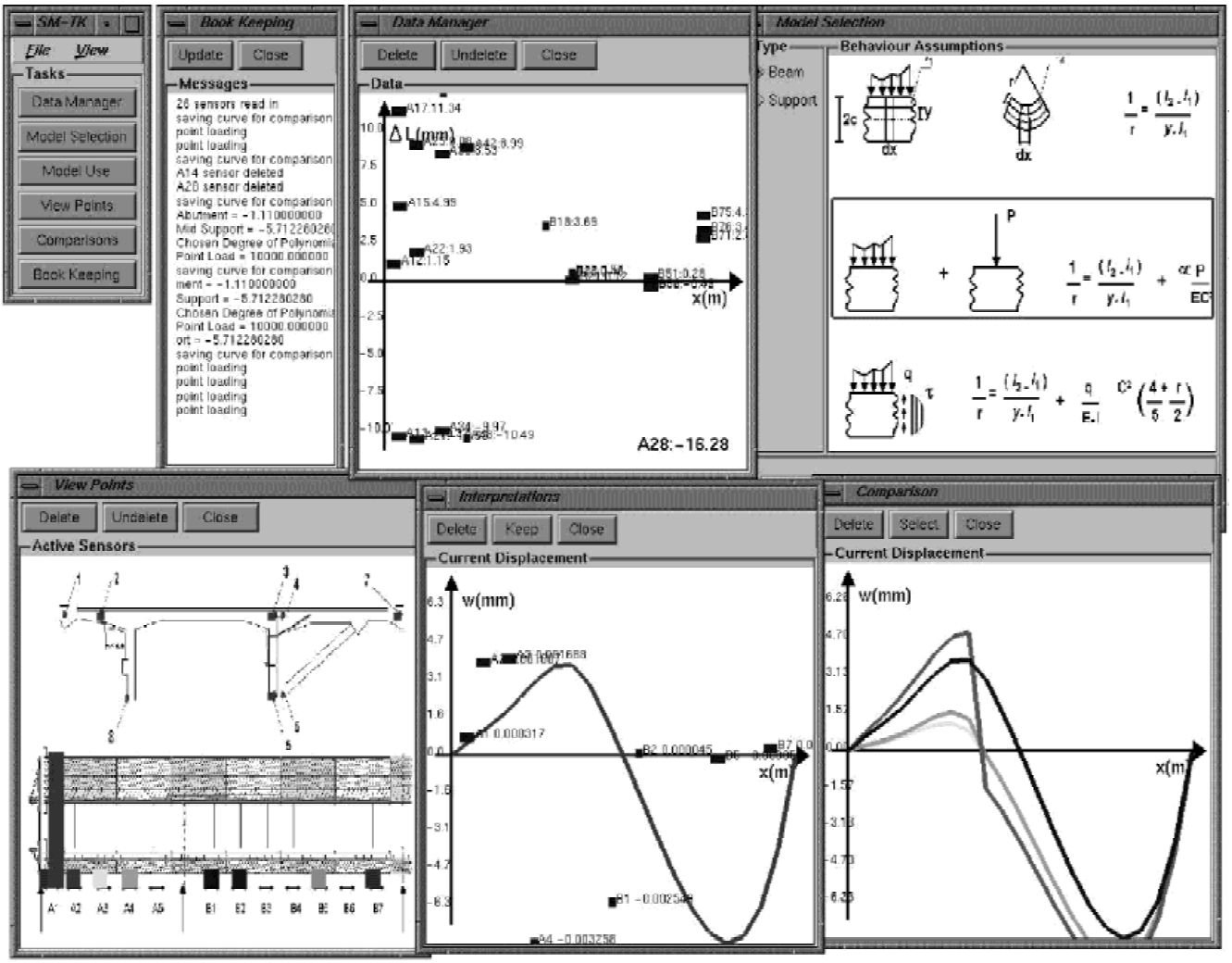

Fig. 11. A screen shot of the SMTK. In the top left-hand corner of the figure is the SMTK tool box through which the engineer opens and closes windows. The windows are from top left clockwise bookkeeping, data manager, model selection, comparison, model use, and viewpoints.

sensor and its corresponding local deformation. The data points are plotted on the *x* axis to represent their position in the beams, and on the *y* axis to represent their deformation. The DM window in Figure 11 illustrates a typical Versoix Bridge data set. Visualizing the data in this way makes it easier to see aberrant values and allows the engineer to become familiar with a scattering of data and more easily identify bad data sets. The engineer may delete data as required. Clicking on a data point displays the information in the bottom right-hand corner of the module. However, if an engineer clicks on the delete button and then clicks on a data point, the data point is removed from the data set. This removal is reflected in the DM window, in the viewpoints window, and in the end calculation.

