Portable implementation of computer aided design environment for composite structures

Santosh Kumar Pagadala E.
West Virginia University

PORTABLE IMPLEMENTATION OF COMPUTER AIDED DESIGN ENVIRONMENT FOR COMPOSITE STRUCTURES

Santosh Kumar Pagadala E.

Thesis submitted to the College of Engineering and Mineral Resources at West Virginia University in partial fulfillment of the requirements for the degree of

Master of Science
In
Mechanical Engineering

Department of Mechanical and Aerospace Engineering
Morgantown, West Virginia
2004

Keywords: Micromechanics, Macromechanics, Beams, GUI.

ABSTRACT

PORTABLE IMPLEMENTATION OF COMPUTER AIDED DESIGN ENVIRONMENT FOR COMPOSITE STRUCTURES

Santosh Kumar Pagadala E.

Composite materials are widely used due to their low weight, long durability and the ability to tailor their properties to specific design requirements. Their wide range of applications requires a solid understanding of their behavior under different load conditions. The calculations needed for efficient and accurate design of composites could be exhaustive and time consuming. Therefore an efficient computer program which would facilitate effective design becomes a key factor to supporting commercial use of composite materials.

A portable software tool was developed in Java programming environment for design analysis of composite materials. This software tool is superior to its predecessor, the Computer Aided Design Environment for Composite software (CADEC). The Java software is an exact replica of the CADEC software, which is written in Toolbook. The only major difference is that the new program is rewritten in a portable language. The software imposes no restrictions on the accessibility or the usability of this tool. The software can be accessed via internet and it can be run on any operation system as long as a Java enabled browser is available. The software tool has been evaluated using the example problems taken from the text book “Introduction to Composite Materials Design” written by Dr. Ever J. Barbero. The results obtained by using the new tool are sufficiently close to the text book results.

Dedicated to my Mom

Mrs. Saraswathi Pagadala

---

because she taught me the childhood I had,

because she gave me the knowledge I treasure,

because she gave me the happiness I cherish,

because she gave me the strength I possess,

because she gave me the life I live.

I love you Mom.

ACKNOWLEDGEMENTS

I would like to thank my father Mr. Eshwar Dass Pagadala, my sister Shalu and my brother Shiva (Simon), whose love, affection and encouragement has given me the strength and ability to achieve the goals which I strive and be the best to reach my dreams.

It is a particular pleasure to recognize the involvement of Dr. Ever J. Barbero as my advisor and committee chair. Personally, I would like to thank him for providing me this opportunity and for his support and guidance without which, my thesis would not have come to a good end.

I am highly indebted to Dr. Jacky C. Prucz, and Dr. James D. Mooney for being on my committee. I really appreciate the time and guidance they provided.

Lastly I would like to thank my friends Chaitu, Suri, Vinay, Sreeman, Kalyan, Suresh, Vijay, Kaparthi, Partha, Rekh, Narsimha and Baldwin guys for their love and support. You all have a special place in my heart.

TABLE OF CONTENTS

TABLE OF FIGURES

Chapter 1. Introduction

The world where we live in is developing and each day brings new inventions into the world. Some of these inventions are good and some of them are bad. But most of these inventions were started with the sole purpose of making the human life process easier. Today world is a sophisticated place where things are done much quicker, much faster and more accurately. Technology allowed man to land on the moon. Today he succeeded in sending a machine to Mars. Technology made it possible to travel farther distances in lesser time. It would have taken months, years to travel to another country but today it takes hours and at most days. Technology includes software (programs) which are cheaper, portable, secure and efficient. One such technology is Internet. Today internet is used for almost everything, It is used for research, money transferring (electronic transfer), banking, buying and exchanging stocks, emailing etc. There are certain softwares, which make these things possible on the internet. One such language which provides some of these features is java.

Portability can be defined as the ability to transfer an object or software from one system to another with minimal or no changes. Java with its portable feature is gaining popularity and is widely used these days. Another reason why Java is widely used is because it is freely available on the web and can be easily downloaded, installed and executed. Programs in Java are easy to develop and maintain when compared to other programming languages. Java has an built in capability to run on a variety of computers and operating systems with minimal code changes.

Java consists of small programs called applets. Applets are programs that can be embedded in Internet Web pages and are executed under the control of the web browser on the client side. Java also consists of Frames, which can be converted into applets. The Java programs will run just as well on a PC running Windows 95/98/NT/2000/XP as it will on Linux or a Sun Solaris workstation [1]. It would execute using the **Java Virtual Machine** provided by the browser. Java also provides packages like Remote Method Invocations (RMI) using, so that client and the server are able to communicate and exchange data between them.

Java with all these features is used to create applications in various fields. In this Thesis, a portable application is created using Applet and RMI for analysis of composite materials. Frames are created using in Java, and these frames provide the Graphical User Interface (GUI) part for the program. Later these frames are converted into applets so that they can be viewed through the Internet Web pages.

Composite materials are widely used these days because of their light weight, low cost and long durability. Another equally important reason is that, what we learn from the field of composites could have far-reaching implications in many fields of science [2]. Composite materials get their attributes from their constituent materials. Since composites have wide range of applications these days, modeling their behavior becomes a key factor. Their behavior under different load conditions aids the designer to create more accurate and efficient products. Analysis involves calculations that can be lengthy and time consuming. A small error would result in recalculation, which often is time consuming. This is where the need for a computer program becomes a necessity. This Thesis deals with the design of software for design composites, using the Java program. Since Java is portable and its application can be embedded inside the browser as an applet, the current program will be accessible to all the users through the World Wide Web.

Chapter 2. Literature Review

2.1. Computer Applications:

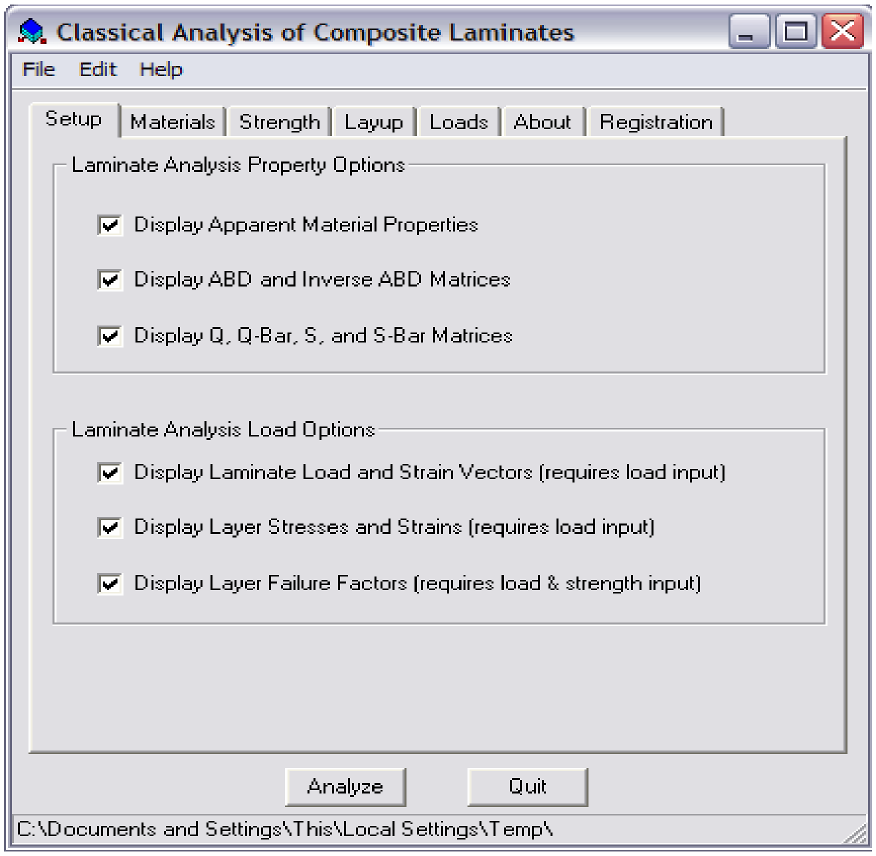
Today computers have become a house hold name. What started as a solution to solve big and complex numerical problems, eventually became the part of the main stream, have become a part of the daily life. Computers can be used for anything and everything as long as they can be programmed to do so. Computers once programmed to do a specific task can easily replace a human doing the same task for its efficient, error free, fast and reliable. All these and many such aspects have helped computers to be part of all fields of science and technology, mechanical being no exception.

Composite materials can be defined as a mixture of fiber and matrix. The behavior of the composite depends on the properties of the constituent materials. Thus the design of composite material becomes important. A computer program that does the analysis of composite material would really prove beneficial. It can provide the flexibility of changing the composition and study its behavioral aspects. However, very few software applications have been developed for composite materials. The following is a literature review of available software.

2.2. Classical Analysis of Composite Materials:

Also called as a Laminator, this program is being written by Mr. Michael Lindell. The program was basically developed in C++ language using Power++ Developer. Laminator is a shareware engineering program that analyzes laminated composites plates according to classical laminated plate theory [3].

The program uses an interface with a tabbed pane. The user chooses which page to view by selecting the tab corresponding to the desired page. The laminator also has a provision for check boxes those when checked will enable the user to view the result page corresponding on those options. The screen capture of the main page of “Classical Analysis of Composite Laminates” is shown in the Figure 2.1 below.



Laminate Analysis Property Options

- [✓] Display Apparent Material Properties
- [✓] Display ABD and Inverse ABD Matrices
- [✓] Display Q, Q-Bar, S, and S-Bar Matrices

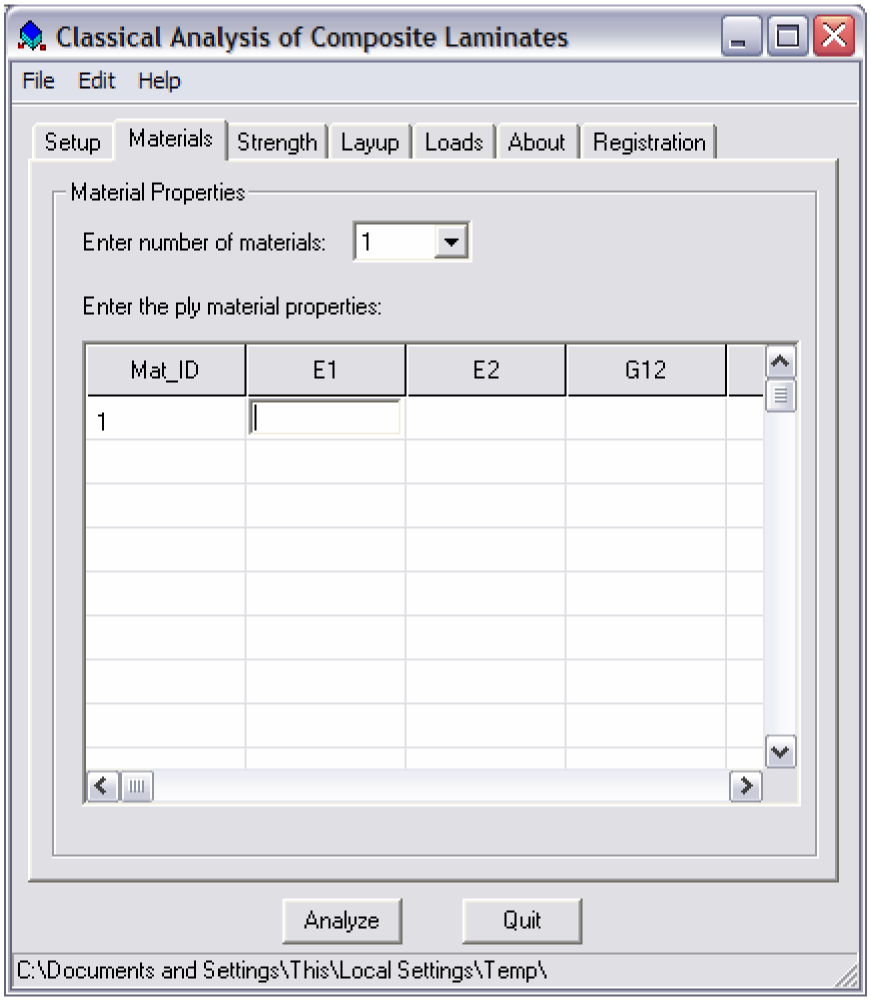
Laminate Analysis Load Options

- [✓] Display Laminate Load and Strain Vectors (requires load input)
- [✓] Display Layer Stresses and Strains (requires load input)
- [✓] Display Layer Failure Factors (requires load & strength input)

C:\Documents and Settings\This\Local Settings\Temp\

Figure 2.1 Graphical User Interface showing different options for Selection [4].

The user has the option to enter the values of ply material properties, material strengths (to determine failure factors), ply fiber orientation and stacking sequence, mechanical loads, strains, temperature and moisture loads. The Figure 2.2 shown below is a screen capture of the program that shows a page where a user can enter the ply material properties.


Classical Analysis of Composite Laminates
File Edit Help
Setup Materials Strength Layup Loads About Registration
Analyze
Quit

Material Properties

Enter number of materials: 1

Enter the ply material properties:

Mat_ID	E1	E2	G12
1			

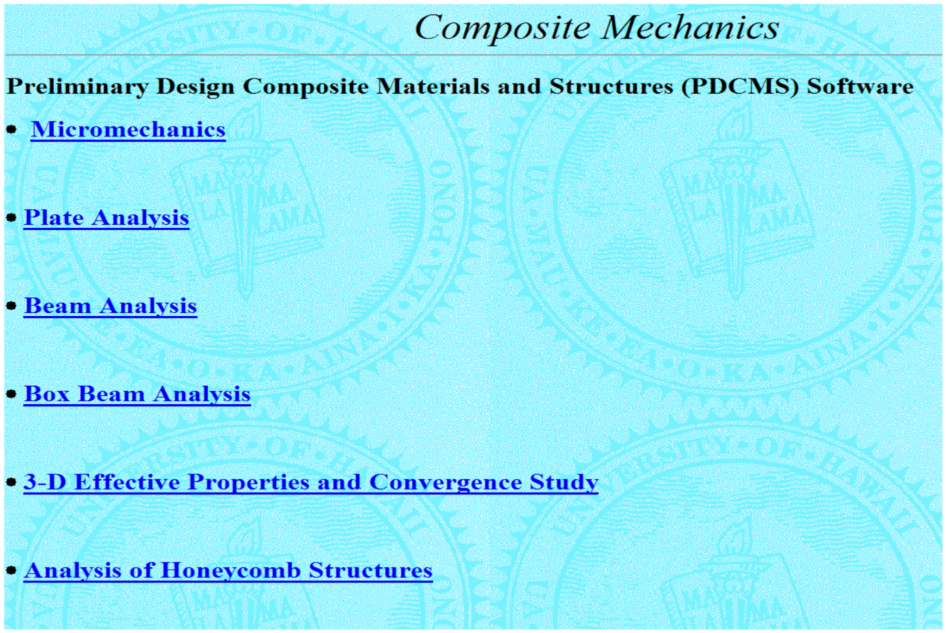
C:\Documents and Settings\This\Local Settings\Temp\

Figure 2.2 Graphical User Interface to enter ply material properties [4].

Once all the values are entered, clicking the analyze button will execute the program and the output file is created for the user to view the results. This software is paid software and needs to be downloaded and installed on the User/Client machines.

2.3. Preliminary Design Composite Materials and Structures Software (PDCMS):

This is an integrated software, written by Dr.Mehrdad Ghasemi Nejhad and Mr. Aniruddha Pant of University of Hawaii at Manoa, which links composite materials database, micromechanics, plates, beams, shells, etc. modules. To invoke any of the analysis modules, such as plate, the user needs to click on that module [4]. A part of the front page capture is shown below.


Composite Mechanics

Preliminary Design Composite Materials and Structures (PDCMS) Software

- Micromechanics
- Plate Analysis
- Beam Analysis
- Box Beam Analysis
- 3-D Effective Properties and Convergence Study
- Analysis of Honeycomb Structures

Figure 2.3 Graphical User Interface giving user the option to select module [4]

Throughout the software the values used are in either SI units or British units. The software accepts the values from the user in an interactive mode and all the simulations by various modules are performed online and in real-time. The outputs are available in both numerical and graphical forms [4]. This is an online application and is freely accessible through the web.

2.4. Automated System for Composite Analysis (ASCA):

ASCA is Microsoft Windows^{TM} based software for analysis and design of composite materials [5]. Figure 2.4 shows the main page of the program.

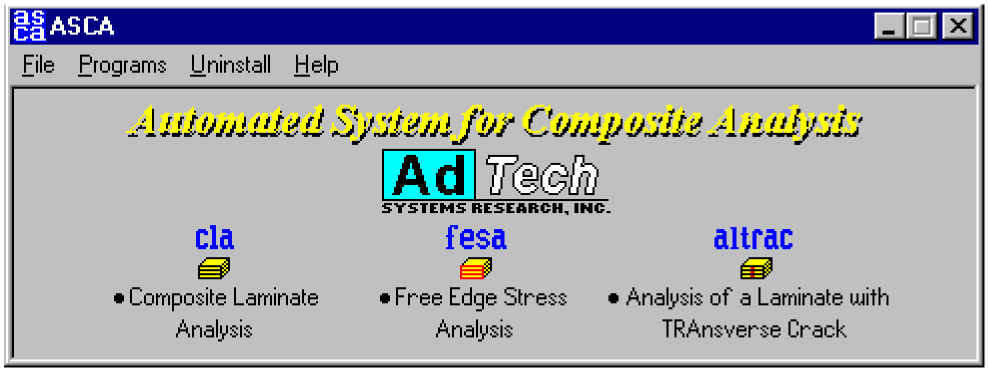
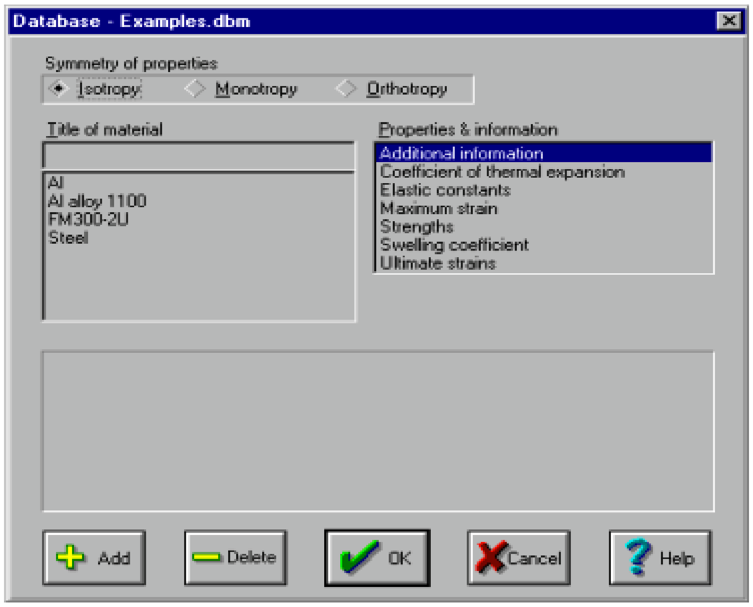

ASCA
File
Programs
Uninstall
Help
Automated System for Composite Analysis
Ad Tech
SYSTEMS RESEARCH, INC.
cla
• Composite Laminate Analysis
fesa
• Free Edge Stress Analysis
altrac
• Analysis of a Laminate with TRAnsverse Crack

Figure 2.4 Main Page of ASCA [5]

The main window shows three modules that the user can select and analyze. The software is designed as a menu-driven software allowing one to open existing work files or create new ones, define the laminate, select material(s) of different layers, select a strength predicting criterion, compute results and display them in convenient tabular or graphical form [5]. The software also has the feature of saving data of both metallic and non metallic materials. The screen capture for an example is shown in Figure 2.5.



Selected: [✓ Isotropy] Monotropy Orthotropy

Title of material:

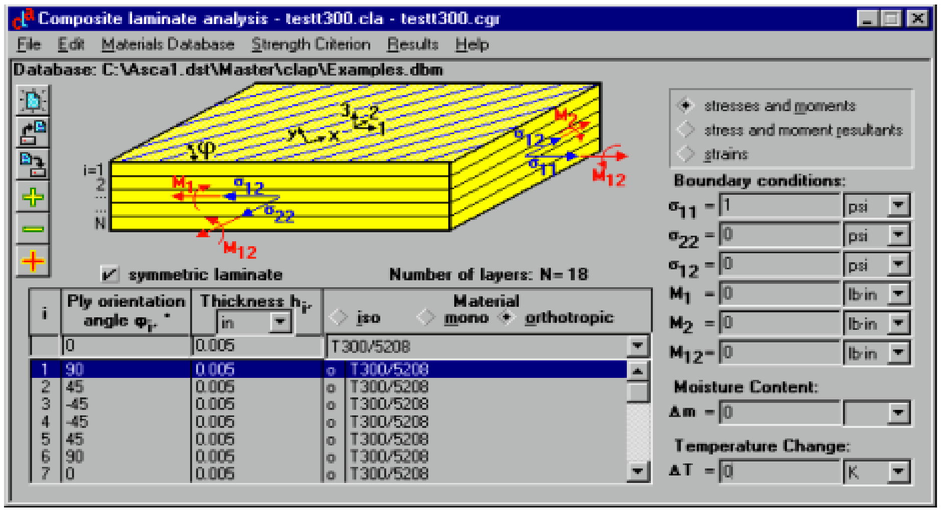
Al
Al alloy 1100
FM300-2U
Steel

Properties & information:

Additional information
Coefficient of thermal expansion
Elastic constants
Maximum strain
Strengths
Swelling coefficient
Ultimate strains

Figure 2.5: Materials database dialog box [5]

The properties required to define the composite such as material definition, material selection, strength criterion selection, boundary conditions, temperature changes and ply values updates can be entered inside the Complete Laminate Analysis window. Figure 2.6 below shows the window.



Database: C:\Asca1.dst\Master\clap\Examples.dbm

[✓ stresses and moments] stress and moment resultants strains

[✓] symmetric laminate

Number of layers: N= 18

i	Ply orientation angle φ_i, °	Thickness h_i, in	Material: iso mono [✓ orthotropic]
	0	0.005	T300/5208
1	90	0.005	o T300/5208
2	45	0.005	o T300/5208
3	-45	0.005	o T300/5208
4	-45	0.005	o T300/5208
5	45	0.005	o T300/5208
6	90	0.005	o T300/5208
7	0	0.005	o T300/5208

Boundary conditions:

σ_{11} =	1	psi
σ_{22} =	0	psi
σ_{12} =	0	psi
M_1 =	0	lb·in
M_2 =	0	lb·in
M_{12} =	0	lb·in

Moisture Content:

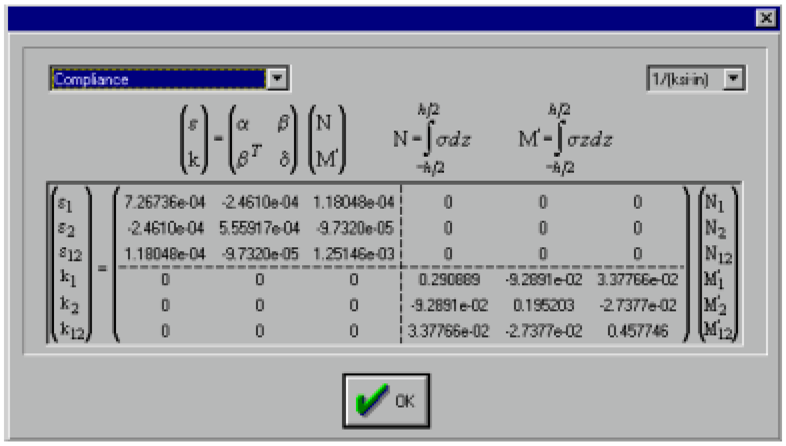
Δm = 0

Temperature Change:

ΔT = 0 K

Figure 2.6 Complete Laminate Analysis [5]

The computed results using this software can be viewed in tabular form, multiple x-y graphs and in ASCII text version. The tabular formatted are in the matrix form for compliance matrices for a composite lay-up [5]. The color-coded graphs that corresponds to the color–coded bar can be used add or delete the particular laminate layer. The two results GUI pages are shown in Figure 2.7 and Figure. 2.8. This is a commercial code.

