JTruss: A CAD-Oriented Educational Open-Source Software for Static Analysis of Truss-Type Structures

FRANCESCO ROMEO,[1] GIANLUCA PADOAN[2]

[1]Dipartimento di Ingegneria Strutturale e Geotecnica, Università di Roma "La Sapienza", Via Gramsci 53, 00197 Rome, Italy

[2]Prima Facoltà di Architettura "L. Quaroni", Università di Roma "La Sapienza", Via Gianturco 2, 00196 Rome, Italy

Received 9 November 2005; accepted 16 November 2006

ABSTRACT: A CAD-oriented software (JTruss) for the static analysis of planar and spatial truss-type structures is presented. Developed for educational purposes, JTruss is part of an open-source project and is characterised by complete accessibility (i.e. platform independent) and high software compatibility. CAD methodologies are employed to implement commands for handling graphic models. A student friendly graphical interface, tailored mainly for structural mechanics introductory courses in engineering and architecture programs, is conceived. Accordingly, the standard sequence involved in the software design, namely pre-processing, processing and post-processing, is implemented aiming to improve the structural behaviour interpretation. ©2008 Wiley Periodicals, Inc. Comput Appl Eng Educ; Published online in Wiley InterScience (www.interscience.wiley.com); DOI 10.1002/cae.20150

Keywords: CAD software; open-source; truss-type structures

INTRODUCTION

The software presented in this work, JTruss, is a structural analysis CAD developed for educational purposes in the engineering and architecture areas. It is the core of an Open-Source educational project where accessibility, portability as well as simplicity have been the primary requirements (Fig. 1).

Correspondence to F. Romeo (francesco.romeo@uniroma1.it).

Open Source represents a new way of conceiving the development and distribution of software [1]: anyone can access the source code, point out and fix bugs, give suggestions for improvements as well as propose new features. Few basic rules govern the rights of Open Source software users, namely: the user has free access to the original and complete software source code; the user can modify the software even to create new software; the user can make copies of the original software and distribute them and the Open Source license is extended to the new software. The