It has been shown that a removal of a total cell of data values may result in a 10% difference in the accuracy of the result (Vurpillot, 1999), and this becomes more obvious when visualizing data in such a way.

3.4.2. MS module

The MS module contains three models. One model was specifically developed for the interpretation of data from the Versoix Bridge (Vurpillot, 1999), based on pure beam bending. The second model extends this base model to deal with point loads over a specific sensor point. The third model emulates shear.

Pure beam bending. This model assumes that the law of Bernoulli is satisfied. That is to say, the plane cross sections of the beam remain plane during bending. The model requires that each beam is divided up into sections (as shown in stage 1 of the left-hand side of Fig. 10). It is assumed that each section has a moment of constant inertia, a uniform load across its length, and supports only at its ends. Each section is made up of a group of cells. These cells contain a minimum of two sensors, which are placed at parallel to an assumed neutral axis and are

https://doi.org/10.1017/S0890060402163062 Published online by Cambridge University Press

used to calculate the mean radius of curvature (Fig. 10, Stage 3):

$$\frac{1}{r_m} = \frac{\Delta L_2 - \Delta L_1}{(y_1 - y_2)L}. \tag{2}$$

These local curvatures are used to describe the curvature of the whole section. Therefore, the local radii of curvatures are fitted to a curvature function using a polynomial of the appropriate degree:

$$P_2(x) = ax^2 + bx + c. \tag{3}$$

The polynomial has three unknowns (a, b, c). Therefore, only three independent measurements (i.e., cells) are necessary to express the curvature function of a single beam section of the Versoix Bridge. If there are more than three cell results, which is the case for this bridge, the system of equations should be solved by least squares. A summary of this model is given in Figure 10. For a more detailed description see Vurpillot (1999).

Point loads. Local change in curvature deflections due to point loads may be underestimated using the previous model for transforming deformations into beam deflections. Therefore, a correction for additional deflection of point loaded beam as proposed in Timoshenko and Goodier (1970) is added to the model. The conditions for this model correction validity are that the length of the SOFO sensor is four times greater than half the depth of the beam and that the point load is over the midpoint of a SOFO sensor.

In order to account for the increase in curvature caused by this point load compared with curvature caused by distributed loading, an extra term is added when calculating the radius of curvature. Thus, Eq. (2) becomes

$$\frac{1}{r_m} = \frac{\Delta L_2 - \Delta L_1}{(y_1 - y_2)L} + \alpha \frac{P}{EC^2}, \tag{4}$$

where α is a numerical factor varying along the beam (Poisson's ratio), the value of which is given in Timoshenko and Goodier (1970); P is the force per unit thickness of the beam (e.g., wheel load); and C is half the depth of the beam. This is useful for modeling point loads (e.g., a truck during test conditions is parked over a SOFO sensor). An illustration of the test conditions of such a test that was performed in Switzerland is given in Perregaux (1998).

Shear. The shear model emulates structural behavior using the same approach. The assumptions are that there is uniform loading on the beam and the SOFO sensors are placed parallel to the neutral axis. By adding a term, which is referred to as the effect of shearing force (Timoshenko & Goodier, 1970), Eq. (3) is rewritten as

$$\frac{1}{r_m} = \frac{\Delta L_2 - \Delta L_1}{(y_1 - y_2)L} + \alpha \frac{q}{EI} C^2 \left(\frac{4}{5} + \frac{r}{2}\right). \tag{5}$$

Supports. Following the first group of assumptions as described by each of the models above (pure beam bending, point loads, shear), the second group of assumptions consists of the support conditions. These are chosen by the engineer, who clicks on the desired model and the support assumptions, activating the data interpretation process.

3.4.3. Model use

The model-use module is used to refine the following model parameters in order to change the interpretation. These parameters were already used in the equations in the previous section. The variables are made accessible to the engineer through the SMTK GUI.

Degree of polynomial: The degree of the polynomial may be specified by the engineer.

Position and weight of point load: The position and the weight of the point load (P) can be specified in order to indicate where the results of the SOFO sensor should be adjusted.