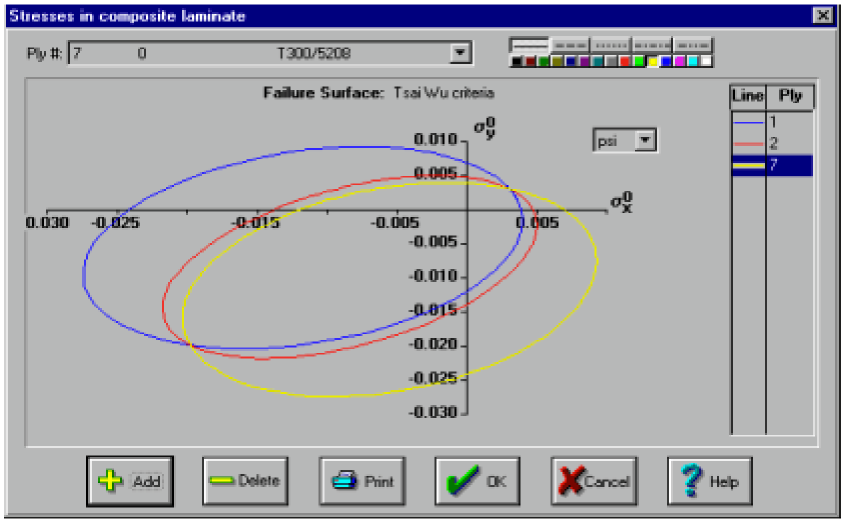


Compliance ▾ 1/(ksi·in) ▾

$$\begin{pmatrix} \varepsilon \\ k \end{pmatrix} = \begin{pmatrix} \alpha & \beta \\ \beta^T & \delta \end{pmatrix} \begin{pmatrix} N \\ M' \end{pmatrix} \qquad N = \int_{-h/2}^{h/2} \sigma dz \qquad M' = \int_{-h/2}^{h/2} \sigma z dz$$

$$\begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \varepsilon_{12} \\ k_1 \\ k_2 \\ k_{12} \end{pmatrix} = \left(\begin{array}{ccc:ccc} 7.26736e\text{-}04 & -2.4610e\text{-}04 & 1.18048e\text{-}04 & 0 & 0 & 0 \\ -2.4610e\text{-}04 & 5.55917e\text{-}04 & -9.7320e\text{-}05 & 0 & 0 & 0 \\ 1.18048e\text{-}04 & -9.7320e\text{-}05 & 1.25146e\text{-}03 & 0 & 0 & 0 \\ \hdashline 0 & 0 & 0 & 0.290889 & -9.2891e\text{-}02 & 3.37766e\text{-}02 \\ 0 & 0 & 0 & -9.2891e\text{-}02 & 0.195203 & -2.7377e\text{-}02 \\ 0 & 0 & 0 & 3.37766e\text{-}02 & -2.7377e\text{-}02 & 0.457746 \end{array}\right) \begin{pmatrix} N_1 \\ N_2 \\ N_{12} \\ M'_1 \\ M'_2 \\ M'_{12} \end{pmatrix}$$

Figure 2.7 Results table dialog box. [5]



Stresses in composite laminate

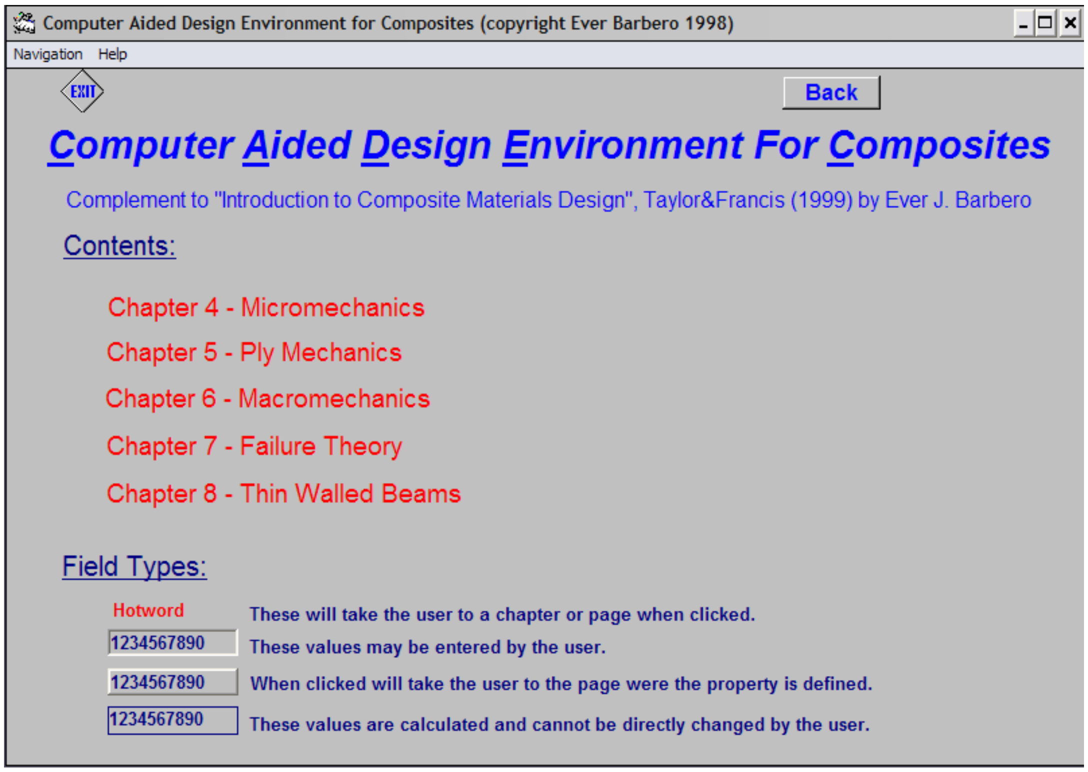
Ply #: 7 0 T300/5208

Failure Surface: Tsai Wu criteria

Figure 2.8 Results in x-y Graph form. [5]

2.5. Computer Aided Design Environment for Composites (CADEC):

CADEC is a Microsoft Windows application developed by Dr. Ever J Barbero, Mr. Enoch Ross, Mr. Steve Golf, Matt Fox etc… This software designed based on the "Introduction to Composite Materials Design" written by Dr. Ever J. Barbero [6], at West Virginia University. This program is divided into different modules, each module deals with different aspects of the composite materials. The Figure 2.8 shows the screen capture of the Main page of CADEC software [6].


Computer Aided Design Environment for Composites (copyright Ever Barbero 1998)
Navigation
Help
EXIT
Back

Computer Aided Design Environment For Composites

Complement to "Introduction to Composite Materials Design", Taylor&Francis (1999) by Ever J. Barbero

Contents:

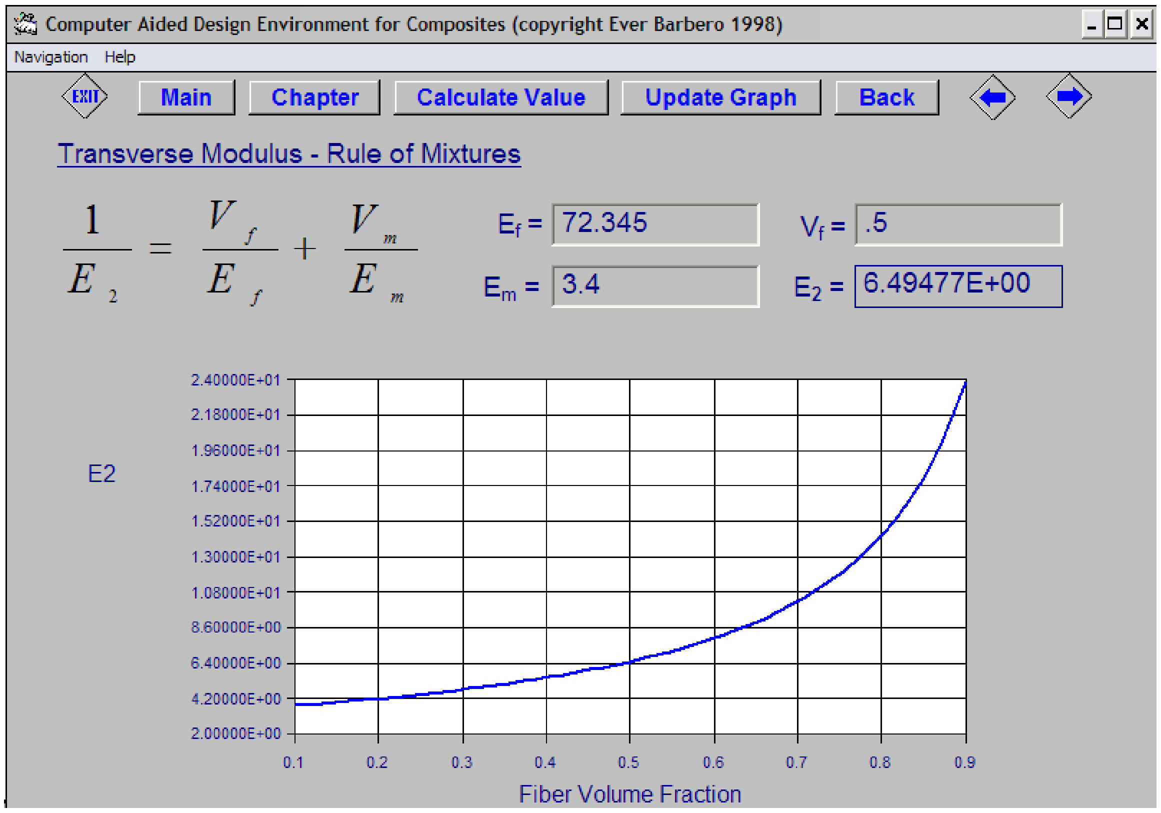
Chapter 4 - Micromechanics
Chapter 5 - Ply Mechanics
Chapter 6 - Macromechanics
Chapter 7 - Failure Theory
Chapter 8 - Thin Walled Beams

Field Types:

Hotword	These will take the user to a chapter or page when clicked.
1234567890	These values may be entered by the user.
1234567890	When clicked will take the user to the page were the property is defined.
1234567890	These values are calculated and cannot be directly changed by the user.

Figure 2.8 Main Page of the CADEC software [6]

The required properties are calculated and updated automatically based on the material properties entered by the user. In module Micromechanics, the software also plots and updates the graphs based on the user values. The Figure 2.9 below shows the screen capture of one such GUI page.



Transverse Modulus - Rule of Mixtures

$$\frac{1}{E_2} = \frac{V_f}{E_f} + \frac{V_m}{E_m}$$

E_f = 72.345

V_f = .5

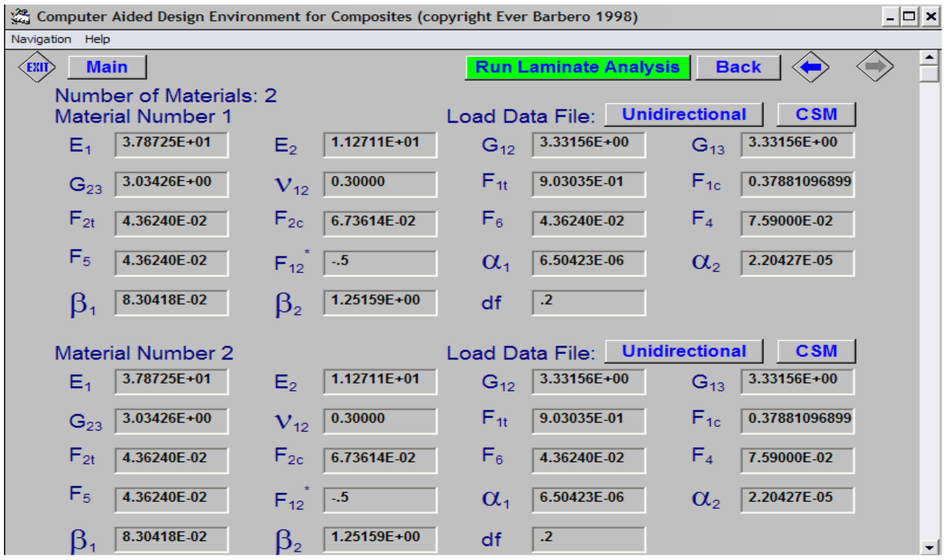
E_m = 3.4

E_2 = 6.49477E+00

Figure 2.9 GUI showing graphs for Transverse Modulus. [6]

CADEC software also analyzes laminates and beams. A Graphical User Interface is provided for each where the user can enter all the material properties values. Based on these values, the results for both laminate and beams are obtained. The

Figure 2.10 and Figure 2.11 show the part of the screens of both laminate and section user input GUI pages.


Computer Aided Design Environment for Composites (copyright Ever Barbero 1998)
Navigation Help
EXIT
Main
Run Laminate Analysis
Back

Number of Materials: 2

Material Number 1

Load Data File: Unidirectional CSM

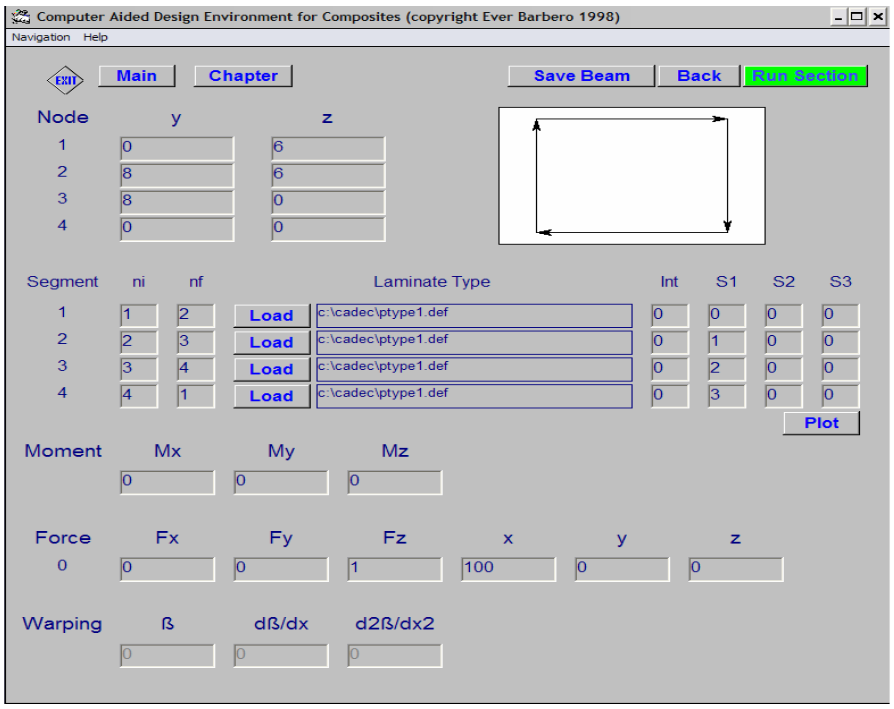
E_1	3.78725E+01	E_2	1.12711E+01	G_{12}	3.33156E+00	G_{13}	3.33156E+00
G_{23}	3.03426E+00	ν_{12}	0.30000	F_{1t}	9.03035E-01	F_{1c}	0.37881096899
F_{2t}	4.36240E-02	F_{2c}	6.73614E-02	F_6	4.36240E-02	F_4	7.59000E-02
F_5	4.36240E-02	F_{12}^*	-.5	α_1	6.50423E-06	α_2	2.20427E-05
β_1	8.30418E-02	β_2	1.25159E+00	df	.2		

Material Number 2

Load Data File: Unidirectional CSM

E_1	3.78725E+01	E_2	1.12711E+01	G_{12}	3.33156E+00	G_{13}	3.33156E+00
G_{23}	3.03426E+00	ν_{12}	0.30000	F_{1t}	9.03035E-01	F_{1c}	0.37881096899
F_{2t}	4.36240E-02	F_{2c}	6.73614E-02	F_6	4.36240E-02	F_4	7.59000E-02
F_5	4.36240E-02	F_{12}^*	-.5	α_1	6.50423E-06	α_2	2.20427E-05
β_1	8.30418E-02	β_2	1.25159E+00	df	.2		

Figure 2.10 Graphical User Interface for analyzing Laminate [6].


Computer Aided Design Environment for Composites (copyright Ever Barbero 1998)
Navigation Help
EXIT
Main
Chapter
Save Beam
Back
Run Section
Plot

Node	y	z
1	0	6
2	8	6
3	8	0
4	0	0

Segment	ni	nf		Laminate Type	Int	S1	S2	S3
1	1	2	Load	c:\cadec\ptype1.def	0	0	0	0
2	2	3	Load	c:\cadec\ptype1.def	0	1	0	0
3	3	4	Load	c:\cadec\ptype1.def	0	2	0	0
4	4	1	Load	c:\cadec\ptype1.def	0	3	0	0

Moment	Mx	My	Mz
	0	0	0

Force	Fx	Fy	Fz	x	y	z
0	0	0	1	100	0	0

Warping	ß	dß/dx	d2ß/dx2
	0	0	0

Figure 2.11 Graphical User Interface for analyzing Beam [6].

Chapter 3. Tool Design

The Computer Aided Design Environment for Composites (CADEC) software written in Toolbook is limited in its ability to run on an Operating System other than Windows. The software also requires that any user/client, who wishes to use it, has to download and install it on his/her PC. This would imply the following

1) All the users/clients have to be notified about the software changes.
2) Each user/client has to download and install the new version of the software.
3) Software Support during every installation

These limitations created a growing need for software that is portable. This means that the software can be run in a variety of different hardware or software environments with no or minimal changes. The CADEC software rewritten in java is portable and it provides the user/client to access the software over the Internet with a web-interface. This would eliminate the need for downloading/installing the software on the user/client's computer. The user/client need not be aware of any changes that are being done to the software. The changes can be anything from a simple bug fix to major software upgrades, providing new features etc.

The current CADEC software is a Java-based application. More specifically it’s an **Applet**, a Java program that can be embedded in Internet Web pages through the HTML page. When a user/client uses a browser to view the page with this applet, all the applet code is transferred to his/her pc and then executed by the browser’s **Java Virtual Machine**. If a user/client doesn’t have the required **plug-in** to run the software, the browser would prompt to download the same.

Java was the ultimate choice for writing portable software with ease. Java supports the object oriented programming (OOP) concept that makes it easy to write and maintain the software. Moreover Java is based on the concept of write once and run anywhere. The main features of Java that make it an apt choice are

a. Data Encapsulation
b. Object oriented
c. Portability
d. Robust & Secure

The software is designed based on the text book, “Introduction to Composite Materials Design”, written by Dr. Ever J. Barbero, professor at West Virginia University. This software is being divided into different modules.

1. Micromechanics
2. Plymechanics
3. Macromechanics

4. Failure Theory

5. Thin Walled Beams

Each module deals with different aspects of the Composite material. The Figure 3.1 below shows the screen capture of the main page of the new software.

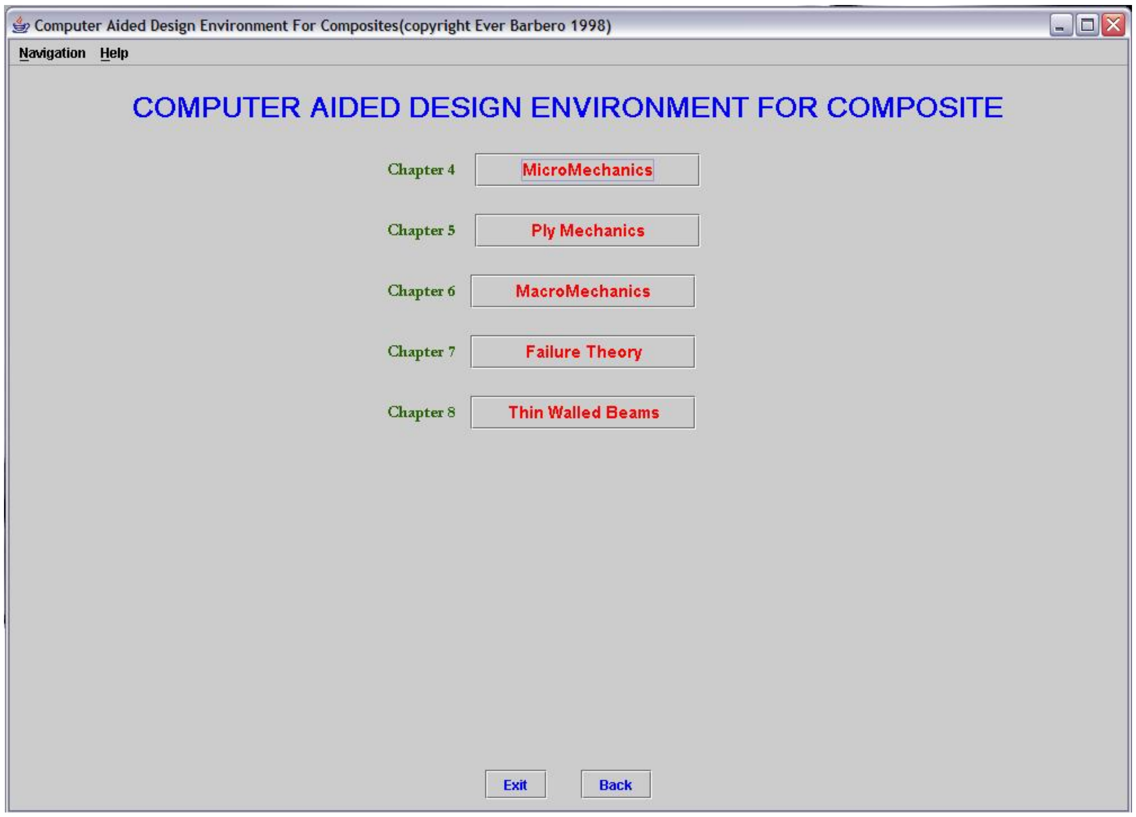


Figure 3.1 GUI of Main Page of CADEC

The main page of the software shows different buttons which when clicked take you to that respective page. The main page also shows the Navigation bar on the top left corner, which the User/Client can navigate.

3.1. Navigation:

Navigation through the GUI pages can be done using the Buttons provided on each page or through the “Navigation” bar provided on the top of each page. Navigation bar takes the user/client to some specific pages and the buttons enable the user/client can navigate to each page of the software. A Help option is also provided as the “Navigation” bar to aid the User/Client with the operation of the software.

3.2. Micromechanics:

Micromechanics is the study of composite materials taking into account the interaction of the constituent materials in detail [7]. It deals with the properties such as Longitudinal Modulus, Transverse Modulus, Inplane Poisson’s Ratio, Inplane Shear Modulus, Interlaminar Shear Modulus, etc. These properties were calculated based on the equations from the text book “Introduction to Composite Materials Design” by Dr. Ever J. Barbero.

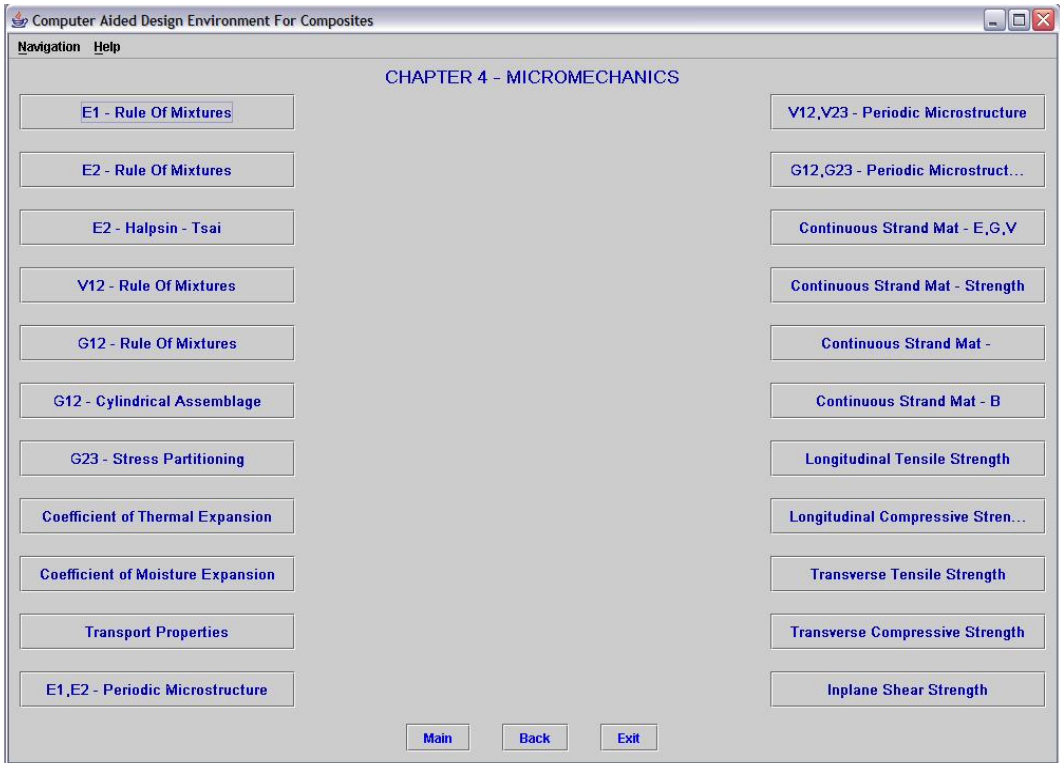
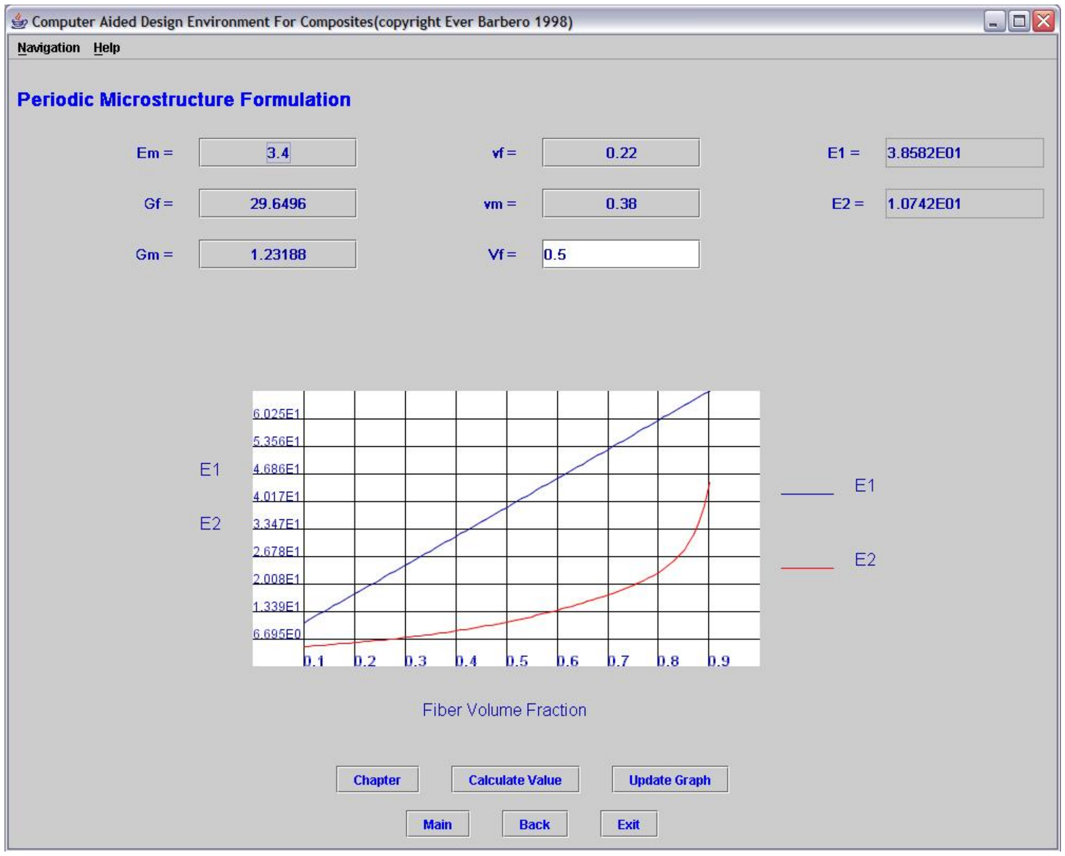

Computer Aided Design Environment For Composites
Navigation
Help
CHAPTER 4 - MICROMECHANICS
E1 - Rule Of Mixtures
E2 - Rule Of Mixtures
E2 - Halpsin - Tsai
V12 - Rule Of Mixtures
G12 - Rule Of Mixtures
G12 - Cylindrical Assemblage
G23 - Stress Partitioning
Coefficient of Thermal Expansion
Coefficient of Moisture Expansion
Transport Properties
E1,E2 - Periodic Microstructure
V12,V23 - Periodic Microstructure
G12,G23 - Periodic Microstruct...
Continuous Strand Mat - E,G,V
Continuous Strand Mat - Strength
Continuous Strand Mat -
Continuous Strand Mat - B
Longitudinal Tensile Strength
Longitudinal Compressive Stren...
Transverse Tensile Strength
Transverse Compressive Strength
Inplane Shear Strength
Main
Back
Exit

Figure 3.2 GUI page for the Micromechanics.