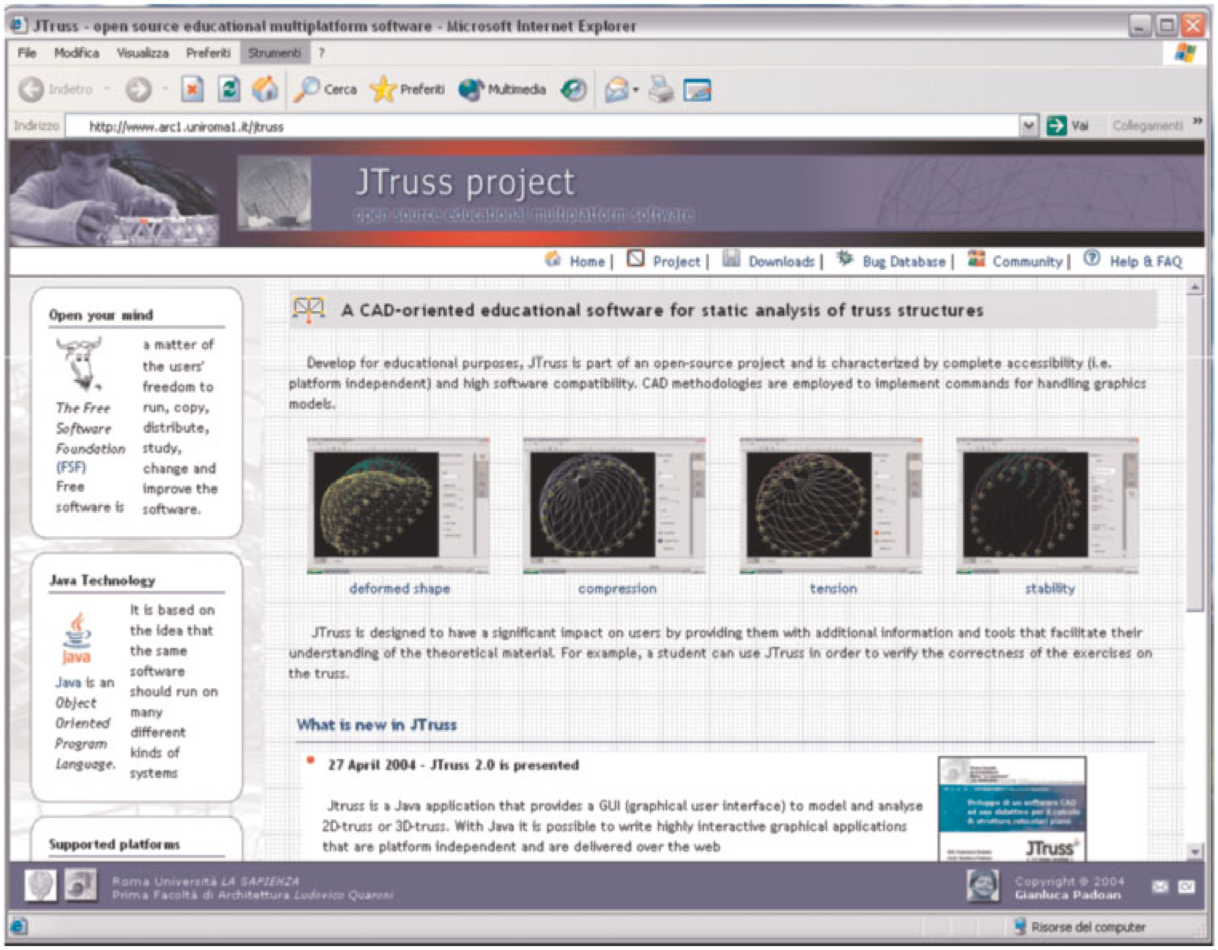

Figure 1 JTruss open-source project: the web site (http://151.100.30.59/jtruss/). [Color figure can be viewed in the online issue, which is available at www.interscience.wiley.com.]

GPL (General Public License GNU) is the most effective and universal contract for free software. It must be emphasised that the Open Source policy arised naturally in the academic environment where the free sharing of information is crucial for the development of science and technology in both research and education contexts. The main advantages can be summarised according to three key points. Open Source softwares and the relevant upgrades are usually free; the interaction among computer programmers coming from all over the world enables a multi-cultural development; the free access to both source and executable codes enhances the software reliability since users can check how the software acts.

The Java programming language has been adopted [2,3]. As well known, when running a program, Java relies on a virtual machine (JVM), similar to a real processor, that executes the bytecodes generated by compiling the source files. Therefore Java is a platform-independent language (Fig. 2). Moreover, Java itself is a free programming language whose development takes advantage from the potential of the Open Source community.

The paper is organised as follows. The Software Design and Development Section highlights how the structural Computer Aided Design logic is implemented according to a Model-View-Controller design pattern. In JTruss Features Section the main features of the software JTruss are presented in terms of pre-processing, processing and post-processing capabilities. JTruss performs the static analysis of planar and spatial truss-type structures [4,5] which play a key role in introductory structural mechanics courses. As a matter of fact, such class of structures is ideal for introducing the students to the main concepts and techniques for solving linear elasticity problems due to the simple algebraic nature of the operators involved in the relevant mathematical formulation [6].

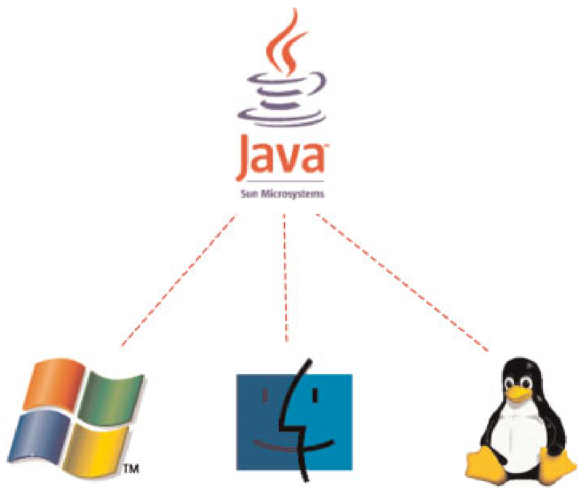

Figure 2 JTruss: a platform-independent software. [Color figure can be viewed in the online issue, which is available at www.interscience.wiley.com.]

SOFTWARE DESIGN AND DEVELOPMENT

As well known the acronym CAD stands for Computer Aided Design and it refers to a software able to compute, store and graphically represent models of object defined by the designer. By integrating the model with a set of information referred not only to the geometry but, for instance, also to the physical properties, it is possible to analyse the real behaviour of the object together with its characteristics such as stresses, environmental impacts, etc. CAD softwares are usually equipped with a graphic pre-processor that allows to draw the model; while using CAD descriptive geometry rules are relied upon, CAD development is mainly grounded on analytic geometry rules governing the representation and the transformation of spatial objects. Methods and strategies to input geometrical data are defined by the CAD theme. In the architectural and civil engineering context three CAD types can be identified: design graphic representation, modelling and animation and structural analysis. While in each CAD type the preprocessing module obeys, in the large, the same rules, the structural analysis CAD is characterised by a peculiar feature of the basic elements: behind a line on the monitor a structural element is hidden. Boundary conditions, such as constraints, complete the model description and the post-processing module provides with the analysis output such as stresses, strains, stability analysis, etc. The CAD logic is based on vectorial images manipulation. The computer representation of a vectorial image is governed by the Euclidean space rules so that, given a Cartesian reference frame, the location of a point is uniquely identified by its coordinates x, y and z.