Moment of inertia: The moment of inertia (I) can be input by the engineer and is used for calculation purposes for the shear model.

Boundary conditions: By default it is assumed that the displacement boundary conditions are zero. However, an engineer may want to declare them to be less than zero in order to represent support settlement.

The model-use module has a graph. The x axis shows the length of the beams, and the y axis illustrates the global deflection. The deflection function has a mathematical label.

3.4.4. Viewpoints

The viewpoints window is used by the engineer to select areas of interest in the bridge. Structural images were scanned from the original plans in order to keep within current working practice of an engineer. The engineer may click on parts of the plans that are representative of the structural knowledge contained in the system. For example, an area of the structure can be selected so that the monitoring data from this area alone is considered. An engineer may want to concentrate on the supports or in the middle of the beam where one expects displacement to be quite small.

The viewpoints window in Figure 11 illustrates the visual presentation of the Versoix Bridge in the viewpoints module. At the top of the window there is a cross-sectional view that illustrates the position of the eight sensors. The sensors, numbered 1 to 8, were placed in this manner in order to measure the behavior of the new concrete (1 and 2) and the interface between the old and new concretes (3 and 5, 4 and 6) and to obtain an overview of global structural behavior (7 and 8). In Figure 11 the plan view contains highlighted cells that have data measurements. The cell that is highlighted with a long line (cell A1) corresponds to the cross-sectional view above.

https://doi.org/10.1017/S0890060402163062 Published online by Cambridge University Press

In this way solutions can be identified, explored, and investigated. This can be done by reducing or increasing the number of data points, selecting one or two beams, and selecting various cells instead of using all of them. The SOFO sensors that are active and provide values for the current data set are highlighted in the bridge plan.

3.4.5. Comparison

Each curve represents a data set, a function of displacement, end conditions, moment of inertia, and so on. Hence, an engineer clicking on a curve will receive information about that curve. SMTK indicates the best fit to the engineer. The best fit is calculated by summing the square of the errors in order to find the smallest error margin.

3.4.6. Bookkeeping

A sixth window, referred to as bookkeeping, is added to the five windows that represent the five TD modules. It keeps a record of all the actions taken during the use of the tool kit. Bookkeeping is a simple window that displays a list of all actions taken by the engineer since software startup. In this way the engineer has a guide to the calculations SMTK has performed so far and if data has been added or deleted.

4. RESULTS

The QUIS^{TM} questionnaire was used to assess subjective user satisfaction with SMTK. The results of a pilot study were inconclusive (see Stalker, 2000). Thus, QUIS was used in a SERV-QUAL manner (Zeithaml et al., 1990) so that user expectations of what SMTK should do could be plotted directly against user perceptions of what SMTK actually does in order to derive conclusions about user satisfaction with SMTK. Two groups of eight people were questioned. One group contained structural engineers. The second group was made up of nonengineers from various disciplines.

4.1. A SERV-QUAL analysis of engineers and nonengineers using SMTK for monitoring data interpretation

The questionnaire was divided into six parts. This article presents only parts 2 and 6 of the results. The rest of the results analyze in more detail each part of the ECI framework. More results and discussion can be found in Stalker (2000). Parts 2 and 6 contained the same questions with only the verb "expect," in part 2 replaced by "perceived" in part 6. Table 3 contains both question numbers (e.g., 2/6.1) and verbs (e.g., expect/perceive) and both ends of the measurement scale (e.g., bad/excellent). Thus, each question can be derived from the header in Table 3 and a question number and entry. For example, by looking at the table, it is possible to know that question 2.1 read, "do you expect the software to be frustrating/excellent?" and question 6.1 is "do you find the software frustrating/excellent? At this point the user had a scale of 1–5 (frustrating–excellent) and had to check a number. Part 2 was given to both groups before using SMTK in order to measure expectations. Part 6 was given after use of SMTK in order to measure software perceptions. Figure 12 shows the mean of each question response. The question numbers in Figure 12 correspond to the question numbers in Table 3, which contains a summary of the questions the users had to answer on a scale of 1–5 before (expectations) and after (perceptions) using SMTK.