The Figure 3.2 above shows the screen capture of the Micromechanics page. The GUI page is designed with the buttons specifying a property on it. Pressing the button for the specific property would take the user/client to that page. Input for the values is being taken through the text fields. The required property is calculated based on the values entered by the user/client. Micromechanics also needs graph to be plotted for each property. The software automatically updates the value of the property entered in a page by storing the value in a buffer memory. Then it recalls its value wherever that property is needed in another page.



Computer Aided Design Environment For Composites(copyright Ever Barbero 1998)

Periodic Microstructure Formulation

Em =	3.4	vf =	0.22	E1 =	3.8582E01
Gf =	29.6496	vm =	0.38	E2 =	1.0742E01
Gm =	1.23188	Vf =	0.5		

Figure 3.3 GUI for Periodic Microstructure Formulation.

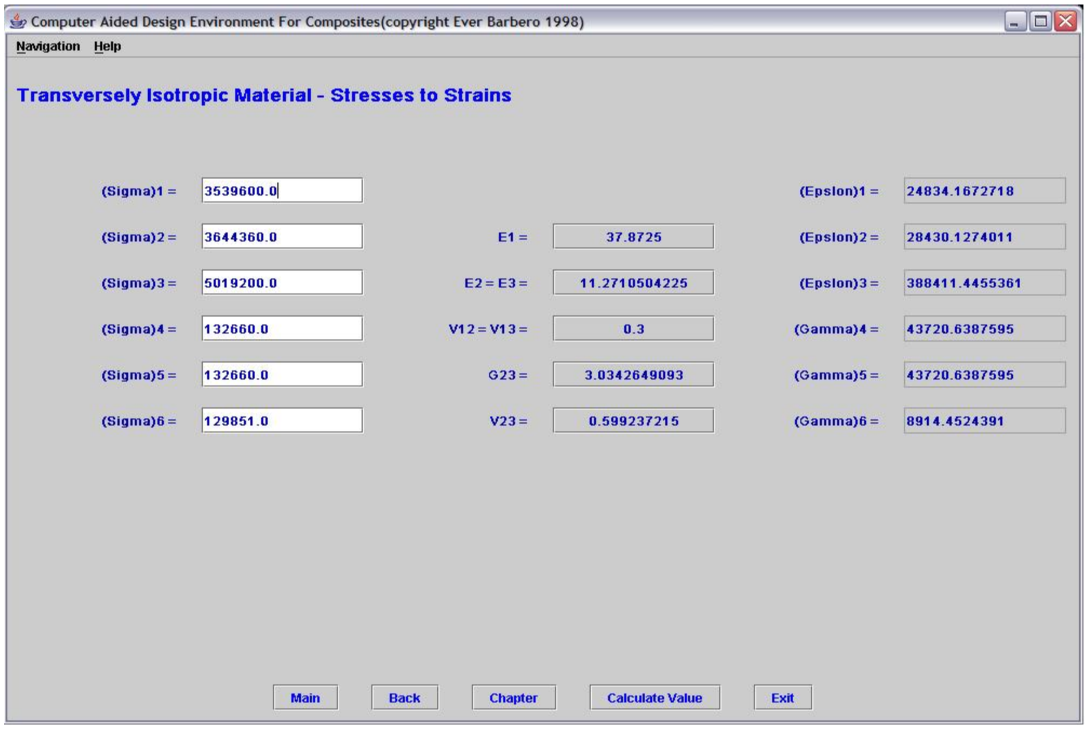
The Figure 3.3 above shows the GUI page of the Periodic Microstructure Formulation it also shows the graphs being plotted for E1, E2 properties along Y-direction and Volume Fraction (***V_f***) along X-direction. Default values are provided for all the fields throughout the software. The values are accessible everywhere by defining them as Global.

3.3. Ply Mechanics:

Unlike traditional materials, composite material properties vary with the orientation, having higher stiffness and strength along the fiber direction. Therefore, the transformation required to analyze composite structures along arbitrary direction not coinciding with the fiber direction are presented here [7].

The Ply Mechanics needed to implement matrix algebra to calculate the properties. The Java Matrix (JAMA) package has been used for this purpose [8]. Most of the basic property values used in this chapter like E1 (Longitudinal Modulus), E2 (Transverse Modulus) etc… are need to be recalled from the Micromechanics. Based on these values the plain stress stiffness and strain transformations, stiffness and compliance transformations and 3D constitute equations are calculated.

The Java Matrix package (JAMA) is a freely available on the Internet. This package is a basic linear algebra package for Java [8]. It provides different classes for manipulating and constructing matrices. Figure 3.4 below shows a GUI page for Ply Mechanics.


Computer Aided Design Environment For Composites(copyright Ever Barbero 1998)
Navigation
Help
Main
Back
Chapter
Calculate Value
Exit

Transversely Isotropic Material - Stresses to Strains

(Sigma)1 =	3539600.0			(Epslon)1 =	24834.1672718
(Sigma)2 =	3644360.0	E1 =	37.8725	(Epslon)2 =	28430.1274011
(Sigma)3 =	5019200.0	E2 = E3 =	11.2710504225	(Epslon)3 =	388411.4455361
(Sigma)4 =	132660.0	V12 = V13 =	0.3	(Gamma)4 =	43720.6387595
(Sigma)5 =	132660.0	G23 =	3.0342649093	(Gamma)5 =	43720.6387595
(Sigma)6 =	129851.0	V23 =	0.599237215	(Gamma)6 =	8914.4524391

Figure 3.5 GUI page for Transversely Isotropic Materials - Stress to Strains

3.4. Client/Server Programming:

The core of network application consists of a pair of programs – a client program and a server program [9]. When the two programs are executed, a server and client process are created. The communication that occurs between the client and the server must be reliable. That is, no data can be dropped and it must arrive on the client side in the same order in which the server sent it [10]. TCP (Transmission Control Protocol) provides the connection that is reliable, point to point and communication channel that the User and Client can use to communicate with each other on the Internet. They both communicate with each other through the sockets specified by the programs.

3.4.1. Sockets:

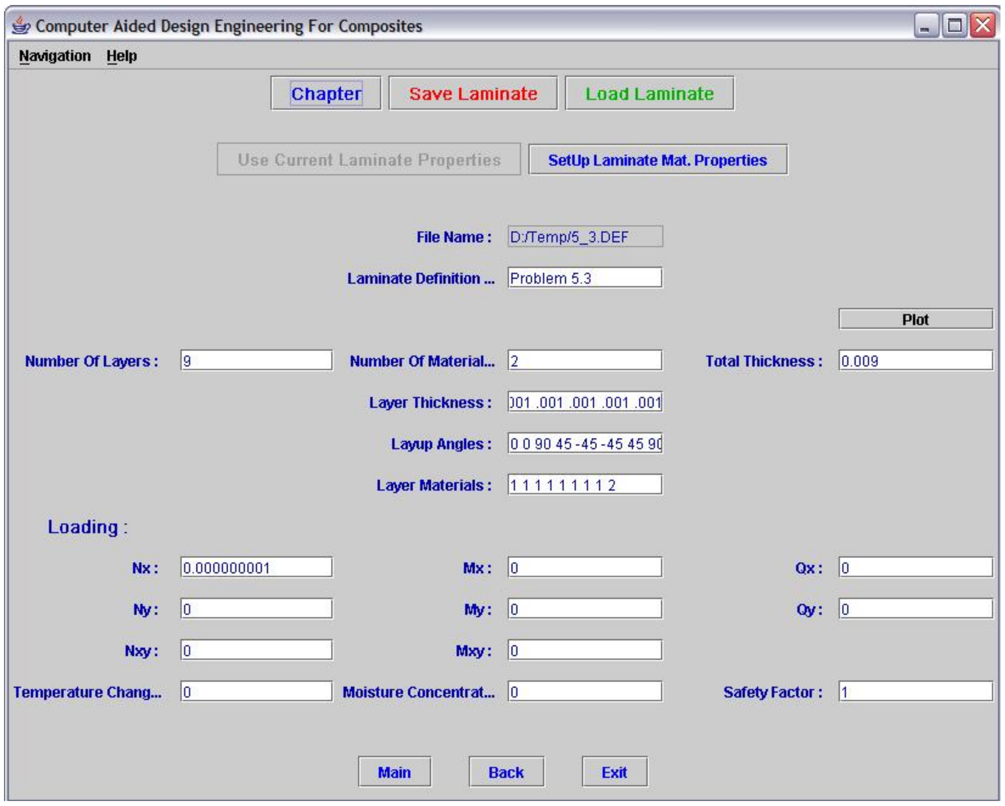
Socket can be defined as one point of two way communication link between the two programs running on the internet [10].

When the client sends the information to the server, the server accepts the information, analyzes it, performs the required tasks accordingly and sends the data back to the client. The client accepts the information and closes the connection.

3.5. Macromechanics:

Macromechanics and Thin walled Beams chapters needed two separate **.exe** files to be executed based on the input data provided by the client. Since the applets work on the user/client side browser, and the "**.exe**" cannot be embedded inside the browser. They have to be executed on the server side. Therefore Client/Server programming has been implemented to serve the purpose.

Macromechanics is divided into two pages where the user/client can enter values before the results can be viewed. The first being the Laminate Definition page where the user/client defines the number of materials and the number of layers for the composites to be analyzed. The screen capture of the Laminate Definition is shown below in Figure 3.6.

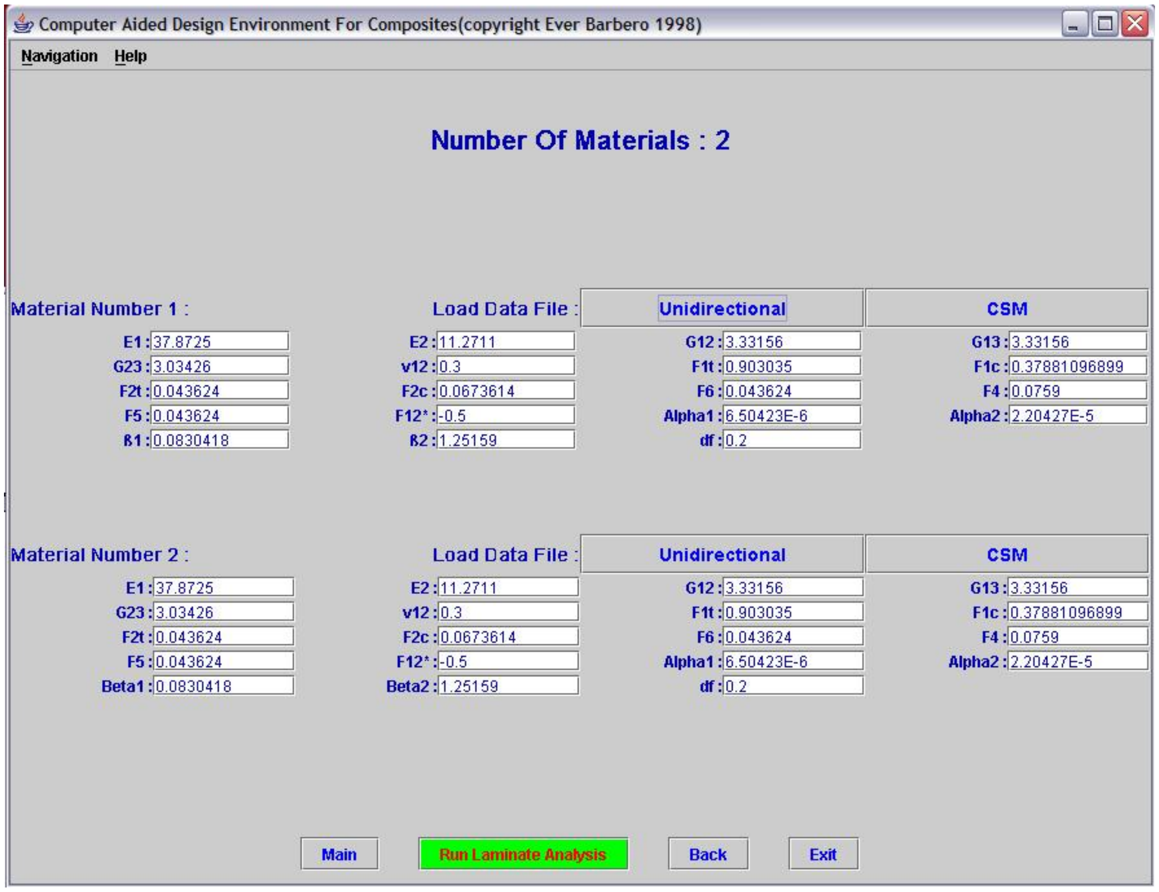

Computer Aided Design Engineering For Composites
Navigation
Help
Chapter
Save Laminate
Load Laminate
Use Current Laminate Properties
SetUp Laminate Mat. Properties
Plot
Loading :
Main
Back
Exit

File Name : D:/Temp/5_3.DEF
Laminate Definition ... Problem 5.3
Number Of Layers : 9
Number Of Material... 2
Total Thickness : 0.009
Layer Thickness :)01 .001 .001 .001 .001
Layup Angles : 0 0 90 45 -45 -45 45 9(
Layer Materials : 1 1 1 1 1 1 1 1 2
Nx : 0.000000001
Mx : 0
Qx : 0
Ny : 0
My : 0
Qy : 0
Nxy : 0
Mxy : 0
Temperature Chang... 0
Moisture Concentrat... 0
Safety Factor : 1

Figure 3.6 show GUI page for Laminate Definition.

Based on the number of materials and number of layers entered the software program checks for the consistency of layer thickness, lay-up angels, and layer materials. If not entered correctly, the software will prompt the user/client for the corrections.

The frame on the second page (Figure 3.7) is constructed dynamically depending on the number of materials specified in the first page (Fig 3.6). If the user/client has selected the number of materials as 2, then the user/client will be asked to provide materials properties for the two materials. Figure 3.7 below shows the screen capture of the same. The values shown here inside the text fields are the material values entered in Micromechanics.


Computer Aided Design Environment For Composites(copyright Ever Barbero 1998)
Navigation
Help
Main
Run Laminate Analysis
Back
Exit

Number Of Materials : 2

Material Number 1 : Load Data File : Unidirectional CSM

E1 :	37.8725	E2 :	11.2711	G12 :	3.33156	G13 :	3.33156
G23 :	3.03426	v12 :	0.3	F1t :	0.903035	F1c :	0.37881096899
F2t :	0.043624	F2c :	0.0673614	F6 :	0.043624	F4 :	0.0759
F5 :	0.043624	F12* :	-0.5	Alpha1 :	6.50423E-6	Alpha2 :	2.20427E-5
ß1 :	0.0830418	ß2 :	1.25159	df :	0.2		

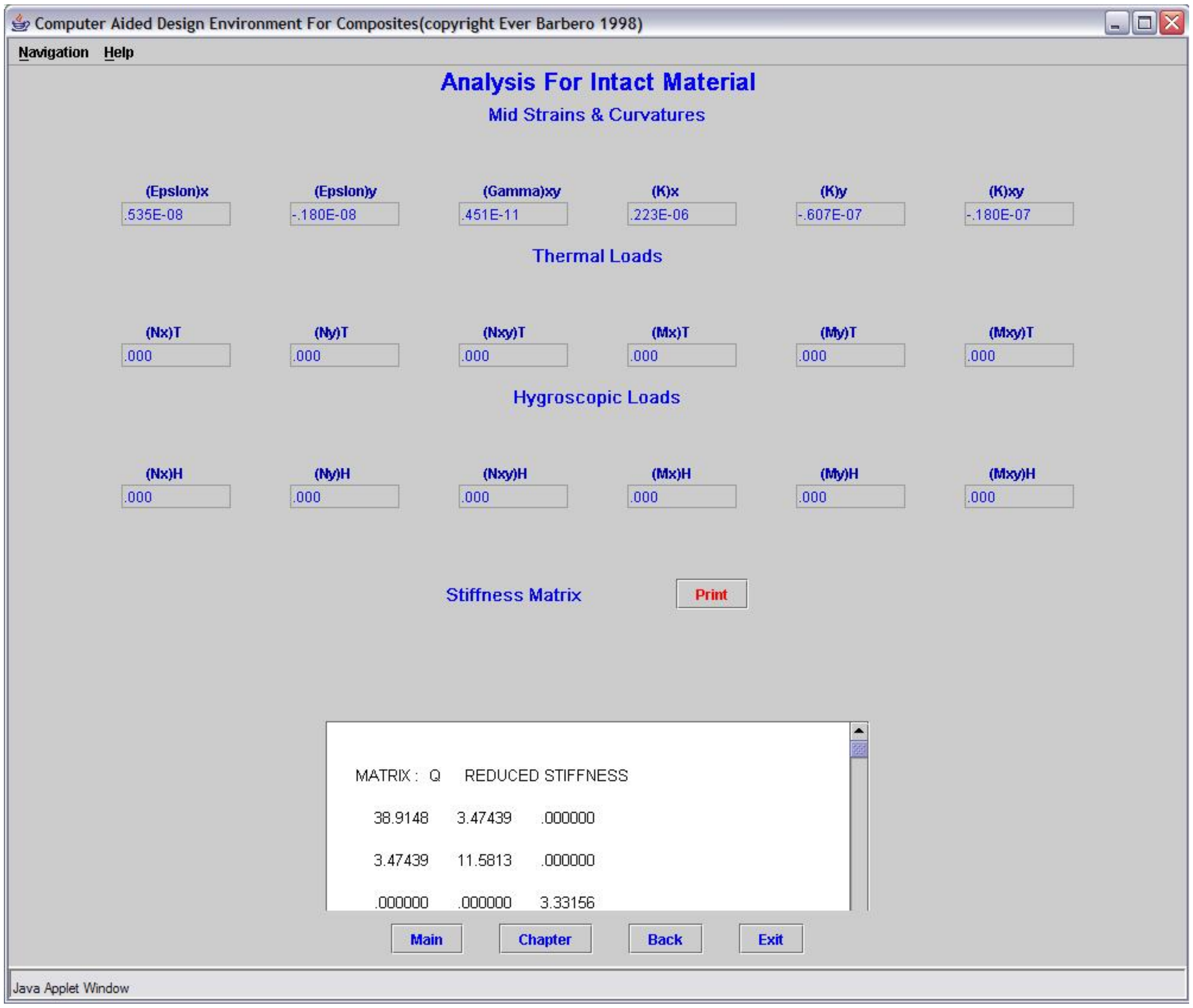
Material Number 2 : Load Data File : Unidirectional CSM

E1 :	37.8725	E2 :	11.2711	G12 :	3.33156	G13 :	3.33156
G23 :	3.03426	v12 :	0.3	F1t :	0.903035	F1c :	0.37881096899
F2t :	0.043624	F2c :	0.0673614	F6 :	0.043624	F4 :	0.0759
F5 :	0.043624	F12* :	-0.5	Alpha1 :	6.50423E-6	Alpha2 :	2.20427E-5
Beta1 :	0.0830418	Beta2 :	1.25159	df :	0.2		

Figure 3.8 shows GUI for Defining Laminate Material Properties.

The maximum number of materials the user can currently enter to analyze is set to 5. Depending on the number of materials entered by the user/client that many numbers of material values, to be entered, will be shown. The rest will be set as invisible dynamically. The Figure 3.8 shows the material values for 2 materials.

The User/Client analyzes the laminate by pressing the button “Run Laminate Analysis”. Once the user/client clicks the run button the client/server program takes action. The client will send messages to the server along with the required data and the file that needs to be executed. The **Server** then accepts the request from the **Client** and reads the data sent from the server along with commands. The server then creates a temporary directory for the client along with the **Input** file needed to execute the “**.exe**” file. After the results are obtained the server sends the results to the client and deletes the directory created for the client. The client accepts the data sent from the server and stores that result data and closes the connection. The tool is created such that multiple users can access it. The tool would remember the client analyze the data and send the results back to the respective clients. The data obtained are later showed on the respective GUI pages. Figure 3.10 below show the screen capture of the Graphical User Interface page for Reduced Stiffness Matrix.


Computer Aided Design Environment For Composites(copyright Ever Barbero 1998)
Navigation Help
Print
Main
Chapter
Back
Exit
Java Applet Window

Analysis For Intact Material

Mid Strains & Curvatures

(Epslon)x	(Epslon)y	(Gamma)xy	(K)x	(K)y	(K)xy
.535E-08	-.180E-08	.451E-11	.223E-06	-.607E-07	-.180E-07

Thermal Loads

(Nx)T	(Ny)T	(Nxy)T	(Mx)T	(My)T	(Mxy)T
.000	.000	.000	.000	.000	.000

Hygroscopic Loads

(Nx)H	(Ny)H	(Nxy)H	(Mx)H	(My)H	(Mxy)H
.000	.000	.000	.000	.000	.000

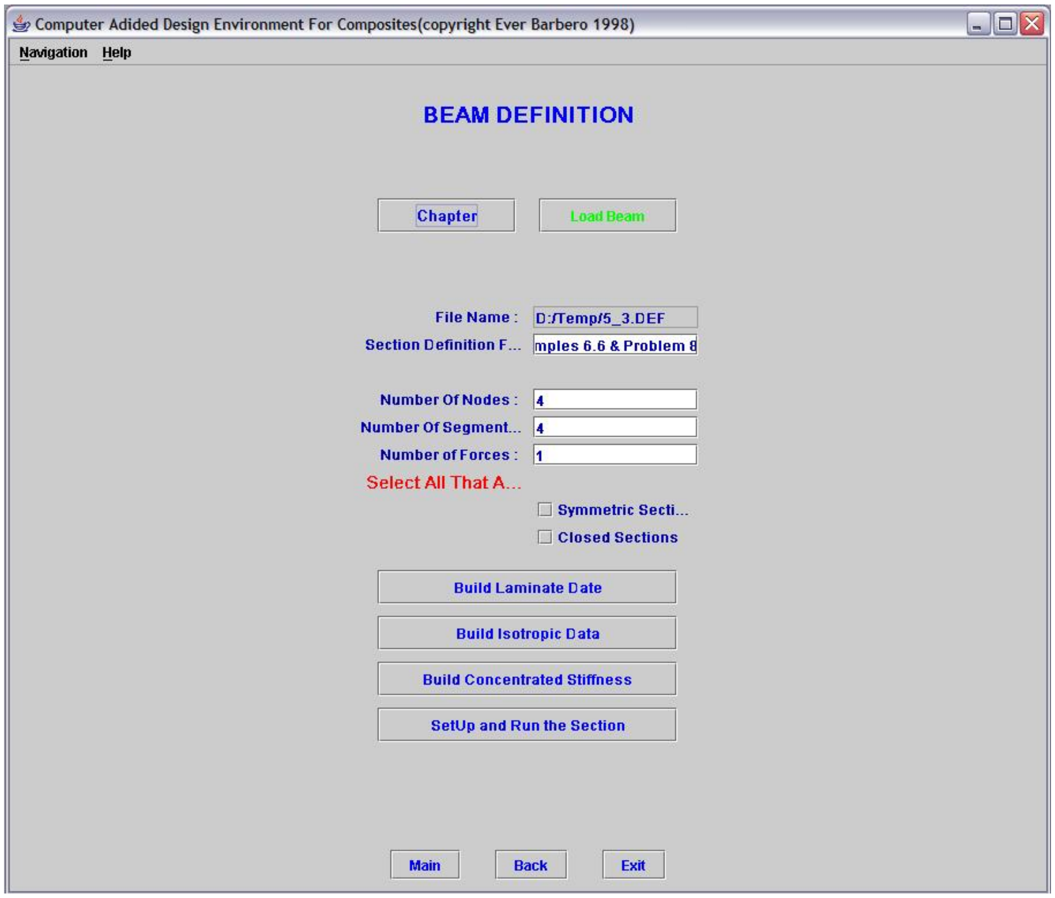
Stiffness Matrix

```
MATRIX :  Q     REDUCED STIFFNESS

    38.9148    3.47439    .000000

    3.47439    11.5813    .000000

    .000000    .000000    3.33156
```

Figure 3.10 Reduced Stiffness Matrix GUI page showing results.