The Object Oriented Programming (OOP) logic, characterising most of the modern software, is also a key component of JTruss. According to the OOP philosophy [7], the software goal is not merely to execute a sequence of instructions, but to relate a set of objects relying on design patterns providing with solution strategies independent of the particular problem at hand. The widely used Model-View-Controller design pattern allows to design independently each software component and here is adapted for structural CADs.

Model

In JTruss truss-type structures are analysed and, in the first step (Model), their object oriented translation is required. This operation is carried out along the line prompted by the underlying *mechanical* and *mathematical* models. As far as the mechanical model, truss-type structures can be viewed as prototype of discrete mechanical systems as they can be described by finite number of elements such as joints, bars and constraints. The joints, considered as punctual rigid bodies, possess merely translational degrees of freedom; the bars, considered as simple internal constraints, behave like translational springs; the external constraints can be either simple supports or hinges. The external forces can only be applied to the joints and, consequently, the bars can either undergo tension or compression. It follows that the joints' characteristics are: the positioning vector with respect to a Cartesian reference frame, the displacement constraint flag in each direction, the nodal force vector and the nodal displacement vector. Moreover, the bars' characteristics consist of: the ends' node indexes, defining the position of the bar in space, the cross-section, the material properties, the normal stresses. As far as the mathematical model, the algebraic problem consists of a set of linear equations whose dimension depends on the number of joints, bars and restrained degrees of freedom entering the model. In order to implement the object oriented model of truss-type structures, the above mentioned elements are organised in classes. The class diagram in Figure 3 shows the model representation according to the Unified Modeling Language (UML) standard. Thus, the truss structure is composed by generic elements that specialise their role becoming joints, bars, constraints and external loads. The union of these elements form the Truss Model where each element has its own properties defining their behaviour (methods). The node is the core of the model:

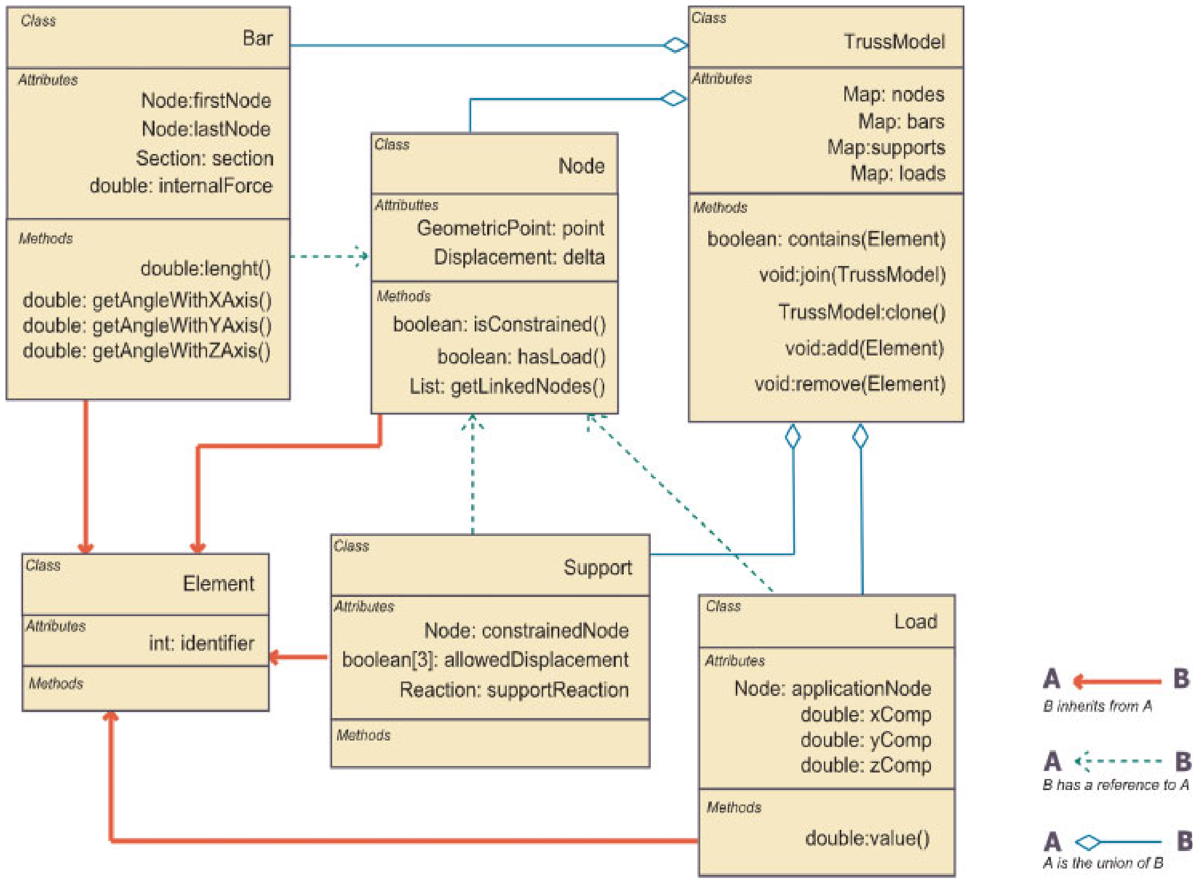

Figure 3 Object oriented model of truss-type structures: class diagram. [Color figure can be viewed in the online issue, which is available at www.interscience.wiley.com.]

each bar, constraint or load can indeed only exist if referred to a node.

Controller

Object oriented implementation of the solution of the linear elastic problem according to the displacement method: sequence diagram. In the Controller stage, JTruss solves the linear elastic problem for truss-type structures according to the direct formulation of the displacement method. The input data consist of: geometry of the structure, constraints, nodal forces and the material and geometrical properties of the bars. The displacement method consists of seeking the unique equilibrium configuration among the infinite kinematically compatible ones. Therefore equilibrium equations expressed in terms of generalised displacement provide with the solving set of equations. Once determined the displacements, bars' stresses and strains and constraints' reactions can be obtained. The sequence diagram in Figure 4 shows the implementation of the controller class TrussSolver. The controller reads the numerical model in the object TrussModel. At first, given a model made up of m bars and n joints, the geometrical and physical data of the bars are processed to build the $m \times n$ compatibility matrix **D** and $m \times m$ elastic matrix **C**, respectively. Then, by introducing the boundary conditions, the number of unknowns is reduced by the number of restrained degrees of freedom. The controller assembles the stiffness matrix **K** and, once the nodal forces vector is cast, the linear system of equations is solved. The results are eventually brought back to the TrussModel in the sequence: nodal displacements, bars strains, stresses and external reactions.

View

The graphic representation of the 2-D or 3-D model on the monitor is the task accomplished in the View step. In other words, the Views are the objects manage the visual representation of the model data by switching from a *real* reference frame to the monitor *bitmap* one. The diagram in Figure 5 shows, for instance, how the class describing the bar properties is specialised for 2-D and 3-D visualisation; such OOP