The midpoint of the rating scale (3) was used as the criterion. Therefore, if the response was above 3, it was perceived to be better than average. In general (Fig. 12) it can be seen that engineers' expectations and perceptions of

Table 3. *Parts 2 and 6 of QUIS questionnaire applied in SERV-QUAL manner for monitoring data interpretation software SMTK*

Question No.	Overall Expectations or Perceptions of SMTK
2.1 or 6.1	Expect or perceive system to be bad or excellent
2.2 or 6.2	Expect or perceive system to be frustrating or satisfying
2.3 or 6.3	Expect or perceive system to be dull or stimulating
2.4 or 6.4	Expect or perceive system to be difficult or easy
2.5 or 6.5	Expect or perceive system to be not useful or useful
2.6 or 6.6	Expect or perceive system to be rigid or flexible
2.7 or 6.7	Expect or perceive system to be nonpertinent or pertinent
2.8 or 6.8	Expect or perceive system to be not user friendly or user friendly
2.9 or 6.9	Expect or perceive system to be non-reliable or reliable

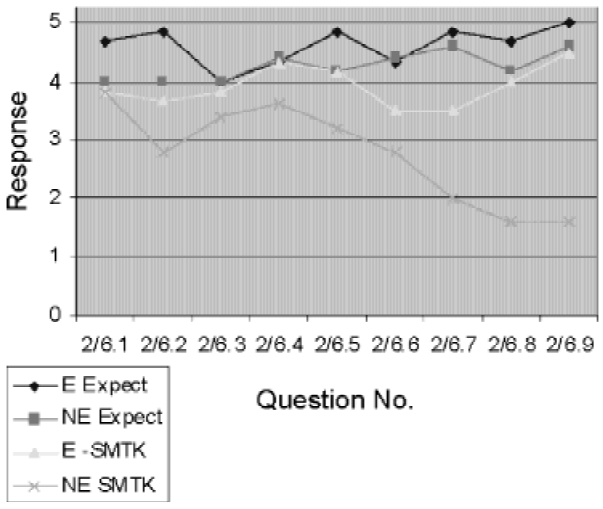

Fig. 12. A SERV-QUAL means analysis of engineer and nonengineer group expectations and perceptions of SMTK for monitoring data interpretation. E-Expect, engineer expectations before SMTK use; NE-Expect, nonengineer group's expectations of SMTK; E-SMTK, engineer perceptions of the software; NE-SMTK, nonengineer perceptions.

https://doi.org/10.1017/S0890060402163062 Published online by Cambridge University Press

SMTK were both above average. In contrast, the nonengineers' perceptions were below average. The results show that engineers were more demanding in their software expectations than their nonengineer counterparts. The line that represents their expectations is higher than that for the nonengineers' expectations. However, engineers did not expect to find the software to be extremely stimulating, flexible, and user friendly (questions 2.3, 2.6, 2.8). Therefore, their expectations of the capabilities of proposed software would in some way be based on their experiences with current software, and it may have been difficult to imagine software, to be flexible and stimulating. Nevertheless, they expected new software to be excellent, satisfying, easy to use, and useful.

The nonengineers were not quite as enthusiastic in their expectations of software that would be useful for interpreting monitoring data. Although the task had been explained to them and the majority of the nonengineer group had experience in interpreting data in a similar way (i.e., looking for trends in the data and curve fitting), they had difficulty anticipating software that could offer sufficient support for such a task. Thus, answers to questions 2.1, 2.2, 6.1, and 6.2 showed that nonengineers were not expecting SMTK to be excellent or stimulating. In terms of stimulation (questions 2.3, 6.3), engineers were not as satisfied as expected. This was because some of the engineers found the models in SMTK to be unsatisfactory. However, their critique of the models in SMTK illustrates that the model assumptions in the system were clearly represented, which allowed the engineers to make comments on the model choice at such an early stage in software evaluation. Also, engineers did not find SMTK to be as flexible and as pertinent as they wanted. For example, SMTK does not link up to a finite elements package and is therefore unable to provide analytical and experimental comparison. This was not the goal of SMTK. Nevertheless, the observation was made that a link between structural analysis and monitoring could be established easily, resulting in a comparison that is rarely performed. SMTK led engineers to imagine ways in which it could be more useful and extended to perform many tasks that they currently have difficulty performing. Finally, engineers expected to be stimulated by SMTK and expected it to be easy to use, which they found to be the case (questions 2.3, 6.3, 2.4, 6.4).