3.6. Thin Walled Beams:

Thin walled beam is defined in two pages similar to Macromechanics. When the User/Client clicks on the button “Thin Walled Beam” on the Main Page, the user is guided to the Beam Definition page before the results are being viewed. The Beam Definition allows the User/Client to define the number of nodes, number of segments, number of forces and whether the Beam is a closed or symmetric section. Figure 3.11 below shows the GUI page for Beam Definition.


Computer Adided Design Environment For Composites(copyright Ever Barbero 1998)

BEAM DEFINITION

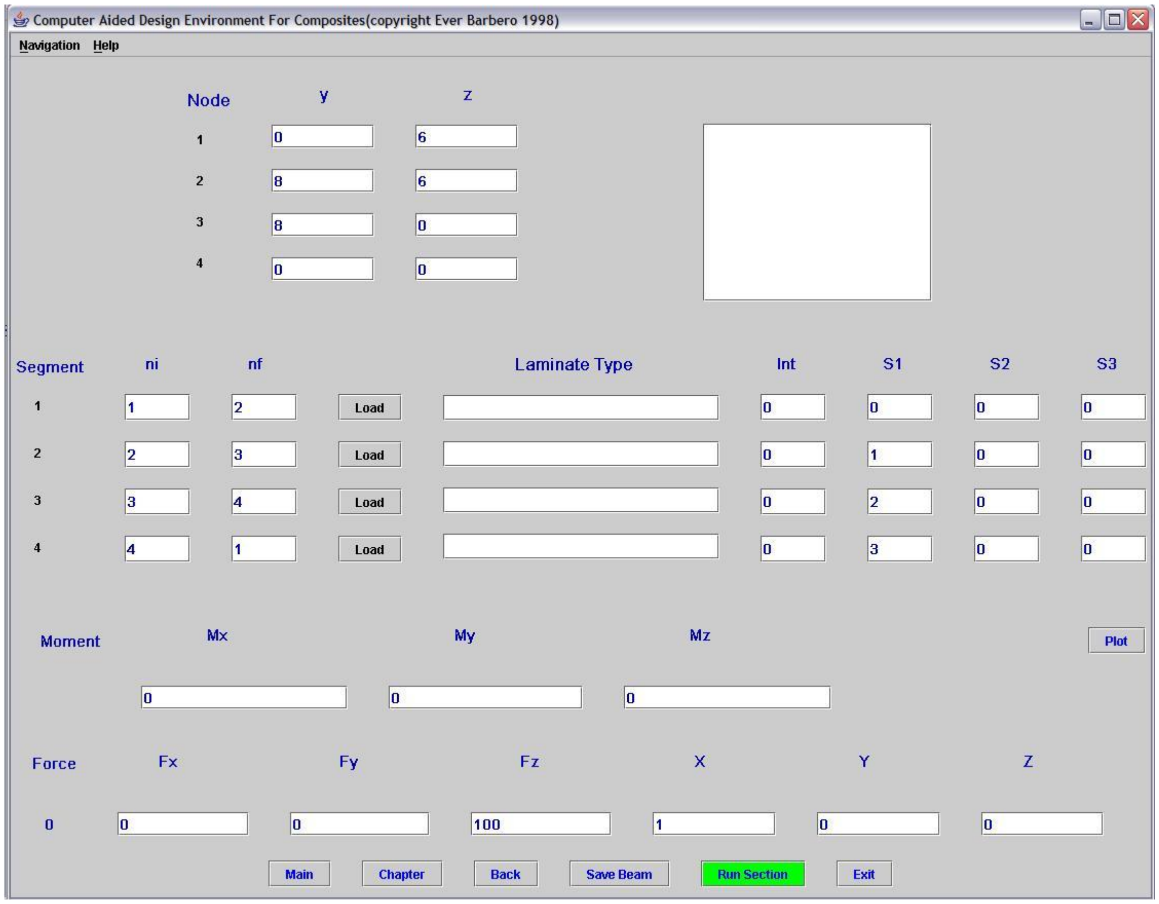
File Name : D:/Temp/5_3.DEF
Section Definition F... mples 6.6 & Problem 8

Number Of Nodes : 4
Number Of Segment... 4
Number of Forces : 1

Select All That A...
☐ Symmetric Secti...
☐ Closed Sections

Figure 3.11 GUI page for the Beam Definition.

The second page of the Thin Walled Beam is dynamically created and takes the material values depending on the values defined inside the Beam Definition. Figure 3.12 below shows the User Interface for the Beam Set Up page.


Computer Aided Design Environment For Composites(copyright Ever Barbero 1998)
Navigation
Help
Load
Plot
Main
Chapter
Back
Save Beam
Run Section
Exit

Node	y	z
1	0	6
2	8	6
3	8	0
4	0	0

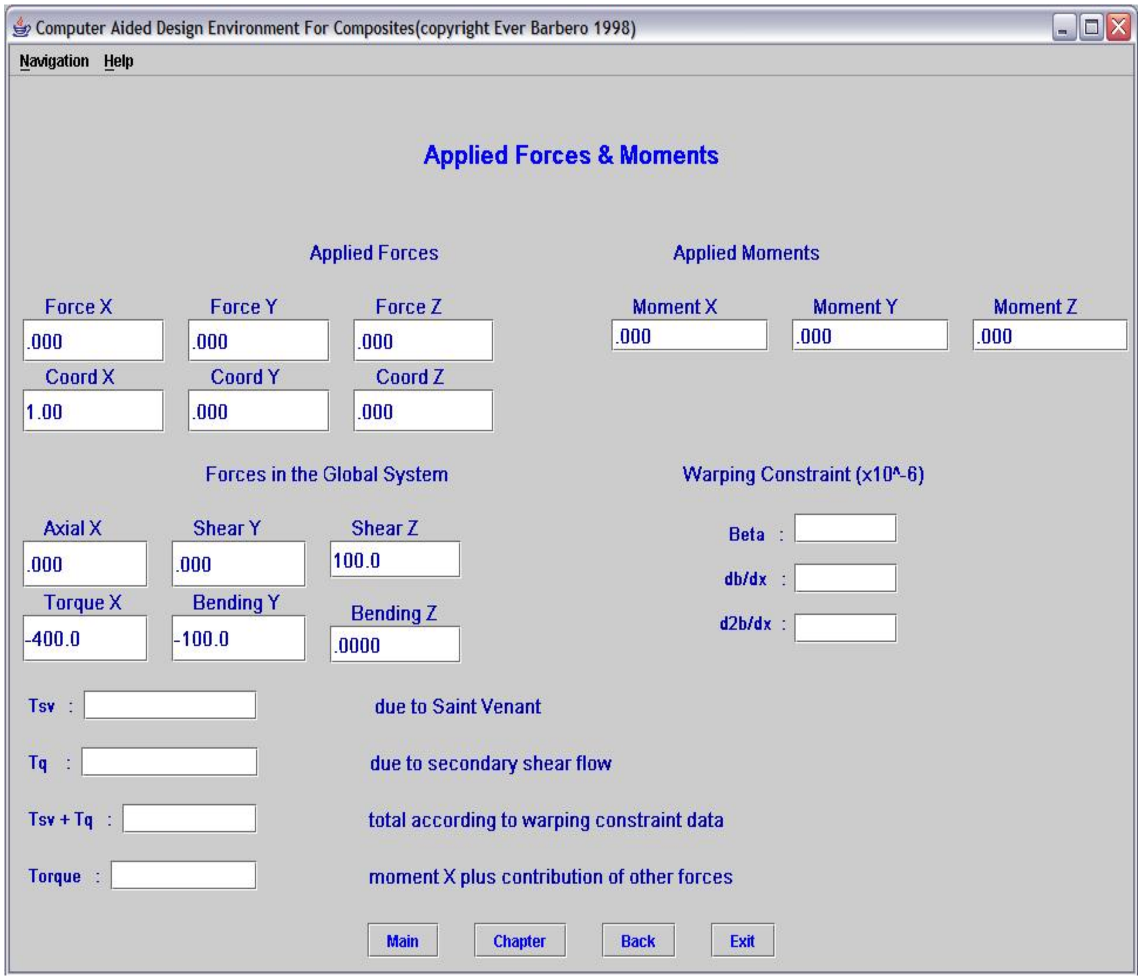
Segment	ni	nf		Laminate Type	Int	S1	S2	S3
1	1	2	Load		0	0	0	0
2	2	3	Load		0	1	0	0
3	3	4	Load		0	2	0	0
4	4	1	Load		0	3	0	0

Moment	Mx	My	Mz
	0	0	0

Force	Fx	Fy	Fz	X	Y	Z
0	0	0	100	1	0	0

Figure 3.12 shows the User Interface for the Beam Set Up Page.

After the user/client is done with entering the values he/she can analyze the beam section by pressing the button “Run Section”. When the “Run Section” button is executed, the Client opens the connection to the Server and it sends the command to analyze the beam along with the required data. The **Server** accepts the connection from the Client, accepts the data sent along with the run command. It then creates the temporary for the client, analyzes the beam, and sends the results back to the Client closing the connection. The Server also deletes the temporary directory created for the client. The Client would accept the results data and closes the connection. The results are later shown on various GUI pages based on the type of the result. Figure 3.13 below shows the result data for the Thin Walled Beams.


Computer Aided Design Environment For Composites(copyright Ever Barbero 1998)

Navigation Help

Applied Forces & Moments

Applied Forces

Force X	Force Y	Force Z
.000	.000	.000
Coord X	**Coord Y**	**Coord Z**
1.00	.000	.000

Applied Moments

Moment X	Moment Y	Moment Z
.000	.000	.000

Forces in the Global System

Axial X	Shear Y	Shear Z
.000	.000	100.0
Torque X	**Bending Y**	**Bending Z**
-400.0	-100.0	.0000

Warping Constraint (x10^-6)

Beta :

db/dx :

d2b/dx :

Tsv : due to Saint Venant

Tq : due to secondary shear flow

Tsv + Tq : total according to warping constraint data

Torque : moment X plus contribution of other forces

Figure 3.13 Results Page for Thin Walled Beams (Applied Forces & Moments).

The software tool is available at:

www.mae.wvu.edu/~cadecjava/cadec.html.

Chapter 4. Documentation

Computers are omnipresent, so much so that they have now become an integral part of life. Computer's can be programmed to do anything and everything. A program can be defined as an organized list of instructions that, when executed, causes the computer to behave in predetermined manner. Without programs, computers are useless [11]. Many programming languages have evolved since the invention of computers. Programming Languages can be divided into two type's namely Low level languages and High level languages. Low level languages are Machine and Assembly level languages. Each type of CPU has its own machine language and assembly language. So a program written in assembly language for one type of CPU won't run on another [12]. High level languages need compilers to convert those into machine language to interact with the computers. Java, COBOL, BASIC, Pascal, FORTRAN, Ada^{TM} Algol, C, C++ etc. are some of the high level languages which are easier for humans to understand and program using them. One such language which is widely used these days is Java. This chapter is intended for Java developers, not for CADEC users.

4.1. The Java Environment:

Java is programming language that lets you do almost anything that you can do with a traditional programming language like FORTRAN or C++ [13].

Java is considered to be relatively simple when compared to other languages. Java is platform independent, which implies that a java program once written can be run on any operating system. This is possible because the program does not run directly on the User/Client machine. It runs on the Hypothetical computer that is called the **Java Virtual Machine,** which is emulated inside the User/Client computer by a program [4]. Figure 4.1 below shows the standard execution of the Java program using the Java Virtual Machine.

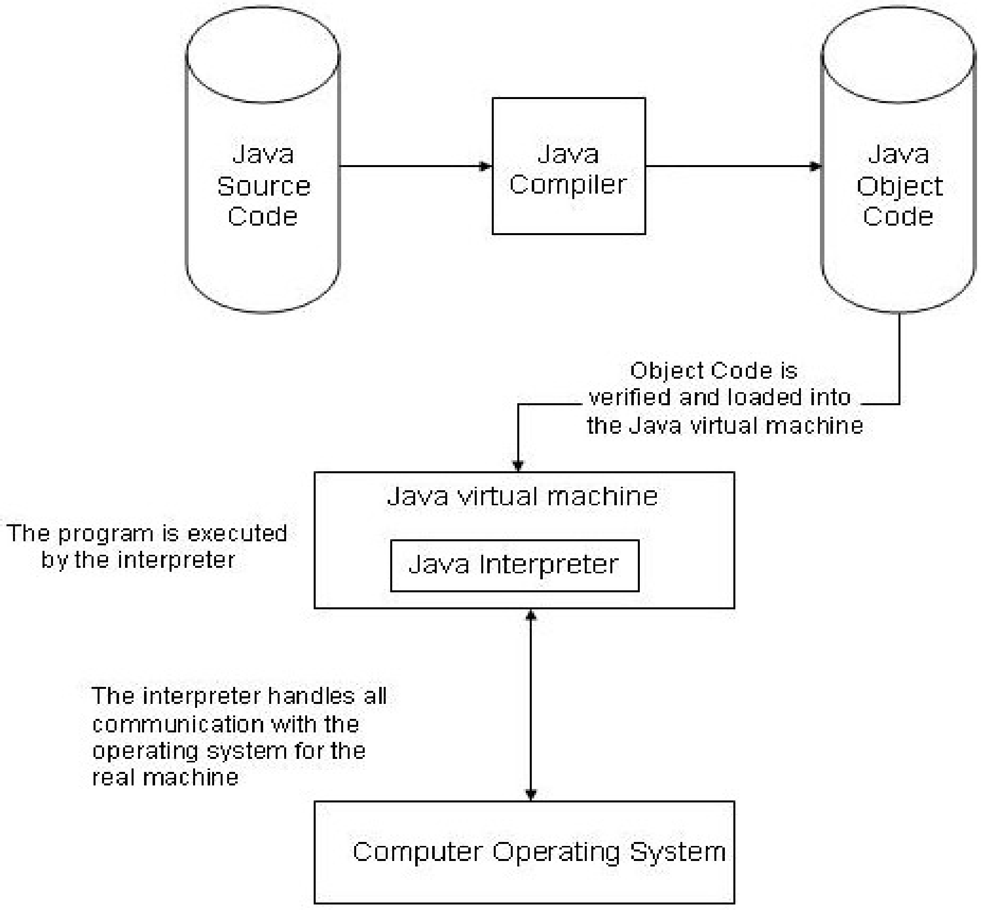

Java Source Code
Java Compiler
Java Object Code
Object Code is verified and loaded into the Java virtual machine
Java virtual machine
Java Interpreter
The program is executed by the interpreter
The interpreter handles all communication with the operating system for the real machine
Computer Operating System

Figure 4.1 Shows Standard Procedure for Execution of Java Program [1]

A high-level language program must be converted into low level instructions. A **Java Compiler** converts the Java source code into binary format consisting of byte codes. These byte codes are the machine instructions for the Java Virtual machine [1]. The Java Interpreter will check and translate the byte code when a Java program is executed. The interpreter checks for and makes sure that data is not being tampered and executes the actions that the byte code specified by the Java Virtual Machine. The compiler also increases the portability of the program. A Java Virtual Machine can run by itself or it can be a part of a browser (e.g., Microsoft Internet Explorer, or Fire Bird, or My IE 2, or Netscape Navigator) where it can be invoked automatically to run applets in a Web page [1].

Java is being used widely these days mainly because it is

- ❖ A Simple
- ❖ Object - Oriented
- ❖ Platform Independent
- ❖ Interpreted
- ❖ Distributed
- ❖ Robust
- ❖ Secure
- ❖ Architectural neutral

- ❖ Portable
- ❖ High – performance
- ❖ Multithreaded language

4.1.1. Object Oriented Programming:

A type of programming in which programmers define not only the data type of a data structure, but also the types of operations (functions) that can be applied to the data structure [14]. The data structures will become an object that defines both data and functions. Java is an Object Oriented language and almost everything in Java are Objects. An **Object** is a software bundle of related **variables** and **methods**. Software objects are often used to model real–world objects you find in every day [15]. The real–world objects have two characteristics namely **state** and **behavior**. For example cars have state (Four wheels, number of gears, current gear, etc...) and behavior (changing gears, accelerating, braking etc...). Software models are modeled in the same way as real world objects in that they too have state and behavior.

A **Class** is a blueprint or prototype that defines the variables and the methods common to all objects of a certain kind [16]. An instance of a class is a term for an Object of that class. For example in real world we have different objects of same kind. A mountain bike is one of many bicycles of its kind. Using

Object-oriented terminology, we can say that our mountain bike object is an instance of the class of objects known as bicycles [16].

4.1.2. Installing Java:

The Figure 4.2 shows the typical directory hierarchy for the Java Distribution Kit (JDK) if installed in the C drive for Windows.

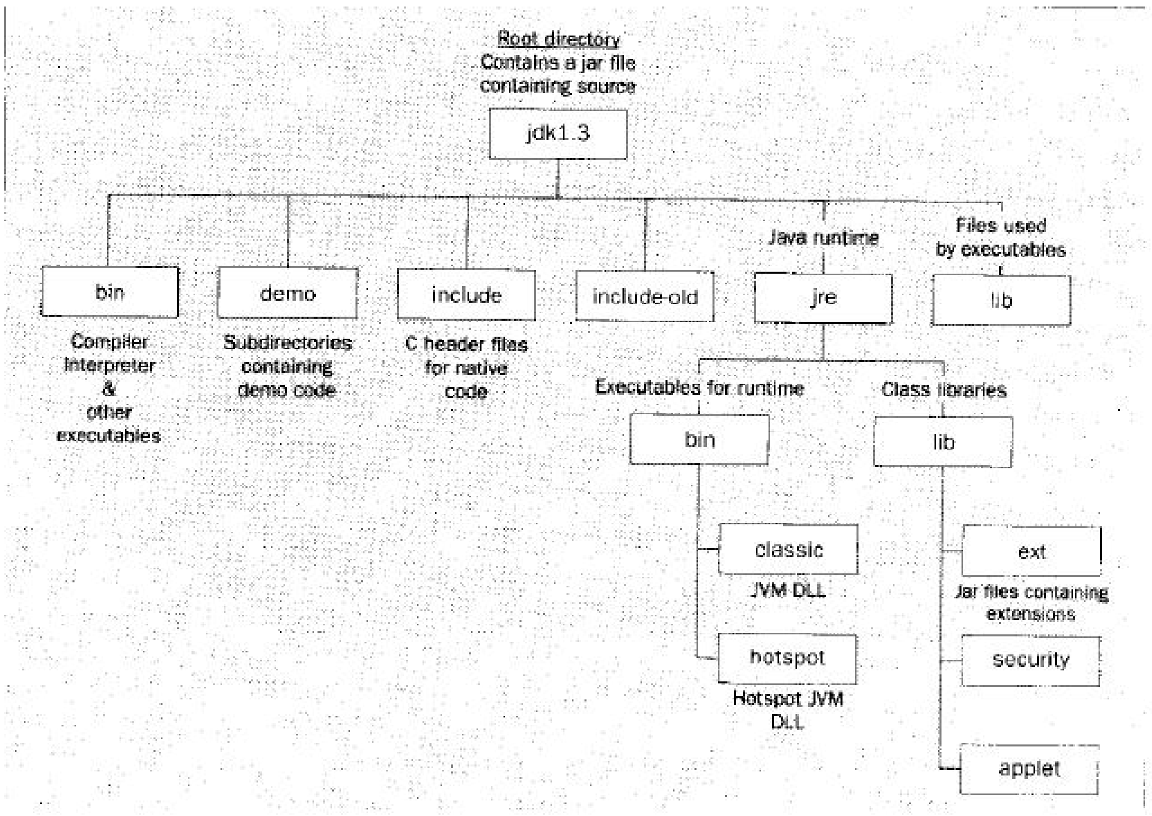

Root directory
Contains a jar file containing source
jdk1.3
bin
Compiler Interpreter & other executables
demo
Subdirectories containing demo code
include
C header files for native code
include-old
Java runtime
jre
Files used by executables
lib
Executables for runtime
bin
classic
JVM DLL
hotspot
Hotspot JVM DLL
Class libraries
lib
ext
Jar files containing extensions
security
applet

Figure 4.2 Shows the Installation Directory Tree for JDK [1].

The Java^{TM} Standard Edition (J2SE) can be downloaded from the following web site at free of cost.

http://java.sun.com/j2se/1.4.2/download.html

The JDK is needed by all the User/Client who intends to run java on their system. After the download process is complete and the user has installed JDK on his computer it is suggested that the user specify the path variables. This will allow the user to run the executables (java.exe, javac.exe, javadoc.exe) from any directory without having to specify the full path of the command. A detailed information regarding the installation process can be obtained through the following web site. This is not necessary for users that just want to use CADEC and do not intend to do any Java development work.

http://java.sun.com/j2se/1.4.2/install-windows.html#install

Java also provides previously written classes and methods that the user can use instead of writing them again, thereby reducing the development time. The user can use them through the instructions provided in the documentation. The documentation for J2SE can be found at http://www.java.sun.com/docs. The user can also download the documentation from the following web address if he intends to work offline http://java.sun.com/j2se/1.4.2/download.html

4.2. A Small Java Program:

The following is a Code for a Java Program.

```
public class MyProgram{
      public static void main(String args[]){
            System.out.println(" My Program is Executed");
      }
}
```

4.2.1. Saving the Java File:

Generally all the Java files are saved with an extension ".java". It is recommended and considered to be a good programming concept to save the file with the same name as that of the class. Here the **class** is defined as "MyProgram" and it can be saved as **MyProgram.java**.

4.2.2. Compiling the Java File:

The java source can be compiled using the java compiler (javac) which comes as a default with the JDK.

The Syntax for compiling the code e.g., on a DOS window session is

Syntax: **javac MyProgram.java**

The prompt has to be in the same directory where the file is being saved. If the user wants to compile the code from a different directory, then the full path has to be provided.

After a successful compilation, the java compiler generates a byte code program that is equivalent to the user source code. The compiler then stores the byte code program in a file with the same name as the source file, but with an extension **".class".** Java executables are always stored in a "**.class**" file extension.

4.2.3. Executing the Java Application:

The Java interpreter can then be used to run the program using the syntax shown below

Syntax: **java MyProgram**

Execution of Java does need the extension since it refers to the "**.class**" instance rather than "**.java**". The java Interpreter than analyzes the code and executes it based on the byte code commands. The Java Virtual Machine is similar to all the computer environments supporting Java. This is one of the main reasons for Java to be portable.

The output of the program will show “**My Program is Executed**” at the command prompt (e.g., DOS command prompt, Linux command prompt, etc.).

4.3. Class:

Conceptually, a class represents a concept, a template for creating instances (Objects). In a computer, a class is a chunk of memory, set aside once, when the class is loaded at run-time [17]. The class is always defined with a pair consisting of opening and closing braces. The rest of the program is written inside them. The syntax for defining a class is as follows

Syntax:

```
public class classname {
        class variables;
        declaring  global variables or instance variables;
        defining methods (parameters if necessary) {
        method or local variables,
        //body of method or equations in our case.
        }
}
```

4.3.1. Class Variables:

These types of variables are shared by all objects of the class. They have one copy of each of these kinds irrespective of how many class objects are created. These variables are also called static fields. Class variables exist even when no objects of the class have been created.

Example:

```
public class MyProgram{
       static String myProgram = "My Program is Executed";

       public static void main(String args[]){
              System.out.println(myProgram);
       }
}
```

In the program above the variable "String" is defined as a static and every instance (Object) of **myProgram** will return the value "My Program is Executed". The output of the program will be the same as the previous one shown above. The word "void" defined above signifies that the method "**main**" does not return any values.

4.4. Method:

```
method(){

        method variables
        //Body of the Method.
}
```

A method consists of opening and closing braces similar to the class. All the methods start with a small alphabet.

4.4.1. Method Variables:

These variables defined inside the method are only known to that method.

```
public class MyProgram
{
    public static void main(String args[])
    {
        String myProgram = "My Program is Executed";
        System.out.println(myProgram);
    }
}
```

Here the String variable is defined inside the “main” method and that variable will only be known inside that method.

4.5. public

The **public** defined in front of the **class** allows creating instances (Objects) of that class. It also allows using the methods defined inside that class. If the class is not defined as public then the class will be treated as a private class thus restricting other classes from using the methods defined in that class.

4.6. Print Line:

System.out.println("My Program is Executed ");

The command line above will print anything written in between quotes in the DOS window. Here **My Program is Executed** will be printed. All the statements in the program should be terminated by a semicolon at the end.

4.7. Comments:

Java provides two types of comments inside its program. A single line comment and a multi-line comment, these comments are usually used to give the description of a method or a class or a variable used.

4.7.1. Single Line Comment:

A single line comment start’s with two back slashes. The compiler ignores the message written after the comment in that whole line.

// This line is ignored by java compiler.

4.7.2. Multi-Line Comment

The syntax for writing multi-line comment is

/*

Anything written in between will be ignored, by the java compiler. This can run through any number of lines.