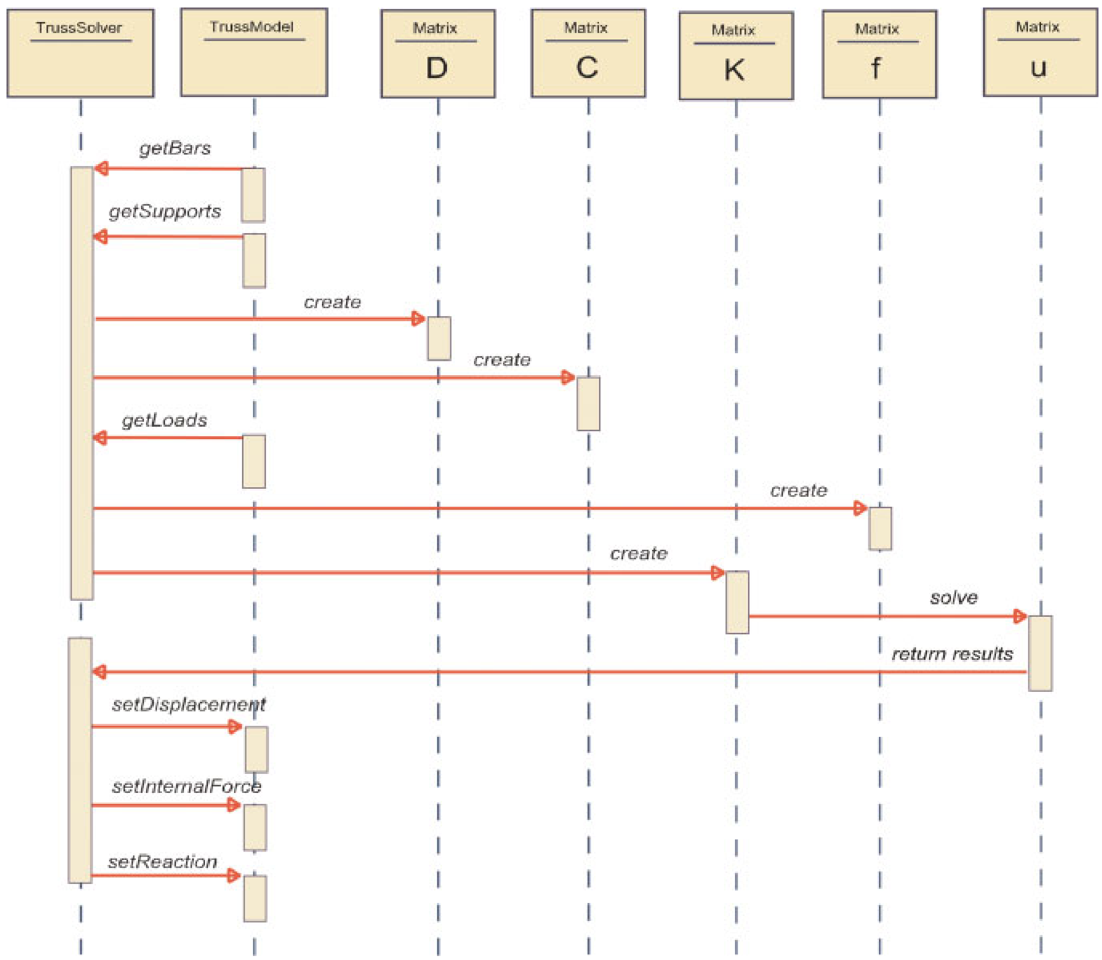

Figure 4 Object oriented implementation of the solution of the linear elastic problem according to the displacement method: sequence diagram. [Color figure can be viewed in the online issue, which is available at www.interscience.wiley.com.]

technique is referred to as inheritance. The specialised class, that inherits the behaviour and all the properties of the *father* class, entails the reference to the Cartesian coordinates in use: such object manages the transformation from the real coordinates to the monitor ones. All the elements entering the modelled structure implement the interface called Draw that leads the programmer to define how to visualise the object on the monitor. Thus, each element composing the structure is no longer interpreted as a node, bar, constraint, etc., but it is viewed as an object to be drawn. The Graphical User Interface (GUI) is a key component of the View objects. As well known, it allows the user to control the software by acting through an input device (e.g. mouse, keyboard) on signs/symbols on the monitor. In the CAD herein proposed, the central area of the screen, called ViewPort, is the space devoted to the numerical model graphical representation. The GUI is composed by two modules, namely the pre-processor (model) and the post-processor (solution). The first provides with a number of tools to draw and modify the truss structure, whilst the latter displays the model solution.

JTruss FEATURES

In this section the main features of JTruss are presented according to the usual steps involved in the logical sequence of structural softwares, namely: pre-processing, processing and post-processing. Great care has been devoted to the graphical user interface development in order to assure the software *usability*. Thus, the commands are implemented through either standard CAD environment toolbars and a large number of expressly designed icons.

Pre-Processing

As far as the pre-processing module is concerned, the model input is optimised in order to ease the definition