Nonengineers found SMTK to be relatively (in comparison to the engineers) frustrating (questions 2.2, 6.2), nonpertinent (question 2.7, 6.7), not user friendly (question 2.8, 6.8) and nonreliable (question 2.7, 6.7). SMTK uses symbols that the nonengineers did not understand. It has graphical representations, symbols, and plans of structures that are not the tools nonengineers use daily. Thus, these are ECI attributes that are difficult for nonengineers to appreciate. SMTK was described as nonreliable by nonengineers because they did not know what sorts of results to anticipate. Therefore, it was difficult for them to judge whether SMTK gave the right (relevant) answers.

Five of the nonengineer question means were below the average point (3). These results illustrate that SMTK is not suitable for nonengineers.

Engineers reacted more favorably to SMTK than nonengineers because the software corresponded to their needs through the employment of ECI. Engineers were able to understand and use the engineering terminology in the system. Their appreciation of the separation of structural and behavioral information and the explicit rendering of model assumptions was illustrated by the positive response. They found that it was easier to perform monitoring data interpretation using SMTK. Moreover, they had many suggestions to improve this instantiation of ECI for SMTK application. These suggestions were linking SMTK to a finite elements package so that more sophisticated models could be used, creating a link so that a comparison between analysis and monitoring could take place, and exploring how ECI could be applied to other engineering tasks. These ideas are developed in Chapter 7 of Stalker (2000).

The engineers' main criticism of SMTK was the choice of models used in the system. However, the transparency of the interface, which was constructed using the ECI identikit, is illustrated in this criticism because it was possible for the engineers to understand the models almost immediately.

Nonengineers liked SMTK less than engineers because they had difficulty understanding the symbols and found the graphical representations less useful. However, they did appreciate that the task of interpreting monitoring data was well structured by the use of ECI TD, which lent an impression of simplicity to SMTK.

5. CONCLUSIONS

ECI represents a contribution to the need for a specific approach to the design, implementation, and evaluation of interactive decision-support systems for engineering tasks. This paper has presented the application of ECI to the task of structural monitoring. A tool kit called SMTK, which interprets bridge monitoring data, was developed according to ECI. SMTK was given to engineers to evaluate, and their reactions, collected by questionnaire, were analyzed. Results show that this software is closer to engineers' expectations of good software than the packages they currently use. SMTK offers appropriate decision support that enables engineers to perform more tasks in a more satisfactory manner than is currently possible. Future work will involve the application of ECI to the tasks shown in the ECI organizational schema.

ACKNOWLEDGMENTS

The authors would like to thank David Brown, Simon Bailey, Esther Gelle, Chimay Anumba, and Paul Xirouchakis for their comments and feedback on this research. This work was performed while Ruth Stalker was at the Institute of Structural Engi-

neering and Mechanics, EPFL. Logitech SA and Silicon Graphics Inc. are also thanked for their support of this work.

REFERENCES

Anumba, C.J. (1994). Considerations in user-interface design for knowledge-based systems. In *Artificial Intelligence and Object Oriented Approaches for Structural Engineering* (Topping, B.H.V., & Papadrakakis, M., Eds.), pp. 47–51. Edinburgh: Civil-Comp Press.

Billington, D.P. (1995). *The Tower and the Bridge: The New Art of Structural Engineering*, New York: Basic Books.

Boulanger, S. (1997). *Preliminary bridge design navigation tool for novices*. PhD Thesis. Lausanne, Switzerland: Swiss Federal Institute of Technology.

Boulanger, S., & Smith, I. (1994). Knowledge representation for design tasks. *Knowledge-Based Systems 7*, 161–168.

Burr, A., Craig, B.A., Chung, C., So, T., Deveau, T., & Shirer, D. (1999). Programs for statistics and data analysis. *Computing in Science and Engineering 1*, 16–21.

CADFIX. (2000). Available on-line at http://www.fegs.co.uk/cae.html.

Dix, A., Finlay, J., Abowd, G., & Beale, R. (1998) *Human–Computer Interaction*, 2nd ed., Hemel Hempstead, UK: Prentice–Hall Europe.

Fenves, S.J. (1989). Expert systems: Expectations versus realities. *Proc. IABSE*, Bergamo, Italy, pp 1–19.

French, S. (1986). *Decision Theory: An Introduction to the Mathematics of Rationality*. Chichester, UK: Ellis Horwood.

Gaylord, C.N. (1990). *Structural Engineering Handbook*, 3rd ed. New York: McGraw–Hill.

Gero, J. (1990). Design prototypes: A knowledge representation schema for design. *AI Magazine 11*, 26–48.