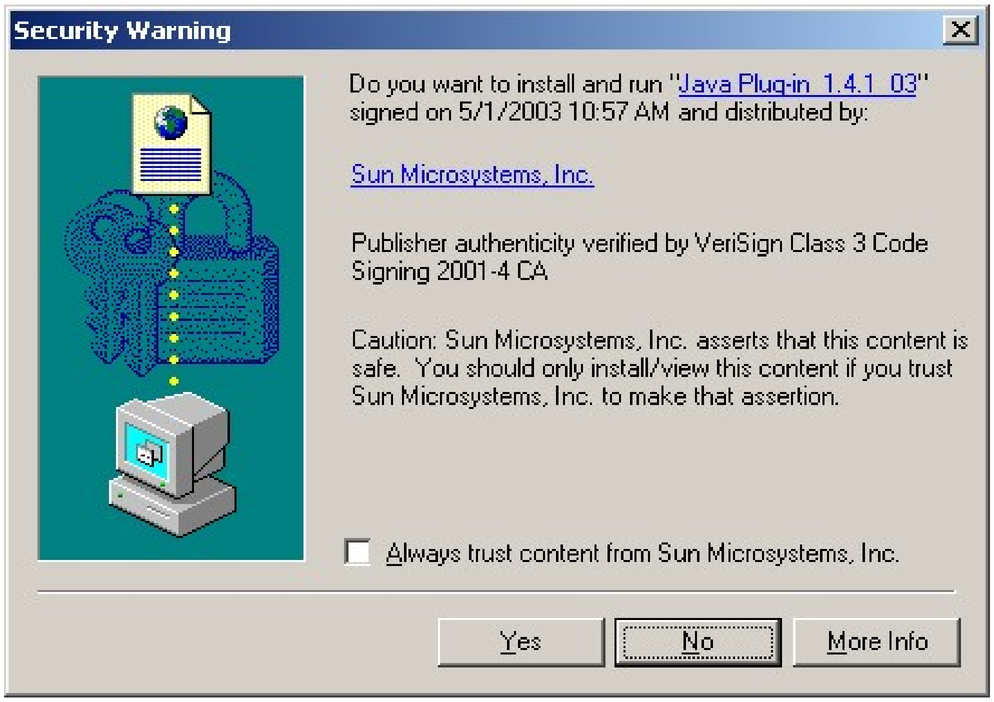
***/**

4.8. Java Class Library:

A **library** in java is a collection of classes usually providing related facilities that the programmer can use in the program [1]. Java also provides **packages**, the classes grouped together in related sets, and each package is stored in a separate directory. Every class in Java is contained in the package. The name used for one class in a package will not interfere with the name used in another package. The package name is related to the path of the directory inside which the classes belonging to the package are stored. The way of using classes from there packages is to simply use an **import** statement for each package needed. These statements usually appear at the beginning of the programs. Based on the classes needed for the specific program general **import** and **package** statements might be used.

4.9. Applets:

Applets are small programs written in java, which can be embedded inside the Internet Web pages providing them with some intelligence. Applets code is transferred to the User/Client machine when the User/Client tries to view the applet page. It is executed using the **Java Virtual Machine** provided by the browser. If the User/Client browser is not Java enabled then the User/Client will be prompted to download the **plug-in** required to install it. The screen capture of the **plug-in** is being shown Figure 4.3.


Security Warning

Do you want to install and run "Java Plug-in 1.4.1_03" signed on 5/1/2003 10:57 AM and distributed by:

Sun Microsystems, Inc.

Publisher authenticity verified by VeriSign Class 3 Code Signing 2001-4 CA

Caution: Sun Microsystems, Inc. asserts that this content is safe. You should only install/view this content if you trust Sun Microsystems, Inc. to make that assertion.

Always trust content from Sun Microsystems, Inc.

Figure 4.3 Installation Window asking for Permission from the User.

4.9.1. Writing a Simple Applet Program:

The code below shows a simple Applet code.

```
import javax.swing.JApplet;
import java.awt.Graphics;

public class MyApplet extends JApplet{

        public void paint(Graphics g){
                g.drawString("My Applet", 40,60);
        }
}
```

Applets are compiled in the same way as other programs. The java compiler inside the JDK will compile both the applets and the applications.

Applets are viewed through the Internet browser and have to be embedded first inside an HTML(Hyper Text Markup Language) page. There are two methods to execute the Applets, one using the browser Java Virtual Machine and the other is to use the applet viewer provided as a part of the JDK kit. The applet class is then embedded in an html file.

The syntax for using the appletviewer through the DOS prompt is

```
appletviewer  MyApplet.html
```

If the user wants to view it through the java enabled browser, the user has to open the MyApplet.html inside the browser.

4.9.2. A simple HTML file:

The code below shows a simple way of embedding an executable file inside the HTML is as follows.

```
<HTML>
<HEAD>
<TITLE>My Applet Program</TITLE>
</HEAD>
<HR>
```

```
<APPLET code="MyApplet.class" width=300 height=300>

</HR>

</APPLET>

</HTML>
```

4.10. Frame:

Frame is a top level window that provides a banner, title and various methods to manage the behavior and appearance of the window.

Another way of writing an applet is to create a frame first and then adding the frame to the applet. By doing so, many frames can be created, then the Main Page of the program can be added to the applet.

The code below shows an example of a Frame program.

```
import javax.swing.*;

public class GuiExample extends JFrame{

    public GuiExample(){

        this.setSize(300,300);

    }

    public static void main(String args[]){
        GuiExample ge = new GuiExample();
        ge.setVisible(true);
    }
}
```

The code when executed will provide the following output.

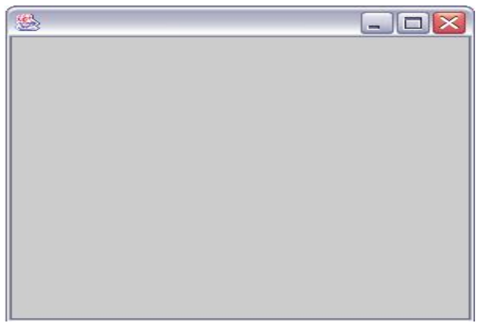


Figure 4.4 Shows output for the program GuiExample

4.11. Constructors:

Constructors always have the same name as the class and no return type [18]. Classes call the constructor when its instance (object) is created. If the programmer does not define a constructor the class defines a default constructor and initializes all the non initialized variables and fields to zero. In the example program GuiExample, its constructor is defined with the same name as the program i.e., **“GuiExample()”**

4.11.1. Instance:

The general syntax for creating and instance (object) of a class is

classname instancename = new **classname();**

Creating an instance of a class would enable the programmer to access and work on data created by that class. The Figure 4.4 below shows three instances of an example class.

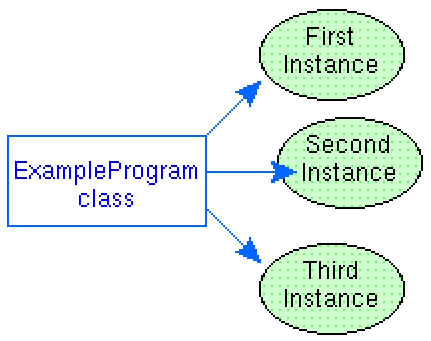


Figure 4.5 Instances of the ExampleProgram class.[17]

4.12. Class Declaration:

Each of the GUI components is defined inside the Abstract Window Toolkit (AWT) package. Java also defines **Swing** package which extends AWT providing good look and feel effect and assisting technology support. The JFrame extends the Frame class, which is part of the AWT.

4.13. Instance Variables:

Java library defines different component classes that the programmer can use. Swing defines different classes. Using the instances of swing classes, the programmer can create a Graphical User Interface. All the defined classes can be viewed using the Application Program Interface (API) documentation. The classes JLabel, JButton, JPanel, JTextField etc…. are useful in creating a GUI.

4.14. Client/Server Programming:

The Java platform is highly regarded in part because of its suitability for writing programs that use and interact with the resources on the Internet and the World Wide Web [19]. Java supports various protocols. It supports Internet Protocol, File Transfer Protocol (FTP), Hyper Text Transfer Protocol (HTTP), Transmission Control Protocol (TCP) etc..., and Remote Method Invocation (RMI).

There are two types of communication protocols that the programmer can use to communicate on the Internet, datagram communication and stream communication. The order in which the data is received and sent is crucial to the success of the application.

4.14.1. Socket:

A socket can be defined as an object using which data can be transmitted or received from distant machines running on the internet. The “Java.net” package provides two classes called **Socket** and **ServerSocket**, which implement the client side and the server side connection respectively. A socket is bound to a port number so that the TCP layer can identify the application to which data is to be sent [19].

4.14.2. Port:

A computer has a unique physical connection to the network and all the data will be sent through this connection. Port is a means by which computer remembers which data to send to which computer among multiple computers. The data transmitted over the internet is accompanied by the port number and the computer address information for which it is destined. Ports are identified by their 16-bit number where as computers are identified by their 32-bit IP address.

4.14.3. Data Communication:

The datagram communication protocol also called User Datagram Protocol (UDP) is a connectionless protocol. It means that each time you send datagrams, you need to send the local socket descriptor and receiving socket address [20]. That would imply sending of extra data whenever connection is being made.

4.14.4. Stream Communication:

The Stream communication protocol also called as Transmission Control Protocol (TCP) is a connection oriented protocol, which implies that a connection has to be established first between the pair of sockets. One socket (Server) listens to the request while the other socket (Client) asks for the connection. Once the connection is established they can transmit data (read or write data etc...) in both directions. TCP is useful for implementing network services, such as remote log in, or file transfer protocols (FTP). TCP provides a reliable flow of data between two computers.

4.14.4.1. TCP/IP Programming:

In TCP/IP Programming, the server runs on a specific computer with socket running on a specific port number. The server cannot be dormant; it should be running all the time as a process and should always be ready to listen for the Client at that given particular port number.

The client first initiates the contact with the server. With the server running, the client process can then initiate a connection with the server. This is done by creating a socket object at the client. While creating the socket object the client specifies the IP address of the computer and the port number of the process. Thus the connection is established with the server.

The Server accepts the connection from the client. Once the connection is established the server creates a new socket and a different port number and allocates that to the client. This is important because it allows the server to keep listening on the original port number to accept connection requests from other clients. Figure 4.6 below shows the diagram for the process communication using TCP sockets.

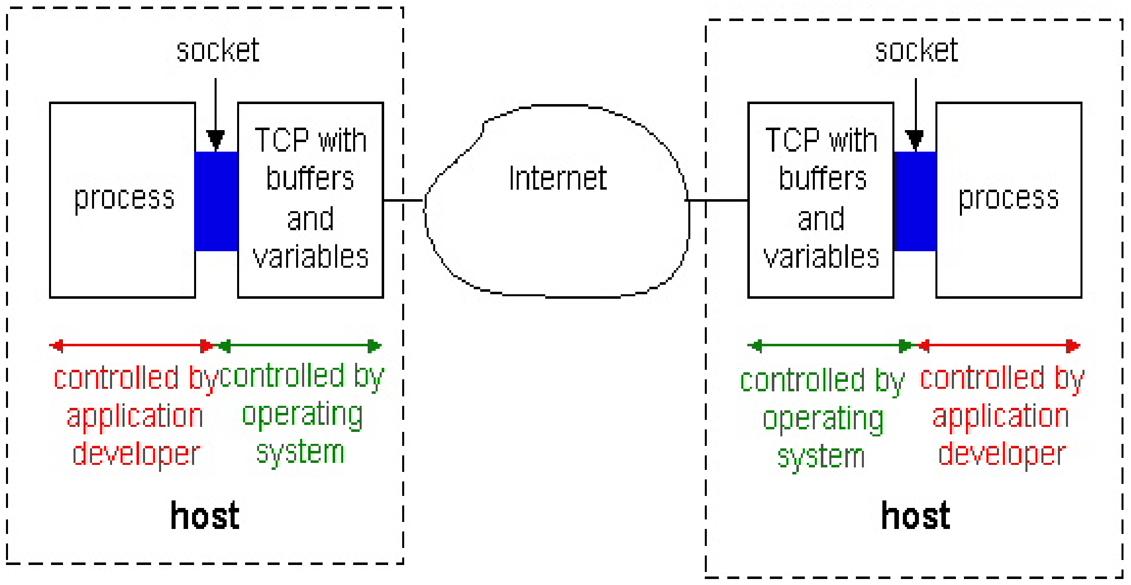

socket
process
TCP with buffers and variables
controlled by application developer
controlled by operating system
host
Internet
socket
TCP with buffers and variables
process
controlled by operating system
controlled by application developer
host

Figure 4.5 Process Communication through TCP sockets [21].

4.15. Example of Source code:

The following source code has been taken from the **MainPage.java** of the Thesis.

Comments have been included to help understanding of the code.

Source code for MainPage.java

```
import java.awt.*;
import java.awt.event.*;
import java.awt.Font.*;
import java.awt.Color.*;
import java.util.*;
import java.lang.*;
import java.awt.Frame.*;
import java.awt.event.*;
import javax.swing.*;


/**
* STEPS to CREATE A GUI with a simple menu,button,label.
* JPanel northJPanel = new JPanel(); // CREATE THE PANEL
* this.getContentPane().setLayout(new BorderLayout());  // CREATE A LAYOUT
* getContentPane().add("North", northJPanel); // ADD THE PANEL TO THE NORTH SIDE
* OF THE FRAME
* northJPanel.setFont(new Font("TimesNewRoman", Font.BOLD, 25)); // SET FONT TYPE
* northJPanel.setForeground(Color.blue); // SET FOREGROUND
* northJPanel.add("North", title = new JLabel* ("COMPUTER AIDED DESIGN
*ENVIRONMENT FOR COMPOSITE ", JLabel.CENTER));  // SET LABEL
* JButton e1Rule = new JButton();   // CREATE A BUTTON
* northJPanel.add(e1Rule = new JButton("E1 - Rule Of Mixtures")); // ADD THE BUTTON TO
* THE PANEL
*
* JMenu fileJMenu = new JMenu(" Navigation");   // CREATE A MENU
* JMenuItem mmItem = fileJMenu.add(new JMenuItem("MicroMechanics")); // ADD A MENU
* ITEM
*
*/

public class MainPage extends JFrame {

   //Creating Objects of the classes MicroMechanics
   //and others in order to use them in this class
   MicroMechanics mm = null;
   MacroMechanics mam = null;
   PlyMechanics   pm = null;
```

Source code for MainPage.java (Continued …)

```
FailureTheory  ft = null;
ThinWalledBeams tw = null;
MicroMechUpdatePage1 mmup1 = null;
LaminateDefinition lamDef = null;
BeamDefinition beamDef = null;

//Initializing Objects.
JMenuItem mmItem, ldItem, sdItem, mainItem, chapItem, backItem, printItem;
JMenuItem exitItem, helpItem, liscItem, aboutItem, unitItem;
JButton  microMech, plyMech, macroMech, failureTheory, thinWalled, ld;
JButton exit, back;
JLabel  title, chap4, chap5, chap6, chap7, chap8;


//constructor
public MainPage(){

   //setting title to the Frame
   this.setTitle("Computer Aided Design Environment For Composites
                 (copyright Ever Barbero 1998)");
   //setting size to the Frame
   this.setSize(1000,720);
   //setting back ground Color.
   this.setBackground(Color.lightGray);

   //method used for closing the window.
   this.addWindowListener(new WindowAdapter() {
      public void windowClosing(WindowEvent e){
         MainPage.this.dispose();
      }
   });

   //Creating Panels.
   JPanel northJPanel = new JPanel();
   JPanel centerJPanel = new JPanel();
   JPanel southJPanel = new JPanel();
   //Adding the panels to the container so that they can be seen in // the frame.
   this.getContentPane().setLayout(new BorderLayout());
   getContentPane().add("North", northJPanel);
   getContentPane().add("Center", centerJPanel);
   getContentPane().add("South", southJPanel);

   // Calling the menubars method so that menubar can be set to this frame.
   // menu bar method is later created in the program. It will appear at the bottom
   // of the code.
   menuBars();
```

Source code for MainPage.java (Continued …)

```
//Adding items to the nothJPanel.
northJPanel.setForeground(Color.blue);
northJPanel.add("North", title  = new JLabel("COMPUTER AIDED DESIGN
                        ENVIRONMENT FOR COMPOSITE", JLabel.CENTER));
title.setForeground(Color.blue);
title.setFont(new Font("TimesNewRoman", Font.BOLD, 25));
northJPanel.setLayout(new FlowLayout(FlowLayout.CENTER, 0, 20));
//Creating GridLayout with 1 row and 3 columns. This will enable us to create panels
// with one row and 3 columns and adding them to the centerJPanel.
centerJPanel.setLayout(new GridLayout(1,3));


//Creating a panel centerLeftJPanel and later adding it to centerJPanel.
JPanel centerLeftJPanel = new JPanel();
centerJPanel.add(centerLeftJPanel);

//Creating a panel centerCenterJPanel inside centerJPanel.
JPanel centerCenterJPanel = new JPanel();
centerCenterJPanel.setLayout(new GridLayout(2,1));
centerJPanel.add(centerCenterJPanel);

//Creating a panel centerJPanel1 and later adding it to centerCenterJPanel1.
JPanel centerJPanel1 = new JPanel();
centerCenterJPanel.add(centerJPanel1);
//creating a gridlayout and adding items to it
centerJPanel1.setLayout(new GridLayout(5,1));

//Creating a panel centersubJPanel1 and later adding it to centerJPanel1.
JPanel centersubJPanel1 = new JPanel();
centersubJPanel1.setLayout(new GridLayout(2,1));
centersubJPanel1.add(chap4 = new JLabel("Chapter4", JLabel.RIGHT));
chap4.setFont(new Font("Garamond", Font.BOLD, 15));
chap4.setForeground(color(40,100,5));
centersubJPanel1.add(microMech = new JButton("MicroMechanics"));
microMech.setPreferredSize(size(200,30));
microMech.setForeground(Color.red);
microMech.setFont(new Font("Arial", Font.BOLD, 15));
centerJPanel1.add(centersubJPanel1);
centersubJPanel1.setLayout(new FlowLayout(FlowLayout.LEFT));

//Creating a panel centersubJPanel2 and later adding to centerJPanel1.
JPanel centersubJPanel2 = new JPanel();
centersubJPanel2.setLayout(new GridLayout(2,1));
centersubJPanel2.add(chap5 = new JLabel("Chapter 5", JLabel.RIGHT));
chap5.setFont(new Font("Garamond", Font.BOLD, 15));
chap5.setForeground(color(40,100,5));
centersubJPanel2.add(plyMech = new JButton("Ply Mechanics"));
plyMech.setPreferredSize(size(200,30));
plyMech.setFont(new Font("Arial", Font.BOLD, 15));
plyMech.setForeground(Color.red);
centerJPanel1.add(centersubJPanel2);
centersubJPanel2.setLayout(new FlowLayout(FlowLayout.LEFT));



//Creating a panel centersubJPanel3 and later adding it to centerJPanel1.
JPanel centersubJPanel3 = new JPanel();
centersubJPanel3.setLayout(new GridLayout(2,1));
centersubJPanel3.add(chap6 = new JLabel("Chapter 6", JLabel.RIGHT));
chap6.setFont(new Font("Garamond", Font.BOLD, 15));
chap6.setForeground(color(40,100,5));
centersubJPanel3.add(macroMech = new Button("MacroMechanics"));
macroMech.setPreferredSize(size(200,30));
macroMech.setFont(new Font("Arial", Font.BOLD, 15));
macroMech.setForeground(Color.red);
centerJPanel1.add(centersubJPanel3);
centersubJPanel3.setLayout(new FlowLayout(FlowLayout.LEFT));

//Creating a panel centersubJPanel4 and later adding  to centerJPanel1.
JPanel centersubJPanel4 = new JPanel();
centersubJPanel4.setLayout(new GridLayout(2,1));
centersubJPanel4.add(chap7 = new JLabel("Chapter 7", JLabel.RIGHT));
chap7.setFont(new Font("Garamond", Font.BOLD, 15));
chap7.setForeground(color(40,100,5));
centersubJPanel4.add(failureTheory = new JButton("Failure Theory"));
failureTheory.setPreferredSize(size(200,30));
failureTheory.setFont(new Font("Arial", Font.BOLD, 15));
failureTheory.setForeground(Color.red);
centerJPanel1.add(centersubJPanel4);
centersubJPanel4.setLayout(new FlowLayout(FlowLayout.LEFT));

//Creating a panel centersubJPanel5 and later adding it to centerJPanel1.
JPanel centersubJPanel5 = new JPanel();
centersubJPanel5.setLayout(new GridLayout(2,1));
centersubJPanel5.add(chap8 = new JLabel("Chapter 8    ", JLabel.RIGHT));
chap8.setFont(new Font("Garamond", Font.BOLD, 15));
chap8.setForeground(color(40,100,5));
centersubJPanel5.add(thinWalled = new JButton("Thin Walled Beams"));
thinWalled.setPreferredSize(size(200,30));
thinWalled.setFont(new Font("Arial", Font.BOLD, 15));
thinWalled.setForeground(Color.red);
centerJPanel1.add(centersubJPanel5);
centersubJPanel5.setLayout(new FlowLayout(FlowLayout.LEFT));
```

Source code for MainPage.java (Continued …)

```
//Creating a panel centerRightJPanel and later adding it to the centerJPanel1.
JPanel centerRightJPanel = new JPanel();
centerJPanel.add(centerRightJPanel);

//Adding items to the southPanel.
southJPanel.setLayout(new GridLayout(1,3,200,10));
southJPanel.add(exit = new JButton("Exit"));
exit.setForeground(Color.blue);
southJPanel.add(back = new JButton("Back"));
back.setForeground(Color.blue);
southJPanel.setLayout(new FlowLayout(FlowLayout.CENTER, 30, 10));

/**
 *    The following code implements the action listeners. It tells the GUI
 *    what to do when the particular button is pressed on the current page.
 */

//Adding ActionListener for exit button and implementing it.
exit.addActionListener(new ActionListener() {
   public void actionPerformed(ActionEvent ae){
         MainPage.this.dispose();
   }
});

//Adding ActionListener for microMech button and implementing it.
microMech.addActionListener(new ActionListener(){
   public void actionPerformed(ActionEvent ae){
      mm = new MicroMechanics();
      MainPage.this.setVisible(false);
      mm.setVisible(true);
   }
});

//Adding ActionListener for plyMech button and implementing it.
plyMech.addActionListener(new ActionListener(){
   public void actionPerformed(ActionEvent ae){
      pm = new PlyMechanics();
      MainPage.this.setVisible(false);
      pm.setVisible(true);
   }
});

//Adding ActionListener for macroMech button and implementing it.
macroMech.addActionListener(new ActionListener(){
   public void actionPerformed(ActionEvent ae){
      mam = new MacroMechanics();//MainPage.this);
      MainPage.this.setVisible(false);
```

```
                         mam.setVisible(true);
                        JOptionPane.showMessageDialog(null, "Before Displaying results, Please
                        go to Laminate Definition, build a new Laminate, and Analyse it");
        }
    });

    //Adding ActionListener for failureTheory button and implementing it.
    failureTheory.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent ae){
            ft = new FailureTheory();
            MainPage.this.setVisible(false);
            ft.setVisible(true);
        }
    });


    //Adding ActionListener for thinWalled button and implementing it.
    thinWalled.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent ae){
            tw = new ThinWalledBeams();
            MainPage.this.setVisible(false);
            tw.setVisible(true);
            Object[] possibleValues = { "BeamDefinition", "Stay Here"};
            Object selectedValue = OptionPane.showInputDialog(null,
                                    "Choose one", "Input",
                                    JOptionPane.INFORMATION_MESSAGE,  null,
                                    possibleValues,possibleValues[0]);
        }
    });

}

//***** Main Method ********
public static void main(String args[]){
    MainPage mp = new MainPage();
    mp.setSize(1000,720);
    mp.setBackground(Color.lightGray);
    mp.setVisible(true);
}

//Method to return the specified dimensions.
public Dimension size(int w, int h){
    return new Dimension(w,h);
}

//Method to return the color based on the values of red(x), green(y) and black(z).
//(ranging from 0-225).
public Color color(int x, int y, int z) {
    return new Color(x, y, z);
  }

  //Creating method for the menubars.
  public  JMenuBar menuBars(){

    //Creating an Object of the Menu Bar.
    JMenuBar mb = new JMenuBar();

    //Creating Menu "fileJMenu" and adding different items to it.
    JMenu fileJMenu = new JMenu(" Navigation");
    mmItem = fileJMenu.add(new JMenuItem("MicroMechanics"));
    ldItem = fileJMenu.add(new JMenuItem("Laminate Definition"));
    sdItem = fileJMenu.add(new JMenuItem("Section Definition"));
    fileJMenu.addSeparator();
    mainItem = fileJMenu.add(new JMenuItem("Main"));
    chapItem = fileJMenu.add(new JMenuItem("Chapter"));
    backItem = fileJMenu.add(new JMenuItem("Back"));
    fileJMenu.addSeparator();
    printItem = fileJMenu.add(new JMenuItem("PrintPages"));
    fileJMenu.addSeparator();
    exitItem = fileJMenu.add(new JMenuItem("Exit"));
    mb.add(fileJMenu);//Adding the fileJMenu to the MenuBar.
    setJMenuBar(mb);//Setting the MenuBar to the frame.

    //Creating Menu "helpJMenu" and adding items to it.
    JMenu helpJMenu = new JMenu("Help");
    helpItem = helpJMenu.add(new JMenuItem("Page Help F1"));
    aboutItem = helpJMenu.add(new JMenuItem("About"));
    liscItem = helpJMenu.add(new JMenuItem("License"));
    unitItem = helpJMenu.add(new JMenuItem("Units"));
    mb.add(helpJMenu);//Adding the helpJMenu to the MenuBar.
    setJMenuBar(mb);//Setting the MenuBar to the frame.

    //Creating shortcuts to the menus
    fileJMenu.setMnemonic('N');
    helpJMenu.setMnemonic('H');

    //Adding File Menu Accelerators. This will enable the shortcuts to the key letter
    // specified when the program is running.
    mmItem.setAccelerator(KeyStroke.getKeyStroke('M', Event.CTRL_MASK));
    ldItem.setAccelerator(KeyStroke.getKeyStroke('L', Event.CTRL_MASK));
    sdItem.setAccelerator(KeyStroke.getKeyStroke('S', Event.CTRL_MASK));
    mainItem.setAccelerator(KeyStroke.getKeyStroke('A', Event.CTRL_MASK));
    chapItem.setAccelerator(KeyStroke.getKeyStroke('C', Event.CTRL_MASK));
    backItem.setAccelerator(KeyStroke.getKeyStroke('B', Event.CTRL_MASK));
    printItem.setAccelerator(KeyStroke.getKeyStroke('P', Event.CTRL_MASK));
    exitItem.setAccelerator(KeyStroke.getKeyStroke('E', Event.ALT_MASK ));
```