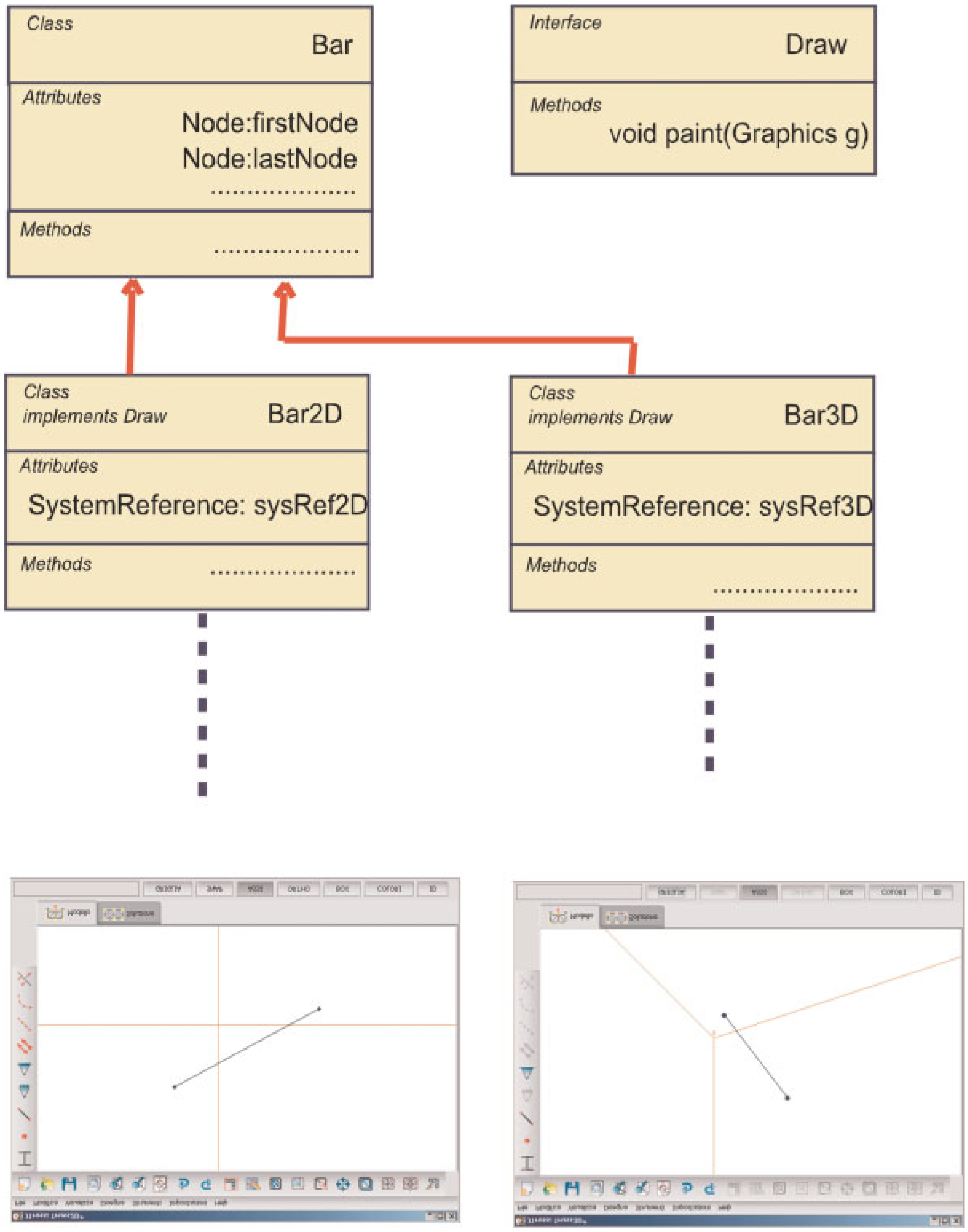

Figure 5 Model graphic representation. [Color figure can be viewed in the online issue, which is available at www.interscience.wiley.com.]

of both geometrical and material properties. The model can be cast by relying on a number of criteria for the automatic generation and projection of joints as well as on the arbitrary orientation of joint constraints. Bars' cross-section and material properties can be either defined by the user or selected among a number of pre-defined choices (Fig. 6a). User defined blocks can be created, stored and inserted in the current model (Fig. 6b). Moreover, by editing elements and/or blocks (copy, multiple copy, scale, mirror, divide, join), arbitrary modular structures can be conveniently assembled. Finally, 2-D and 3-D models can be either created with JTruss or imported from ED-Tridim [8] and AutoCAD^{TM} DXF^{TM} [9] formats.

Processing

As mentioned in Software Design and Development Section, JTruss solves the linear elastic problem for truss-type structures according to the direct formulation of the displacement method. In this realm detailed

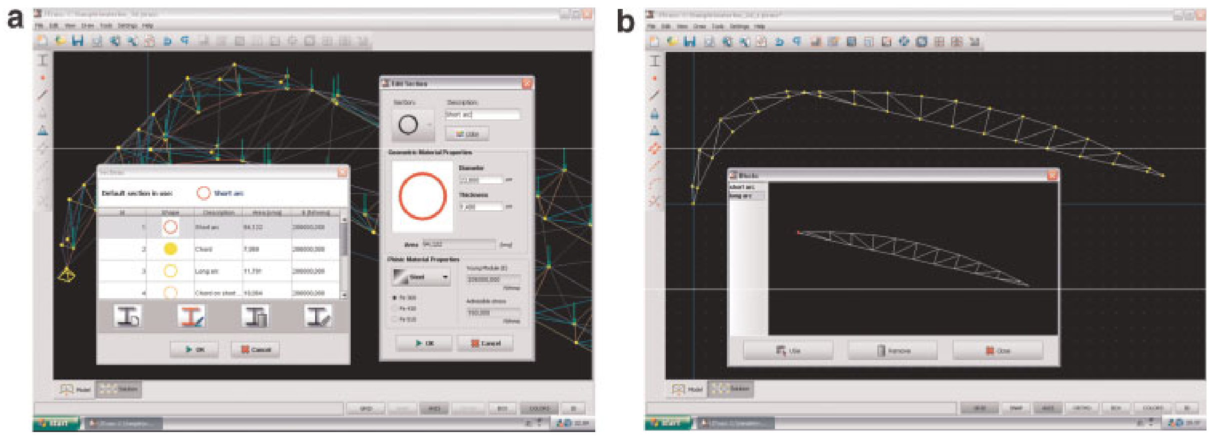

Figure 6 Pre-processing features: (a) bars' material properties; (b) blocks. [Color figure can be viewed in the online issue, which is available at www.interscience.wiley.com.]