Grierson, E.E. (1996). Information technology in civil and structural engineering design: Taking stock to 1996. In *Information Processing in Civil and Structural Engineering Design*, pp. 7–15. Edinburgh: Civil-Comp Press.

Inaudi, D. (1997). *Fiber optic sensor network for the monitoring of civil engineering structures*. PhD Thesis. Lausanne, Switzerland: Swiss Federal Institute of Technology.

McCormack, J. (1975). *Structural Analysis*, 3rd ed. New York: Intext Education.

Megson, T.H.G. (1987). *Strength of Materials for Civil Engineers*, 2nd ed. Newcastle-upon-Tyne, UK: Edward Arnold.

Navinchandra, D. (1991). Exploration and innovation in design: Towards a computational model. In *Symbolic Computation, Artificial Intelligence*. New York: Springer.

Perregaux, N. (1998). *Pont de la Lutrive N9. Equipement et Analyse du Comportement au Moyen du Systeme de Mesure a Fibres Optiques SOFO*, PhD Thesis. Lausanne, Switzerland: Swiss Federal Institute of Technology.

Preece, J., Rogers, Y., Sharp, H., Benyon, D., Holland, S., & Carey, T. (1997). *Human–Computer Interaction*. Reading, MA: Addison-Wesley.

Pressman, R.S. (1994). *Software Engineering: A Practitioner's Approach*, 3rd ed., European ed., London: McGraw–Hill.

Raphael, B., & Smith, I. (1998). Finding the right model for bridge diagnosis. In *Artificial Intelligence in Structural Engineering, Computer Science*, pp. 308–319. LNAI 1454. Heidelberg, Germany: Springer–Verlag.

Robert-Nicoud, Y., Raphael, B., & Smith, I. (2000). Decision support through multiple models and probabilistic search. *Proc. Construction Information Technology 2000*, Icelandic Building Research Institute, Reykjavik, pp. 765–779.

Salvaneschi, P., Cadei, M., & Lazzari, M. (1996). Applying AI to structural safety monitoring and evaluation, *IEEE Expert, Intelligent Systems and Their Applications 11*, 24–34.

Shea, K. (1997). *Essays of discrete structures: Purposeful design of grammatical structures by directed stochastic search*. PhD Thesis. Pittsburgh, PA: Carnegie–Mellon University.

Smith, I.F.C. (1996). Interactive design: Time to bite the bullet. In *Information Processing in Civil and Structural Engineering Design*, pp. 23–30. Edinburgh: Civil-Comp Press.

Smithers, T. (1998). Towards a knowledge level theory of design process. In *Artificial Intelligence in Design*, pp. 3–21. Dordrecht: Kluwer.

Stalker, R.A. (2000). *Engineer–computer interaction for structural monitoring*. PhD Thesis. Lausanne, Switzerland: Swiss Federal Institute of Technology.

Stroud, K.A. (1995). *Engineering Mathematics*, 4th ed. London: MacMillan.

Timoshenko, S.P., & Goodier, J.N. (1970). *Theory of Elasticity*, 3rd ed. New York: McGraw–Hill.

Vurpillot, S. (1999). *Analyse Automatisee des Systemes de Mesure de Deformation pour l'Auscultation des Structures*. PhD Thesis. Lausanne, Switzerland: Swiss Federal Institute of Technology.

Wegner, P. (1997) Why interaction is more powerful than algorithms. *Communications of the ACM 40*, 80–91.

Zeithaml, V.A., Parasuraman, A., & Berry, L.L. (1990) *Delivering Quality Service: Balancing Customer Perceptions and Expectations*. New York: Free Press.

Ruth Stalker is a lecturer in interactive systems for software engineering in the Computing Department at Lancaster University. She received a PhD from the Department of Civil Engineering, Swiss Federal Institute of Technology (EPFL), in 2000 and holds a BS (Hons) in Computing from Liverpool John Moore's University and a PGDip in Artificial Intelligence from Aberdeen University. Dr. Stalker's research interests include the application of HCI to disciplines such as structural engineering and architecture in order to provide theories about specialist user groups.

Ian F.C. Smith is the Head of the Applied Computing and Mechanics Laboratory (IMAC) and Director of the Structural Engineering Institute (ENAC), School of Architecture, Civil and Environmental Engineering, at the Swiss Federal Institute of Technology in Lausanne, Switzerland. Professor Smith's interests are in structural monitoring and ECI.