Source code for MainPage.java (Continued …)

```
/* *
 * The following action listeners will tell the software what to do when the
 * particular key is pressed while the program is running.
 */

//Adding action listener to the mmitem and implementing it.
//mmItem is the shortcut key for MicroMechanics Update Page.
mmItem.addActionListener(new ActionListener(){
   public void actionPerformed(ActionEvent ae){
      mmup1 = new MicroMechUpdatePage1();
      MainPage.this.setVisible(false);
      mmup1.settingValues();
      mmup1.setVisible(true);
   }
});

//Adding action listener to the lditem and implementing it.
//mmItem is the shortcut key for Laminate Definition Page.
ldItem.addActionListener(new ActionListener(){
   public void actionPerformed(ActionEvent ae){
      lamDef = new LaminateDefinition();
      MainPage.this.setVisible(false);
      //lamDef.settingValues();
      lamDef.setVisible(true);
   }
});

//Adding action listener to the sditem and implementing it.
//mmItem is the shortcut key for Beam Definition Page.
sdItem.addActionListener(new ActionListener(){
   public void actionPerformed(ActionEvent ae){
      beamDef = new BeamDefinition();
      MainPage.this.setVisible(false);
      beamDef.setVisible(true);
   }
});

//Adding action listener to the printitem and implementing it.
//mmItem is the shortcut key for Printing Pages.
printItem.addActionListener(new ActionListener(){
   public void actionPerformed(ActionEvent ae){
      JOptionPane.showMessageDialog(null, "Button Not Working Yet");
   }
});
```

```
//Adding action listener to the exititem and implementing it.
//mmItem is the shortcut key for Exiting this Application.
exitItem.addActionListener(new ActionListener(){
   public void actionPerformed(ActionEvent ae){
      MainPage.this.dispose();
   }
});

//Adding actionListener to the backItem and implementing it.
//backItem is the shortcut key for the previous class visited.
backItem.addActionListener(new ActionListener(){
   public void actionPerformed(ActionEvent ae){
      JOptionPane.showMessageDialog(null, "Currently working on it");
   }
});

//******Adding ActionListener to the Help Menu's

//Adding actionlistener to the helpItem and implementing it.
//helpItem is the shortcut key for Help item in this Application.
helpItem.addActionListener(new ActionListener(){
   public void actionPerformed(ActionEvent ae){
      JOptionPane.showMessageDialog(null, "Button Not Working Yet");
   }
});



//Adding actionlistener to the liscItem and implementing it.
//liscItem is the shortcut key for Liscense in this Application.
liscItem.addActionListener(new ActionListener(){
   public void actionPerformed(ActionEvent ae){
      JOptionPane.showMessageDialog(null, "Button Not Working Yet");
   }
});

//Adding actionlistener to the helpItem and implementing it.
//helpItem is the shortcut key for Help item in this Application.
aboutItem.addActionListener(new ActionListener(){
   public void actionPerformed(ActionEvent ae){
      JOptionPane.showMessageDialog(null, "Button Not Working Yet");
   }
});

//Adding actionlistener to the helpItem and implementing it.
//helpItem is the shortcut key for Help item in this Application.
unitItem.addActionListener(new ActionListener(){
   public void actionPerformed(ActionEvent ae){
```

```
                JOptionPane.showMessageDialog(null, "Button Not Working Yet");
            }
        });

        //returning the menu bar value.
        return mb;

    }

}
```

Chapter 5. Problem Solving

In this chapter the problem solving capabilities of the software was tested by solving examples from Chapters 4-6 (Micromechanics, Ply Mechanics, Macromechanics, Failure Theory and Thin - Walled Beams) of [7], and comparing the results obtained using the software.

5.1. Micromechanics:

For Chapter 4 the example problem 4.6 on page 94 – 95 is being considered. The description of the problem is as follows.

5.1.1. Problem:

Estimate the effect of misalignment on the compressive strength of a carbon-PEEK composite fabricated by hand lay-up of prepreg tape, vacuum bagged and oven cured. Sample A was fabricated under controlled laboratory conditions, resulting in a standard deviation of misalignment Ω_A *= 1.01 degrees. Sample B was fabricated under normal processing conditions, resulting in* Ω_B *= 1.41 degrees. The fiber is carbon AS4-D and the matrix is epoxy with* V_f *= 0.6.*

From Tables 2.1 and 2.4, the properties of the carbon AS4-D fiber and the epoxy matrix (9310/9360 at 23^0 C) are E_f = 241 GPa, υ_f = 0.2 (assumed), E_m = 3.12 GPa, υ_m = 0.38 (assumed), σ_{mu} = 75.8 MPa. The compressive strength is given by (4.74) and (4.75). Therefore, the inplane shear stiffness and strength must be determined first.

5.1.1.1. Solution from the Textbook:

Assuming both constituents to be isotropic, which is an approximation for carbon fiber, the shear moduli are.

$$G_f = \frac{241}{2(1+0.2)} = 100\ GPa.$$

$$G_m = \frac{3.12}{2(1+0.38)} = 1.13\ GPa.$$

Using Inplane shear modulus, the cylindrical assemblage model (CAM), formula

$$G_{12} = \left(\frac{(1+0.6)+(1-0.6)1.13/100}{(1-0.6)+(1+0.6)1.13/100}\right) = 4.34\ GPa.$$

The transverse modulus (E_2) needs to be evaluated first before calculating the Inplane shear strength. Using the equation transverse modulus equation from the text book,

$$\eta = \frac{241/3.12-1}{241/3.12+2} = 0.962$$

$$E_2 = 3.12\left[\frac{1+2*0.962*0.6}{1-0.962*0.6}\right] = 15.9\ GPa.$$

Then using the Inplane shear strength equation, and assuming $\tau_{mu} = \sigma_{mu}$ *and no voids* ($C_v = 1.0$).

$$F_6 = 75.8\left[1+\left(0.6-\sqrt{0.6}\right)\left(1-\frac{1.13}{100}\right)\right] = 62.7\ MPa$$

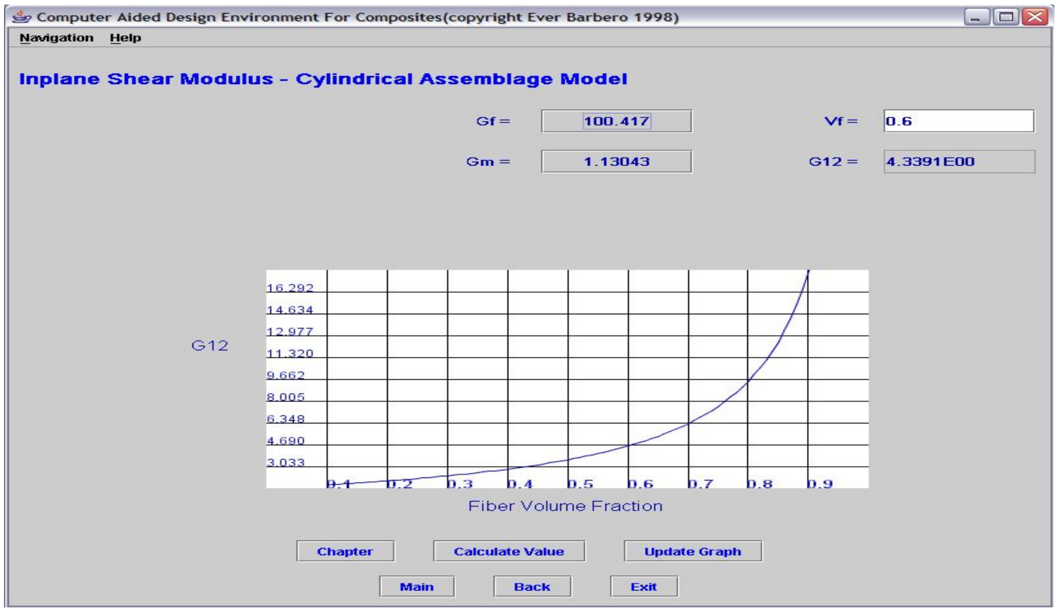
For sample A, using the Longitudinal Compressive Strength formulae,

$$\chi = \frac{4{,}340 * \frac{\pi}{180} * 1.01}{62.7} = 1.22$$

$$F_{1c} = 4{,}340\left(\frac{1.22}{0.21}+1\right)^{-0.69} = 1{,}155\ MPa = 1.155\ GPa\ .$$

5.1.1.2. Solution Using Software:

Using the same material properties the problem is now solved using the software tool and each calculated value is being shown in a separate GUI.


Computer Aided Design Environment For Composites(copyright Ever Barbero 1998)

Inplane Shear Modulus - Cylindrical Assemblage Model

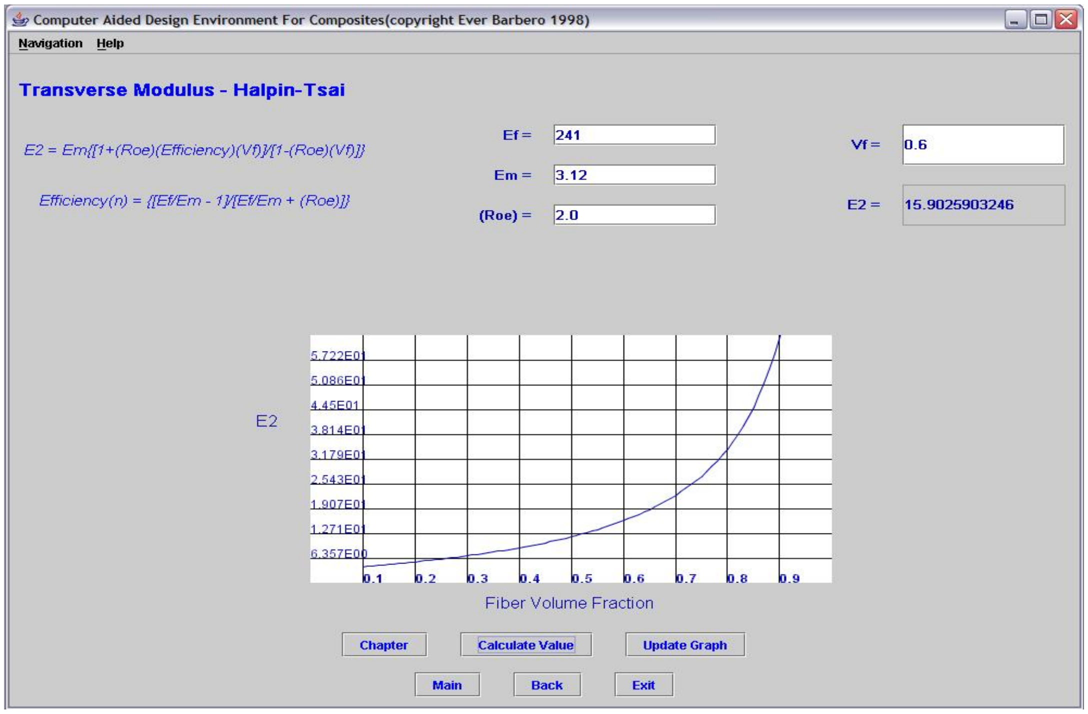
Gf = 100.417

Vf = 0.6

Gm = 1.13043

G12 = 4.3391E00

Figure 5.1 Result Page for Inplane Shear Modulus G_{12} along with calculated values of G_f and G_m



Transverse Modulus - Halpin-Tsai

E2 = Em{[1+(Roe)(Efficiency)(Vf)]/[1-(Roe)(Vf)]}

Efficiency(n) = {[Ef/Em - 1]/[Ef/Em + (Roe)]}

Ef = 241

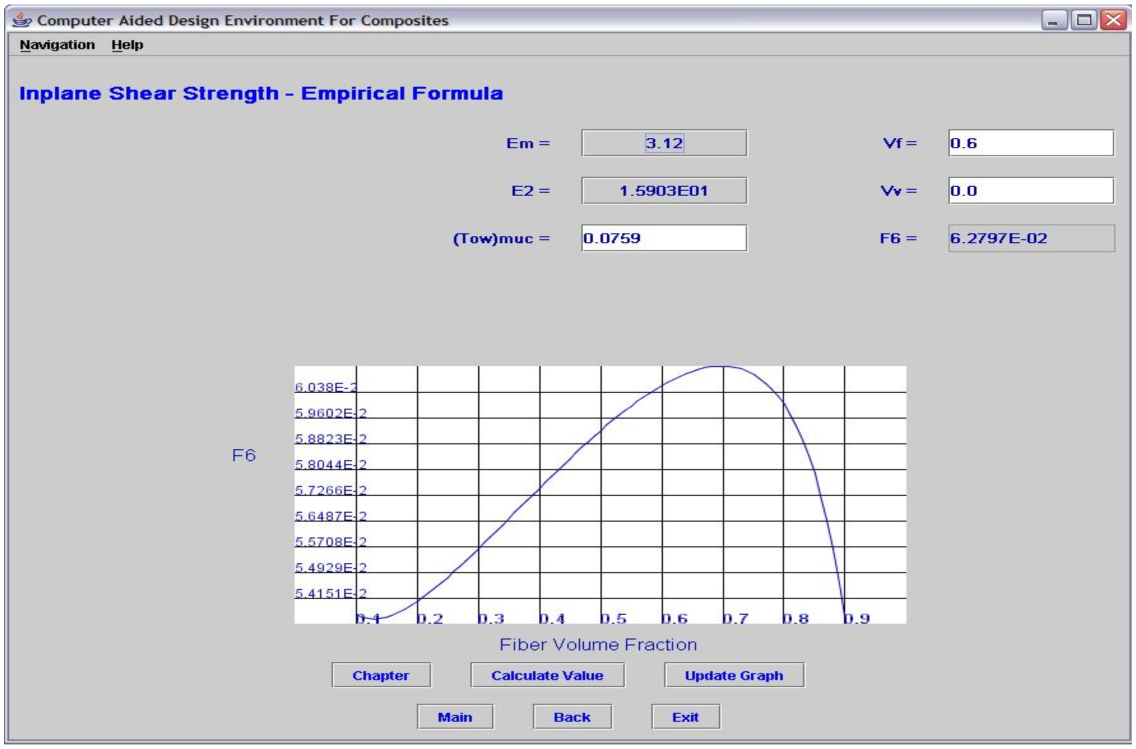
Em = 3.12

(Roe) = 2.0

Vf = 0.6

E2 = 15.9025903246

Figure 5.2 Result Page for Transverse Modulus – Halpin Tsai (E_2).



Inplane Shear Strength - Empirical Formula

Em = 3.12

E2 = 1.5903E01

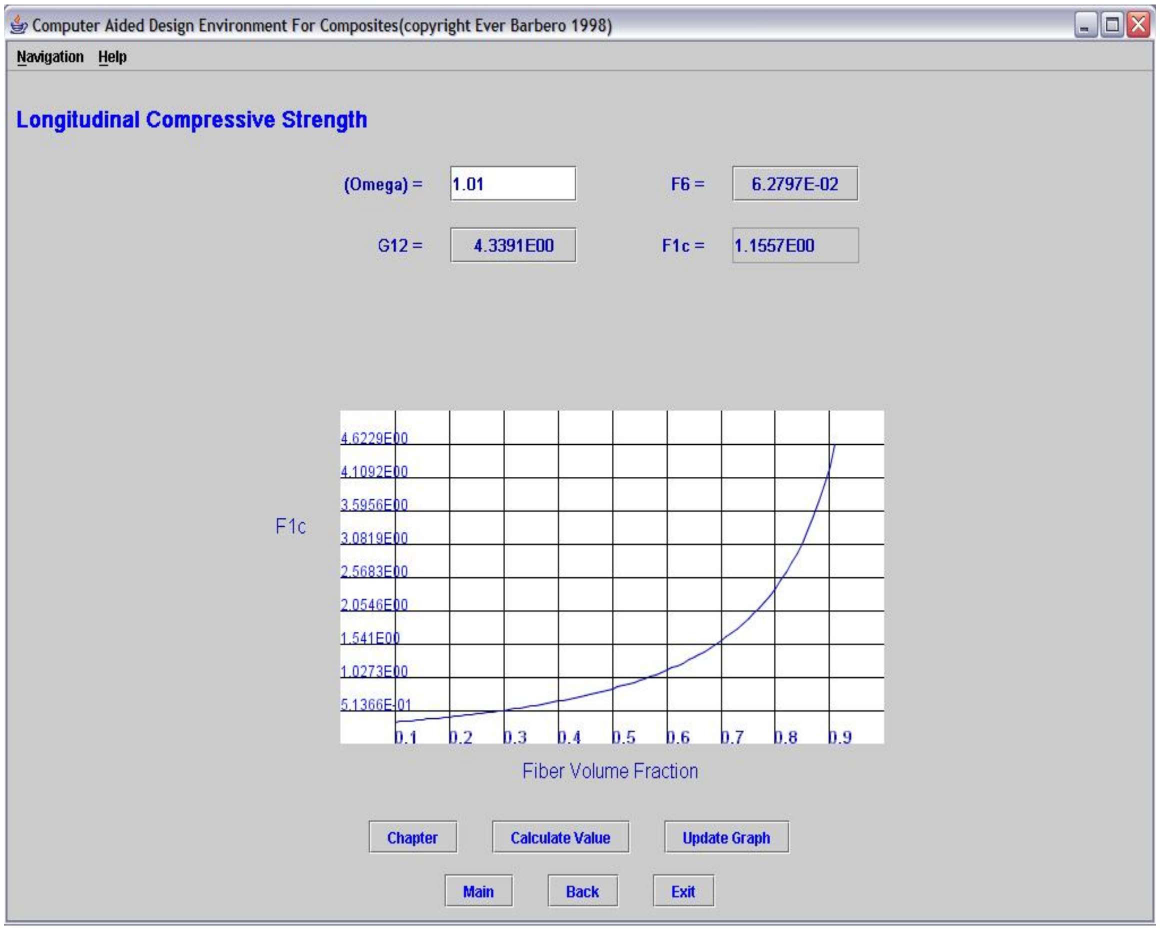
(Tow)muc = 0.0759

Vf = 0.6

Vv = 0.0

F6 = 6.2797E-02

Figure 5.3 Result page for Inplane Shear Strength F_6 .



Longitudinal Compressive Strength

(Omega) = 1.01

F6 = 6.2797E-02

G12 = 4.3391E00

F1c = 1.1557E00

Figure 5.4 Results page for Longitudinal Compressive Strength (F_{1c}).

The result values obtained from the software match the results from the example problem.

5.2. Ply Mechanics:

For Chapter 5 Ply mechanics the example problem No. 5.4 on page 120 – 121 is being chosen. The description of the problem is as follows.

5.2.1. Problem:

Transform the strains $\in_1 = 3.6535(10^{-3})$, $\in_2 = 7.411(10^{-3})$, $\gamma_4 = \gamma_5 = \gamma_6 = 0$ *to a global coordinate system with the x-axis oriented at* 55^0 *counterclockwise (ccw) with respect to the material axis corresponding to the fiber direction (1-axis).*

Note that the shear strain must be divided by 2 before it can be transformed.

5.2.1.1. Solution from the Textbook:

The relationship between global and material strain is given by

$$\begin{Bmatrix} \in_x \\ \in_y \\ \frac{1}{2}\gamma_{xy} \end{Bmatrix} = \begin{Bmatrix} m*m & n*n & -2mn \\ n*n & m*m & 2mn \\ mn & -mn & (m*m)-(n*n) \end{Bmatrix} \begin{Bmatrix} \in_1 \\ \in_2 \\ \frac{1}{2}\gamma_6 \end{Bmatrix}$$

Where m = cos (θ), n = sin(θ).

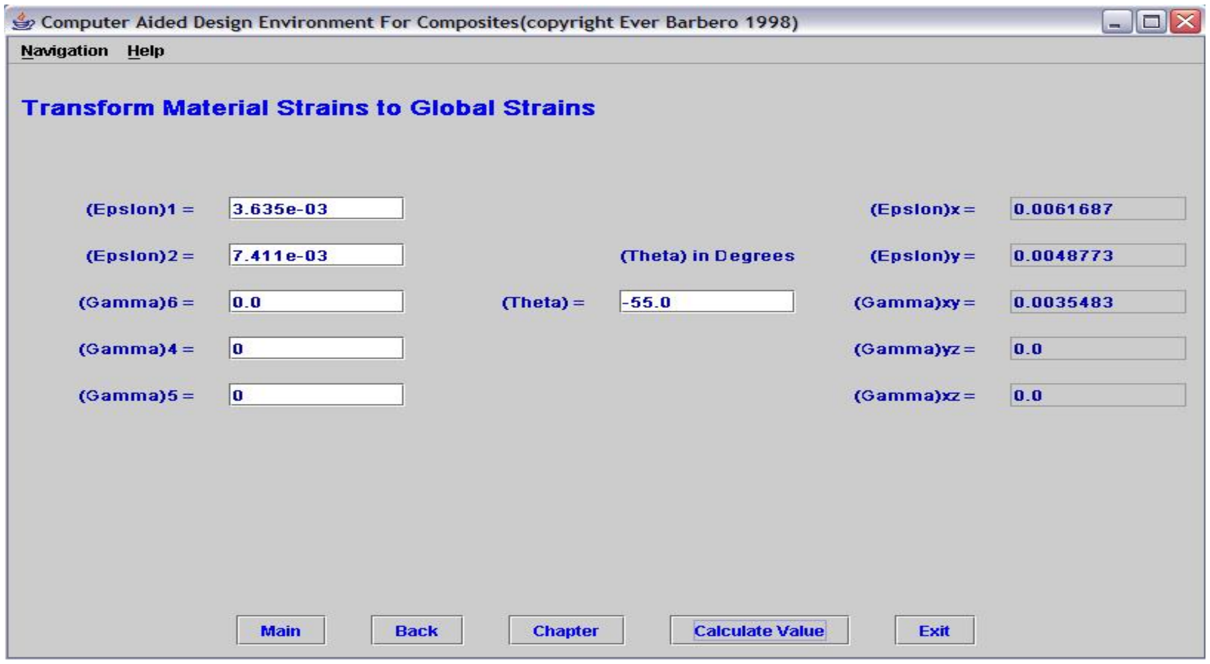
Therefore substituting the values and calculating, we get result values as

$$\begin{Bmatrix} \in_x \\ \in_y \\ \frac{1}{2}\gamma_{xy} \end{Bmatrix} = \begin{bmatrix} 0.329 & 0.671 & 0.940 \\ 0.671 & 0.329 & -0.940 \\ -0.470 & 0.470 & -0.342 \end{bmatrix} \begin{Bmatrix} 3.635 \\ 7.411 \\ \frac{1}{2}*0 \end{Bmatrix} 10^{-3} = \begin{Bmatrix} 6.169 \\ 4.877 \\ 1.774 \end{Bmatrix} 10^{-3}$$

Then $\in_x = 6.169\ (10^{-3})$, $\in_y = 4.877\ (10^{-3})$, and $\gamma_{xy} = 2\ (1.774)10^{-3} = 3.547(10^{-3})$

5.2.1.2. Solution Using Software:

The problem is now solved by the software by entering the material properties provided in the problem 5.4. Figure 5.5 shows the GUI page for the Transformation of strains from Material ($\in$) to Global co-ordinate system along with the results.


Computer Aided Design Environment For Composites(copyright Ever Barbero 1998)
Navigation
Help
Main
Back
Chapter
Calculate Value
Exit

Transform Material Strains to Global Strains

```
(Epslon)1 =  3.635e-03                                  (Epslon)x =   0.0061687
(Epslon)2 =  7.411e-03         (Theta) in Degrees       (Epslon)y =   0.0048773
(Gamma)6 =   0.0      (Theta) = -55.0                   (Gamma)xy =   0.0035483
(Gamma)4 =   0                                          (Gamma)yz =   0.0
(Gamma)5 =   0                                          (Gamma)xz =   0.0
```

Figure 5.5 Results page for Strain Transformation from Material to Global

5.3. Macromechanics:

For Chapter 6 the example problem 6.1 on pages 140 – 141 is being considered. The description of the problem is as follows.

5.3.1. Problem:

Compute the coefficients in the plate stiffness equations (6.9) for a two-layer laminate with $\theta_1 = 55^0$, $\theta_2 = -55^0$, $t_1 = t_2 = 0.635$ mm, with material properties given as $E_1 = 19.981$ GPa, $\nu_{12} = 0.274$, $E_2 = 11.389$ GPa, $G_{12} = G_{13} = 3.789$ GPa. And assume the out-of-plane shear modulus $G_{23} \cong G_m = 0.385$ GPa.