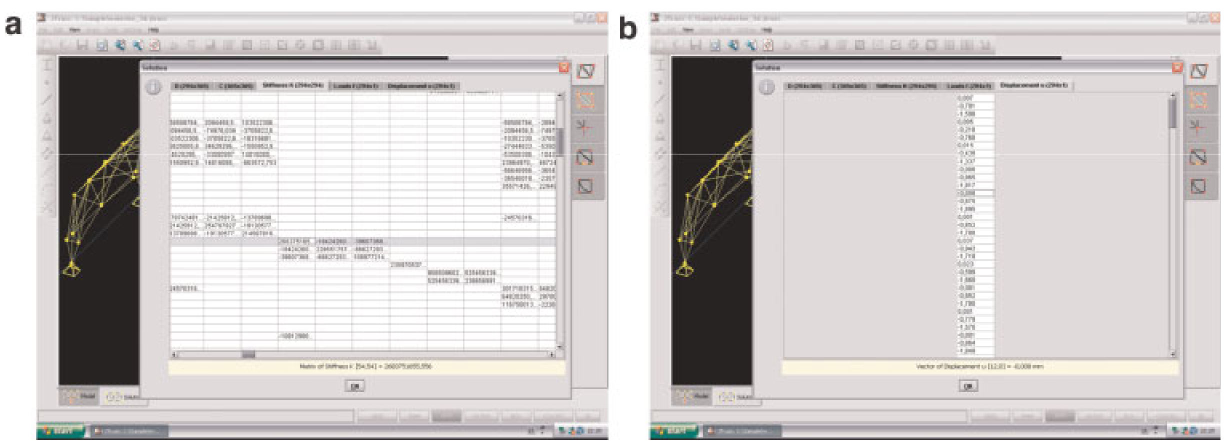

Figure 7 Processing features: (a) stiffness matrix; (b) displacement vector. [Color figure can be viewed in the online issue, which is available at www.interscience.wiley.com.]

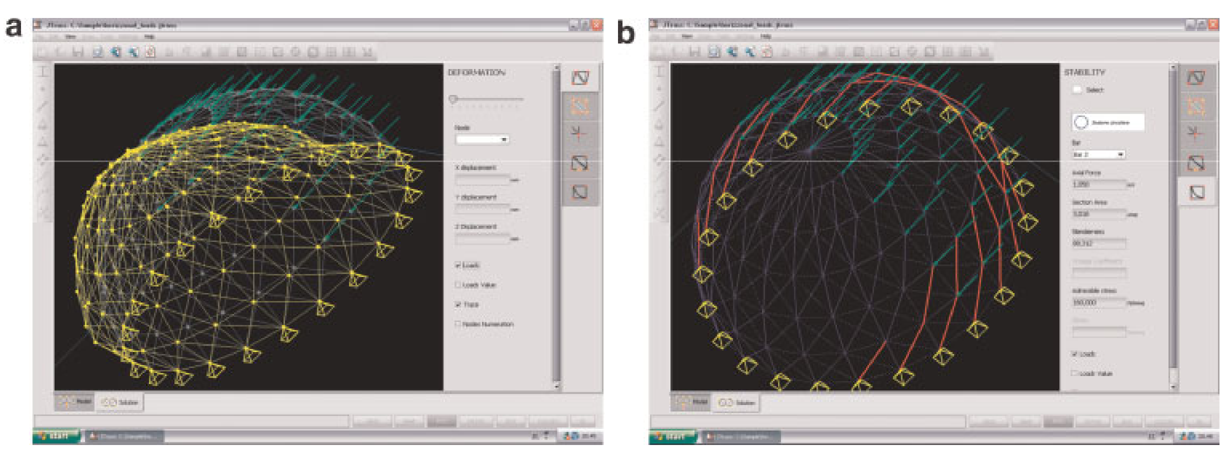

Figure 8 Post-processing features: (a) deformed shape; (b) stability analysis. [Color figure can be viewed in the online issue, which is available at www.interscience.wiley.com.]

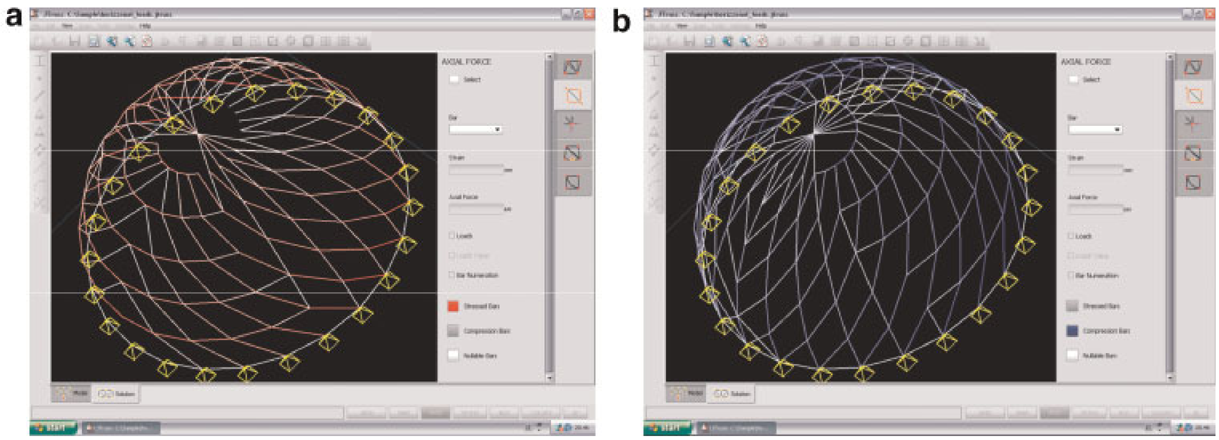

Figure 9 Post-processing features: (a) bars under tension; (b) bars under compression. [Color figure can be viewed in the online issue, which is available at www.interscience.wiley.com.]

information, such as the compatibility, elastic and stiffness matrices, as well as the displacement and force vectors, can be displayed (Fig. 7a,b) allowing the users to follow the solution process.

Post-Processing

In the post-processing stage the solution of the elastic problem is provided both in graphic and text form. In particular, the graphic output is divided according to the main aspects of the solution, namely: kinematic, static, stability analysis and design check. From the kinematic point of view the deformed shape can be visualised and amplified together with bars' strain and joints' displacement (Fig. 8a). From the static side, the intensity of bars' stress is associated to the intensity of the bars' colour providing with a straightforward interpretation of the mechanical behaviour of the structure in terms of bars' tension (Fig. 9a) and compression (Fig. 9b). Information about the possible failure of the bars are also graphically provided with respect to both bars' buckling (Fig. 8b) and strength.

CONCLUSIONS

The initial step of an educational project for engineering and architecture students, consisting of the CAD-oriented software JTruss, for the static analysis of planar and spatial truss-type structures, has been described. The main aspects of the underlying educational approach, namely the development of software characterised by complete accessibility and compatibility, have been pursued relying on both the Open-Source philosophy and the Java programming language. It has been shown that the OOP logic through the common Model-View-Controller design pattern is well suited for the design of structural CAD software. A student friendly graphical interface has been conceived aiming to improve the students involvement in all the stages of the structural analysis. Future developments of JTruss will address the dynamic structural analysis and more complex structural models such as frames and shells. In view of the latter upgrades, the present work must be interpreted as a first effort towards the realisation of a broader educational environment whose development stems from the interaction among the users, namely students and academic staff.

REFERENCES

[1] A. MacKenzie, OPEN SOURCE SOFTWARE: When is a Tool? What is a Commodity? Sci Cult 10 (2001), 541–552.
[2] Sun Microsystems, http://java.sun.com.
[3] A. E. Walsh, and D. Gehringer, Java 3D, Prentice-Hall, Upper Saddle River, NJ, 2002.
[4] R. B. Füller, Ideas & integrities, Prentice-Hall, Englewood Cliffs, NJ, 1963.
[5] J. Chilton, Atlante delle strutture reticolari, UTET, Torino, Italy, 2002.
[6] A. Luongo, and A. Paolone, Meccanica delle strutture, Masson, 1997.
[7] C. Alexander, A pattern language, Oxford University Press, New York, 1977.
[8] B. Suarez, E. Blanco, L. Gil, P. Zapata, and E. Onate, ED-Elas2D: An educational program for computer-aided training in structural analysis using the finite element method, Eur J Eng Educ 23 (1998), 243–254.
[9] AutoCAD. Autodesk, Inc.

BIOGRAPHIES

Francesco Romeo received the PhD in civil engineering from the University of Rome 'La Sapienza' and, in 1998, the MS in engineering from The Johns Hopkins University. He was a research associate in the Department of Structural, Geotechnical and Hydraulic Engineering, University of L'Aquila, from 1999 to 2001. He is currently an associate professor in the Department of Structural and Geotechnical Engineering, University of Rome 'La Sapienza'. He has taught the courses Statics, Structural Mechanics, and Mechanics of Solids. His main research interests are identification and control of structural systems, dynamics of linear and nonlinear periodic structures, and dynamics of parametrically excited structures.

Gianluca Padoan graduated cum laude in architecture from the University of Rome 'La Sapienza' in 2003. Since 2000 he has been a designer and developer of J2EE applications at IBM, where he is currently employed as IT architect technical leader. He is a Sun Ceritfied Programmer for Java 2 Platform 1.4 and a Sun Certified Web Component Developer for Java 2 Enterprise Edition 1.4.