5.3.1.1. Solution from the Text Book:

Firstly the reduced stiffness matrix$[Q]_{3x3}$ is calculated by using the equations,

$$Q_{11} = E_1 / \Delta$$

$$Q_{12} = Q_{21} = \nu_{12} E_2 / \Delta$$

$$Q_{22} = E_2 / \Delta$$

$$Q_{66} = G_{12}$$

$$Q^*_{44} = G_{23}$$

$$Q^*_{55} = G_{13}$$

$$\Delta = 1- \upsilon_{12}\, \upsilon_{21}$$

Substituting the values and calculating will yield the results of matrix Q and Q as*

$$Q = \begin{bmatrix} 20.874 & 3.260 & 0 \\ 3.260 & 11.898 & 0 \\ 0 & 0 & 3.798 \end{bmatrix} GPa.$$

$$Q^* = \begin{bmatrix} 0.385 & 0 \\ 0 & 3.789 \end{bmatrix}$$

Computing the transformed reduced stiffness matrices in the two layers at $\theta = 55^0$

$$[\overline{Q}]^{(1)} = \begin{bmatrix} 12.40 & 5.71 & 1.22 \\ 5.71 & 15.50 & 3.00 \\ 1.22 & 3.00 & 6.24 \end{bmatrix} GPa$$

and at $\theta = -55^0$

$$[\overline{Q}]^{(2)} = \begin{bmatrix} 12.40 & 5.71 & -1.22 \\ 5.71 & 15.50 & -3.00 \\ -1.22 & -3.00 & 6.24 \end{bmatrix} GPa$$

Similarly the values of Q^* at $\theta = 55^0$

$$[\overline{Q}]_{44}{}^{*(1)} = 2.669$$

$$[\overline{Q}]_{45}{}^{*(1)} = 1.599$$

$$\overline{[Q]}_{55}\,{*}^{(1)} = 1.505$$

and at $\theta = -55^0$

$$\overline{[Q]}_{44}\,{*}^{(2)} = 2.669$$

$$\overline{[Q]}_{45}\,{*}^{(2)} = -1.599$$

$$\overline{[Q]}_{55}\,{*}^{(2)} = 1.505$$

where, the superscript 1 & 2 indicates the layer number. Then using the Plain Stiffness and compliance equations the matrices A, B, D and H are computed. The matrices values are found to be as.

$$[A] = \begin{bmatrix} 15.8 & 7.25 & \varepsilon \\ 7.25 & 19.60 & \varepsilon \\ \varepsilon & \varepsilon & 6.24 \end{bmatrix} GPa\ \ mm$$

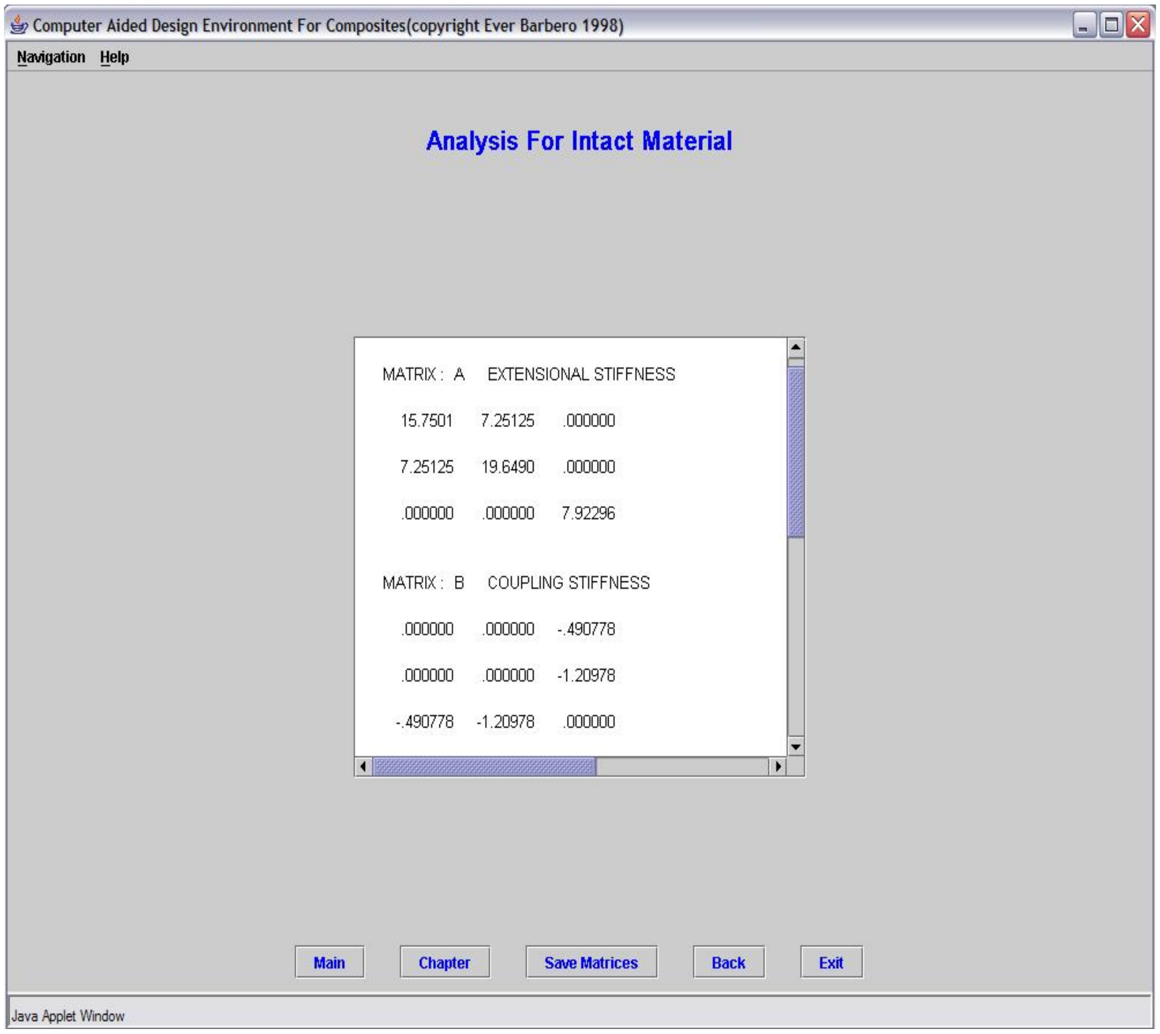
$$[B] = \begin{bmatrix} \varepsilon & \varepsilon & -0.491 \\ \varepsilon & \varepsilon & -1.21 \\ -0.491 & -1.21 & \varepsilon \end{bmatrix} GPa\ \ mm^2$$

$$[D] = \begin{bmatrix} 2.12 & 0.975 & \varepsilon \\ 0.975 & 2.64 & \varepsilon \\ \varepsilon & \varepsilon & 1.06 \end{bmatrix} GPa\ \ mm^3$$

$$[H] = \begin{bmatrix} 2.82 & 0 \\ 0 & 1.59 \end{bmatrix}$$

5.3.1.2. Solution Using Software:

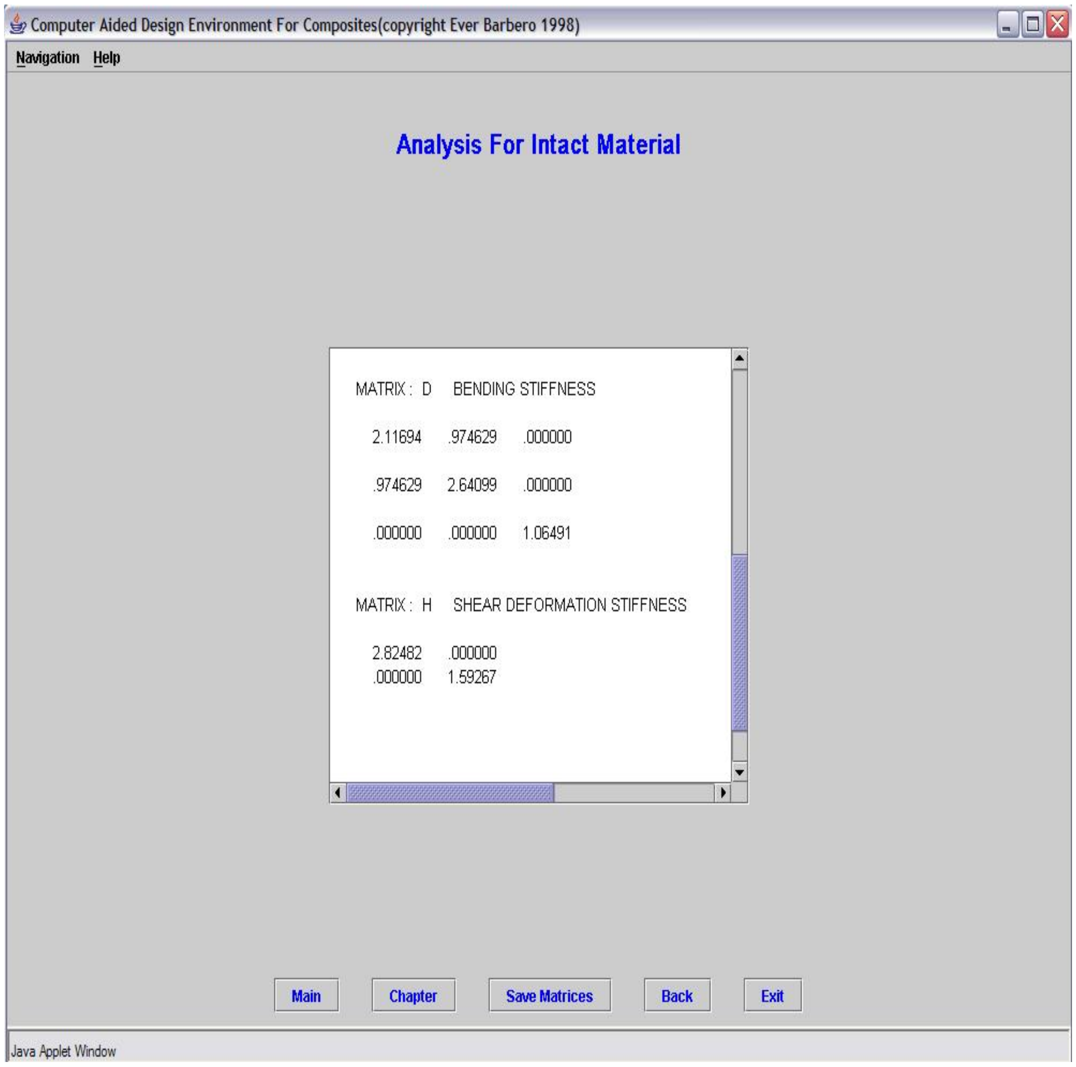
The following GUI's show the calculated values for Matrices A, B, D and H.



Computer Aided Design Environment For Composites(copyright Ever Barbero 1998)
Navigation
Help
Analysis For Intact Material
Main
Chapter
Save Matrices
Back
Exit
Java Applet Window

```
MATRIX :  A     EXTENSIONAL STIFFNESS

  15.7501    7.25125    .000000

  7.25125    19.6490    .000000

  .000000    .000000    7.92296

MATRIX :  B     COUPLING STIFFNESS

  .000000    .000000    -.490778

  .000000    .000000    -1.20978

 -.490778   -1.20978    .000000
```

Figure 5.6 Result page showing Matrices A & Matrix B.


Computer Aided Design Environment For Composites(copyright Ever Barbero 1998)
Navigation
Help
Main
Chapter
Save Matrices
Back
Exit
Java Applet Window

Analysis For Intact Material

```
MATRIX :  D     BENDING STIFFNESS

   2.11694    .974629    .000000

   .974629    2.64099    .000000

   .000000    .000000    1.06491


MATRIX :  H     SHEAR DEFORMATION STIFFNESS

   2.82482    .000000
   .000000    1.59267
```

Figure 5.7 Result page showing Matrices D & Matrix H.

5.4. Failure Theory:

In Chapter 7 the problem number 7.4 is being chosen. The description of the problem is as follows

5.4.1. Problem:

Using the maximum stress criterion, compute the FPF load of a $[0/90]_s$ laminate subject to N_x = 175,000 N/m. Each layer is 2.54 mm thick. The material properties are

$$E_1 = 54\ GPa,\ E_2 = 18 GPa,\ G_{12} = 9\ GPa,\ \upsilon_{12} = .25$$

$$F_{1t} = 1{,}034\ MPa,\ F_{1c} = 1{,}034\ MPa,\ F_{2t} = 31\ MPa,$$

$$F_{2c} = 138\ MPa,\ F_6 = 41\ MPa,$$

5.4.1.1. Solution from the Text Book:

Since the laminate is a symmetric cross-ply and only inplane loads are present, therefore, only the [A] matrix is necessary. Using the Macromechanics analysis the value of Q matrix is found to be.

$$[Q] = \begin{bmatrix} 55.1 & 4.6 & 0 \\ 4.6 & 18.4 & 0 \\ 0 & 0 & 9 \end{bmatrix} GPa.$$

Now using the Plane Stiffness Equation the value of Matrix A is found to be

$$[A] = \begin{bmatrix} 374 & 46.7 & 0 \\ 46.7 & 374 & 0 \\ 0 & 0 & 91.44 \end{bmatrix} MPa\ m$$

Note that the laminate has the same amount of material oriented in the 0-degree and 90-degree directions, so $A_{11} = A_{22}$. Only two of the middle-surface strains are different from zero, which can be computed using equation 6.18, which is

$$\begin{Bmatrix} \in_x^0 \\ \in_y^0 \\ \frac{1}{2}\gamma_{xy}^0 \end{Bmatrix} = \begin{bmatrix} \alpha_{11} & \alpha_{12} & \alpha_{16} \\ \alpha_{12} & \alpha_{22} & \alpha_{26} \\ \alpha_{16} & \alpha_{26} & \alpha_{66} \end{bmatrix} \begin{Bmatrix} N_x \\ N_y \\ N_{xy} \end{Bmatrix}$$

Using the equation the values are found to be as

$$\in_x^0 = 476\ (10^{-6}) \text{ and } \in_y^0 = -59.5\ (10^{-6}).$$

These are also the only strains throughout the laminate since the curvatures are zero. Since there are only two nonzero stresses, they can be easily tabulated along with the values of the strength ratios using Maximum Stress Criterion.

Ply	σ_1 *[MPa]*	σ_2 *[MPa]*	R_1	R_2
OUTER	*26.0*	*1.09*	*39.8*	*28.4*
MIDDLE	*-1.09*	*8.48*	*948.6*	*3.66*

The top and bottom layers are 0-degree plies, which experience the same stress. The two middle layers are the 90-degree plies. The minimum value is R = 3.66. Therefore, the FPF is

$$N_{FPF} = 175{,}000(3.66) = 640\ KN/m.$$

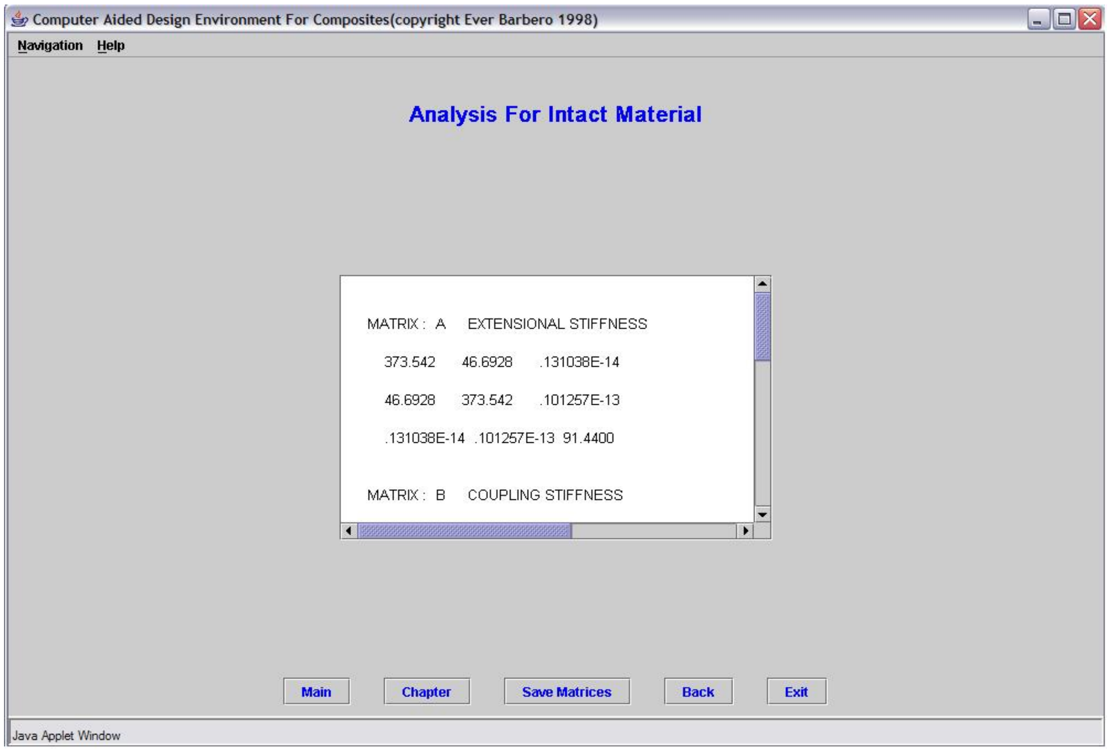
At his load, the two middle layers experience some kind of transverse failure. The stress in the middle layer at failure is

$$\sigma_2 = 8.48(3.66) = 31\ MPa.$$

this, of course, is the transverse strength of the material.

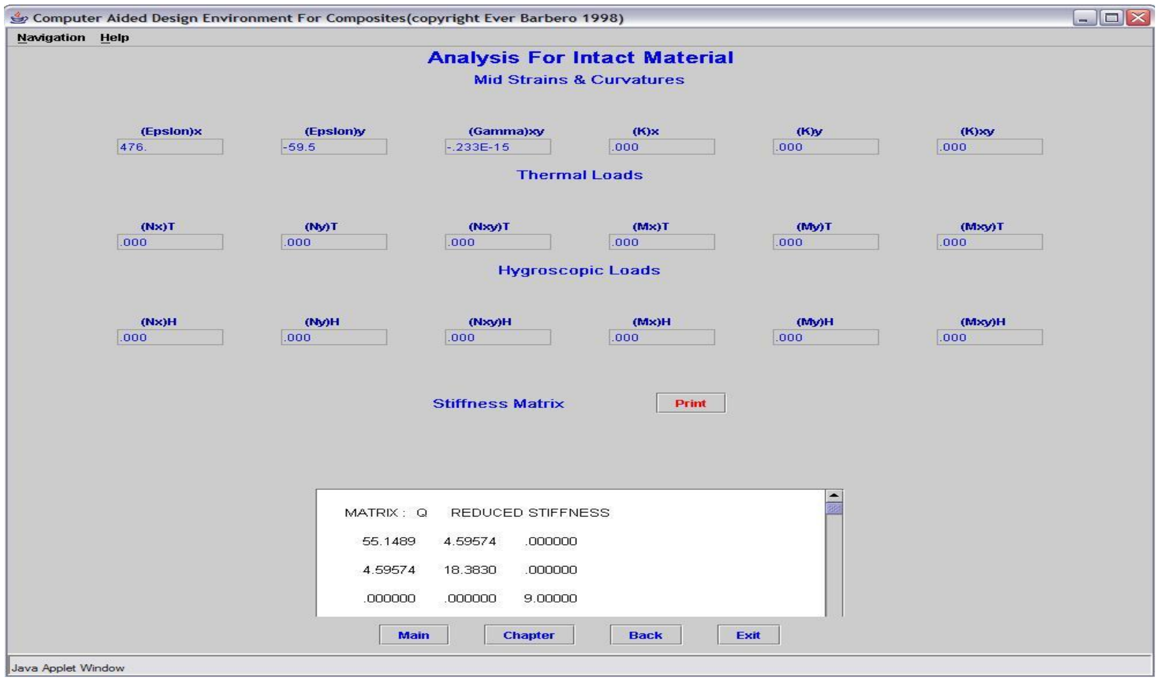
5.4.1.2. Solution Using Software:

The material properties are entered on the Laminate Definition page. The laminate is then analyzed and the results are obtained. The following GUI pages show the results for the problem.


Computer Aided Design Environment For Composites(copyright Ever Barbero 1998)
Navigation Help
Analysis For Intact Material
Main
Chapter
Save Matrices
Back
Exit
Java Applet Window

```
MATRIX :  A     EXTENSIONAL STIFFNESS

   373.542      46.6928      .131038E-14

   46.6928      373.542      .101257E-13

   .131038E-14  .101257E-13  91.4400

MATRIX :  B     COUPLING STIFFNESS
```

Figure 5.8 GUI showing the Matrix A



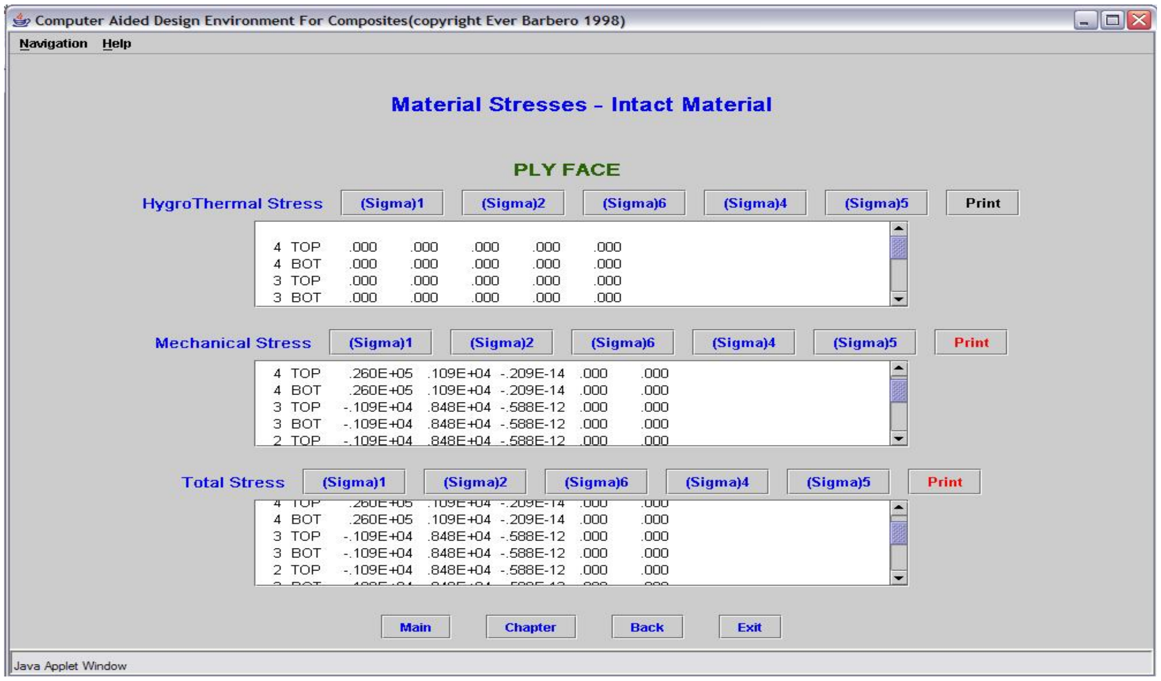
(Epslon)x	(Epslon)y	(Gamma)xy	(K)x	(K)y	(K)xy
476.	-59.5	-.233E-15	.000	.000	.000

(Nx)T	(Ny)T	(Nxy)T	(Mx)T	(My)T	(Mxy)T
.000	.000	.000	.000	.000	.000

(Nx)H	(Ny)H	(Nxy)H	(Mx)H	(My)H	(Mxy)H
.000	.000	.000	.000	.000	.000

```
MATRIX :  Q     REDUCED STIFFNESS

   55.1489    4.59574    .000000

   4.59574    18.3830    .000000

   .000000    .000000    9.00000
```

Figure 5.9 shows the [Q] matrix values along with the values of $\in_x^0$ and $\in_y^0$.



HygroThermal Stress

```
 4  TOP    .000    .000    .000    .000    .000
 4  BOT    .000    .000    .000    .000    .000
 3  TOP    .000    .000    .000    .000    .000
 3  BOT    .000    .000    .000    .000    .000
```

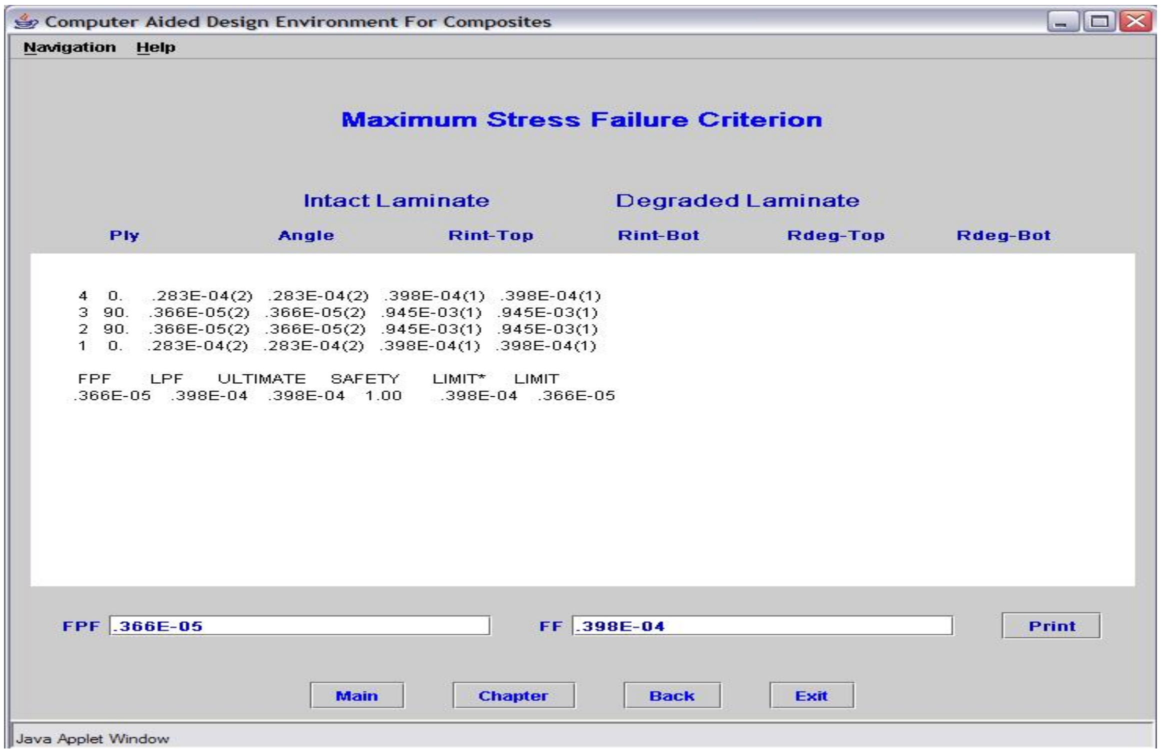
Mechanical Stress

```
 4  TOP    .260E+05  .109E+04 -.209E-14  .000    .000
 4  BOT    .260E+05  .109E+04 -.209E-14  .000    .000
 3  TOP   -.109E+04  .848E+04 -.588E-12  .000    .000
 3  BOT   -.109E+04  .848E+04 -.588E-12  .000    .000
 2  TOP   -.109E+04  .848E+04 -.588E-12  .000    .000
```

Total Stress

```
 4  TOP    .260E+05  .109E+04 -.209E-14  .000    .000
 4  BOT    .260E+05  .109E+04 -.209E-14  .000    .000
 3  TOP   -.109E+04  .848E+04 -.588E-12  .000    .000
 3  BOT   -.109E+04  .848E+04 -.588E-12  .000    .000
 2  TOP   -.109E+04  .848E+04 -.588E-12  .000    .000
```

Figure 5.10 shows the values of the stresses calculated.



```
Ply        Angle       Rint-Top      Rint-Bot      Rdeg-Top      Rdeg-Bot

 4   0.   .283E-04(2)  .283E-04(2)  .398E-04(1)  .398E-04(1)
 3  90.   .366E-05(2)  .366E-05(2)  .945E-03(1)  .945E-03(1)
 2  90.   .366E-05(2)  .366E-05(2)  .945E-03(1)  .945E-03(1)
 1   0.   .283E-04(2)  .283E-04(2)  .398E-04(1)  .398E-04(1)

 FPF       LPF      ULTIMATE   SAFETY    LIMIT*    LIMIT
.366E-05  .398E-04  .398E-04   1.00     .398E-04  .366E-05
```

FPF .366E-05   FF .398E-04

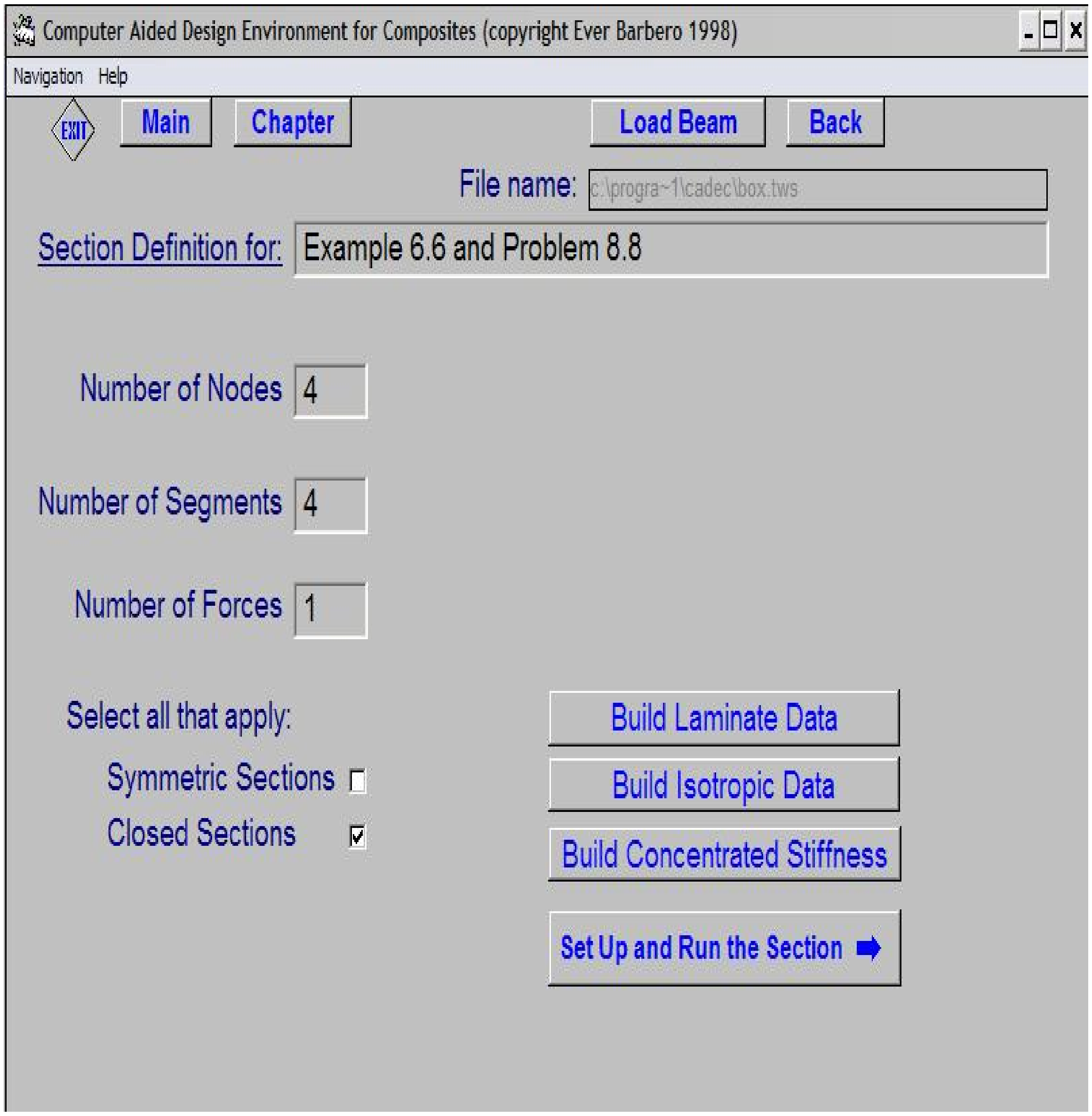
Figure 5.11 shows the Result page using Maximum Stress Failure Criterion.

5.5. Thin Walled Beams:

In Chapter 8 the Toolbook version of CADEC is chosen. It has some predefined files which would can be loaded and analyzed. Therefore the sample file “BOX.TWS” is loaded and analyzed. The same material values are then entered in the new software and analyzed. The results obtained are then compared.

5.5.1. Solution Using Old Version of CADEC:

Figure 5.12 shows the GUI page for “Beam Definition”. This page provides the option of loading a Beam file. A button is provided namely “Load Beam” which enables the user to load files. The files for loading beams are usually in the “**.TWS**” format. The file “BOX.TWS” comes along with the original CADEC software. Once the file is loaded one can go Beam analysis page by clicking on the button “Set Up and Run the Section”. Figure 5.13 shows the GUI with values loaded from file “BOX.TWS”. The beam can be analyzed by clicking on the button “Run Section”. After the beam is analyzed successfully the results for this section can be viewed. Figure 5.14 shows the Result page for Segment Stiffness per Unit Length.


Computer Aided Design Environment for Composites (copyright Ever Barbero 1998)
Navigation
Help
EXIT
Main
Chapter
Load Beam
Back
Build Laminate Data
Build Isotropic Data
Build Concentrated Stiffness
Set Up and Run the Section

File name: c:\progra~1\cadec\box.tws

Section Definition for: Example 6.6 and Problem 8.8

Number of Nodes: 4

Number of Segments: 4

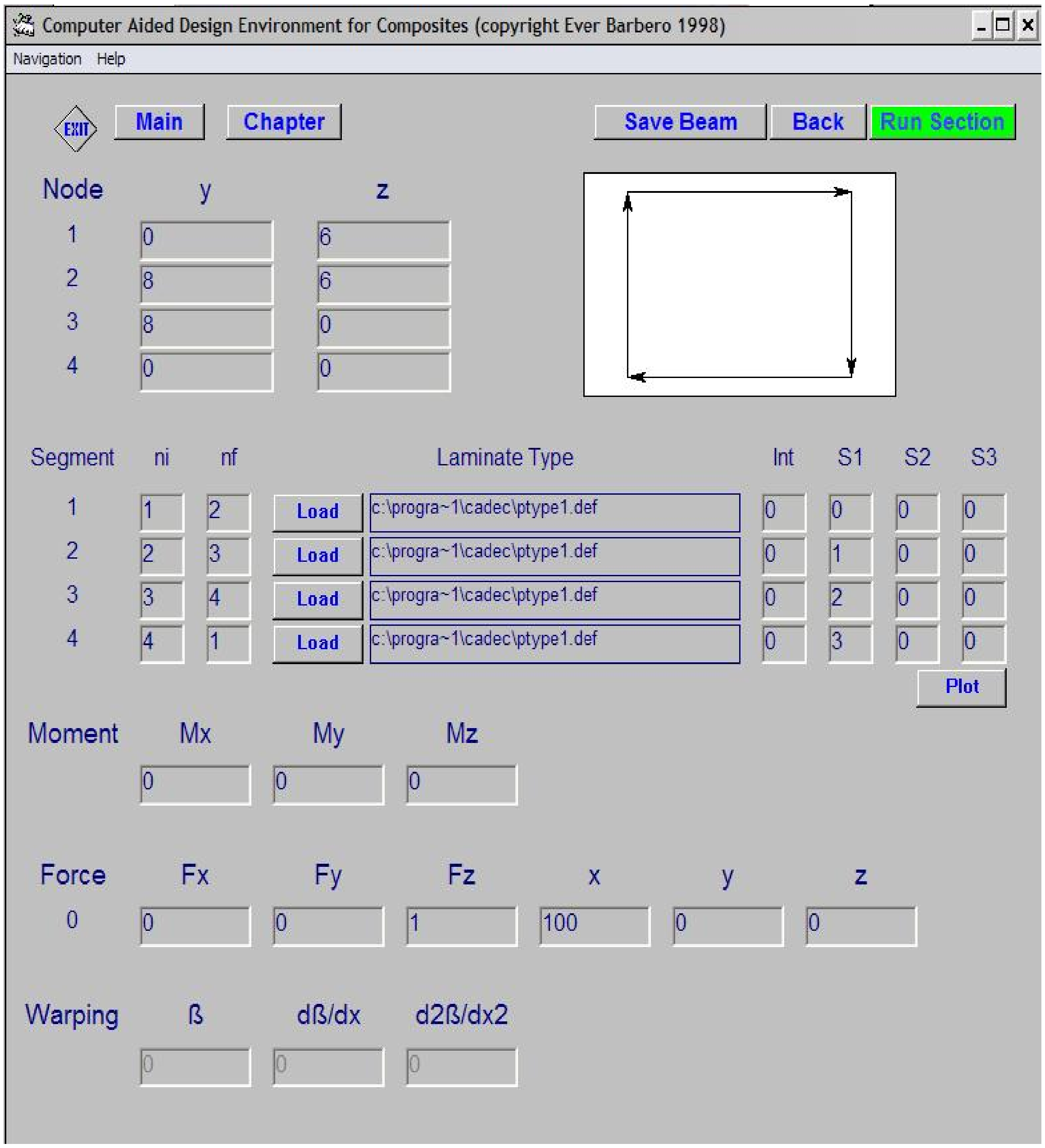
Number of Forces: 1

Select all that apply:

Symmetric Sections ☐

Closed Sections ☑

Figure 5.12 GUI page for Beam Definition


Computer Aided Design Environment for Composites (copyright Ever Barbero 1998)
Navigation Help
EXIT
Main
Chapter
Save Beam
Back
Run Section
Load
Plot

Node	y	z
1	0	6
2	8	6
3	8	0
4	0	0

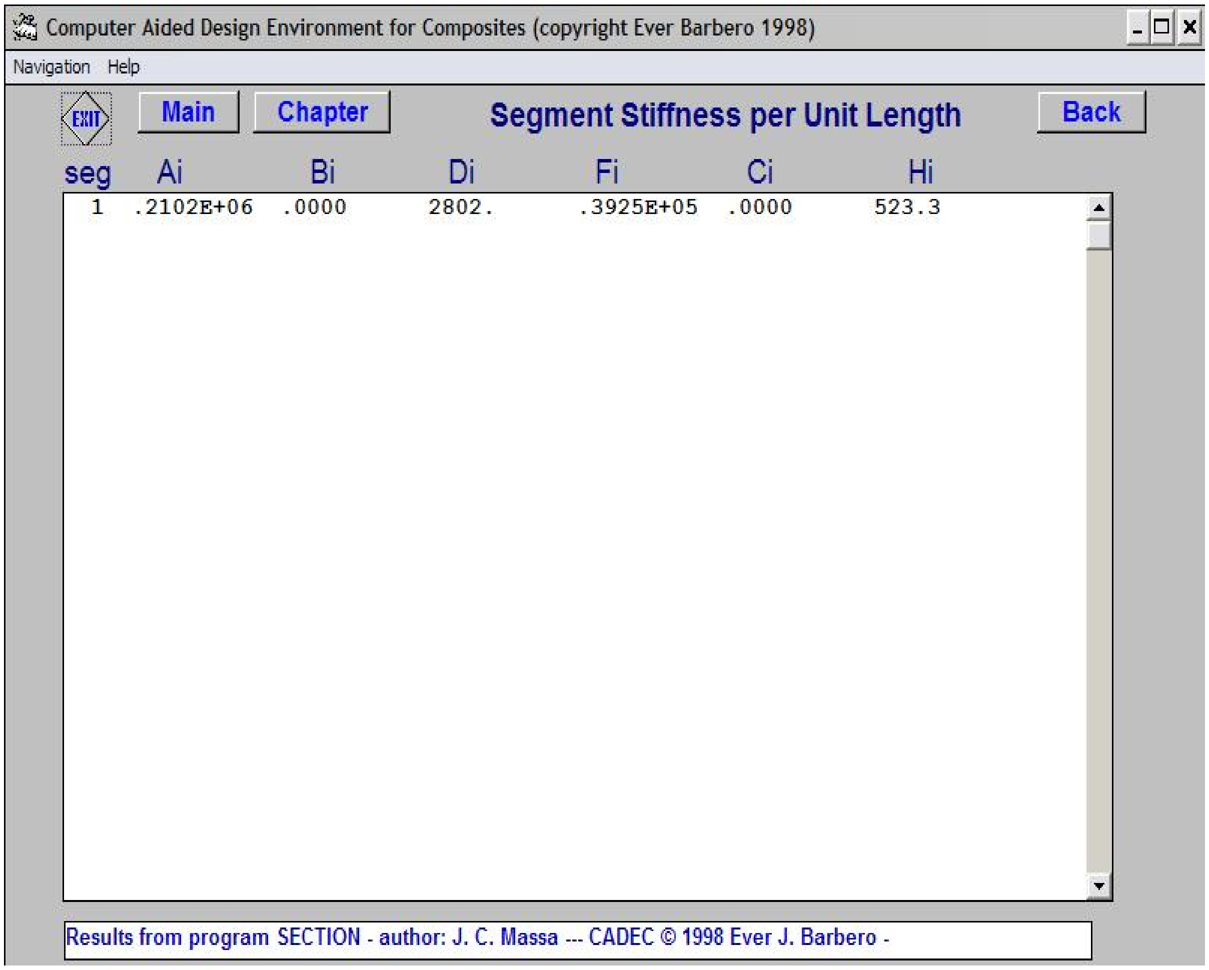
Segment	ni	nf	Laminate Type	Int	S1	S2	S3
1	1	2	c:\progra~1\cadec\ptype1.def	0	0	0	0
2	2	3	c:\progra~1\cadec\ptype1.def	0	1	0	0
3	3	4	c:\progra~1\cadec\ptype1.def	0	2	0	0
4	4	1	c:\progra~1\cadec\ptype1.def	0	3	0	0

Moment	Mx	My	Mz
	0	0	0

Force	Fx	Fy	Fz	x	y	z
0	0	0	1	100	0	0

Warping	ß	dß/dx	d2ß/dx2
	0	0	0

Figure 5.12 Beams GUI page with loaded values of "BOX.TWS".


Computer Aided Design Environment for Composites (copyright Ever Barbero 1998)
Navigation
Help
EXIT
Main
Chapter
Back

Segment Stiffness per Unit Length

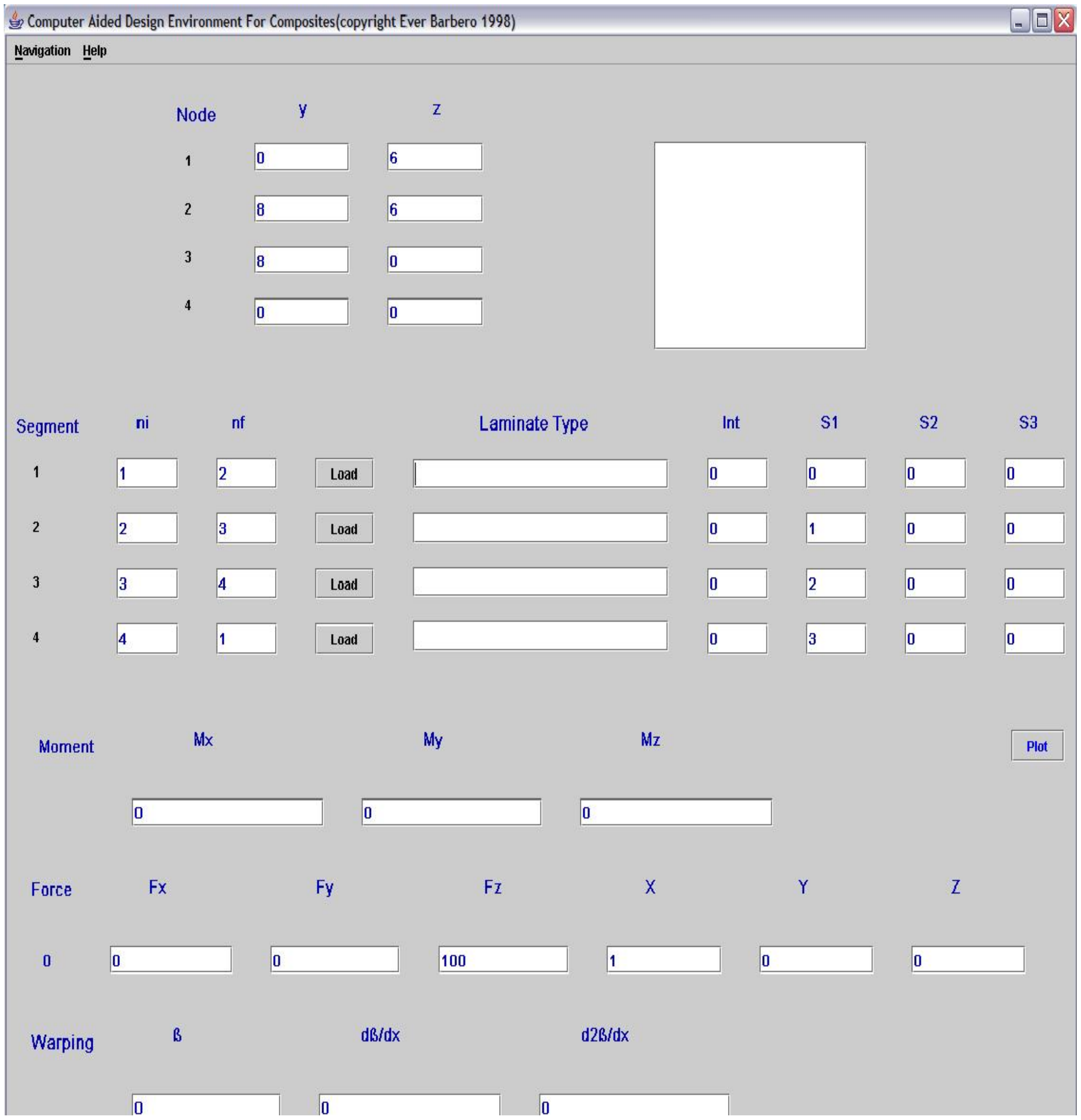
```
seg    Ai          Bi        Di        Fi          Ci        Hi
 1   .2102E+06   .0000     2802.     .3925E+05   .0000     523.3
```

Results from program SECTION - author: J. C. Massa --- CADEC © 1998 Ever J. Barbero -

Figure 5.13 Results page for Segment Stiffness per Unit Length.

5.5.2. Solution Using new CADEC software:

The problem is now solved by the software by entering the material same material properties. Figure 5.14 shows the GUI page with values entered for the beam and Figure 5.15 shows the result page for “Segment Stiffness per Unit Length”.


Computer Aided Design Environment For Composites(copyright Ever Barbero 1998)
Navigation Help
Load
Plot

Node	y	z
1	0	6
2	8	6
3	8	0
4	0	0

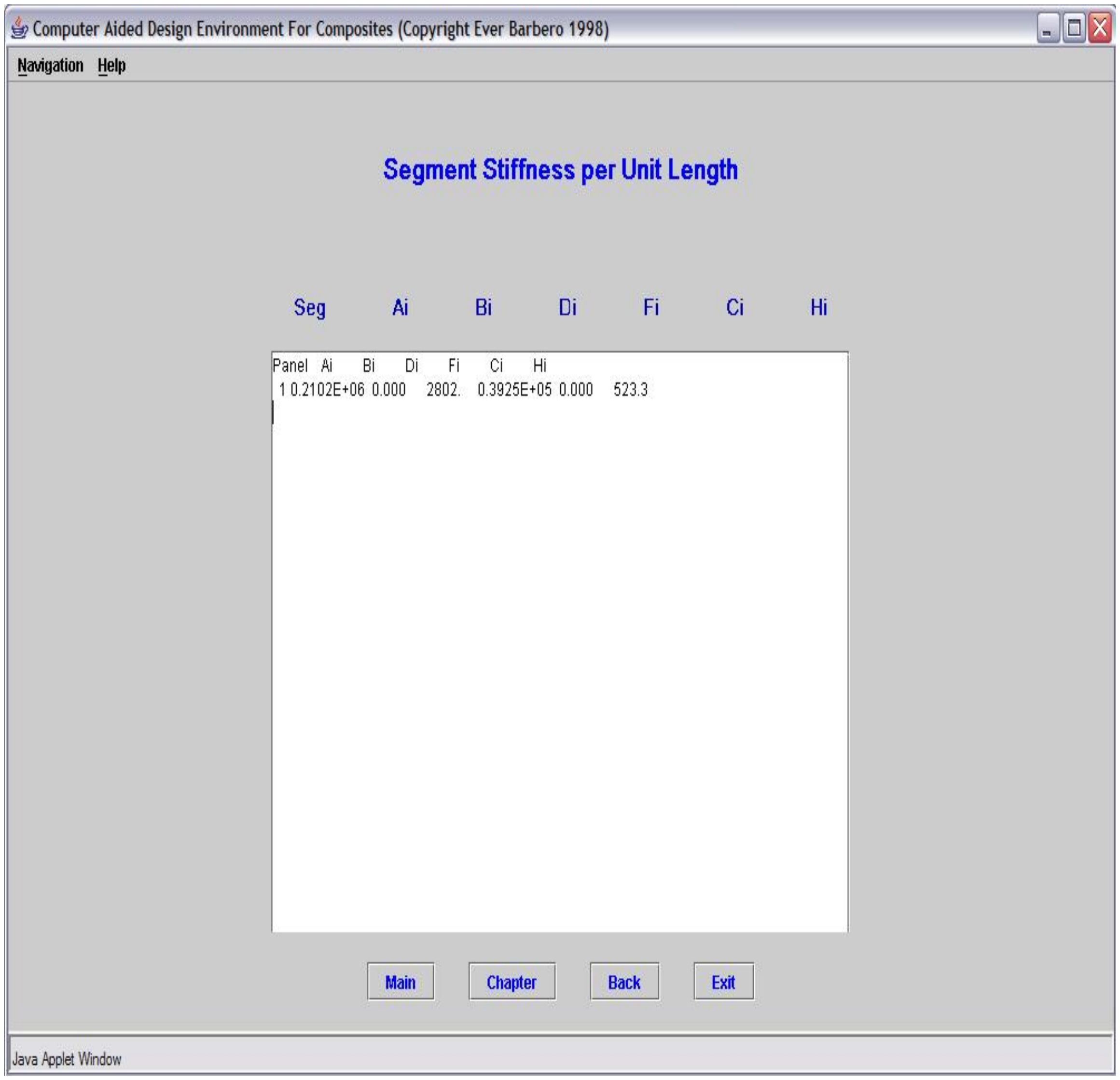
Segment	ni	nf		Laminate Type	Int	S1	S2	S3
1	1	2	Load		0	0	0	0
2	2	3	Load		0	1	0	0
3	3	4	Load		0	2	0	0
4	4	1	Load		0	3	0	0

Moment	Mx	My	Mz
	0	0	0

Force	Fx	Fy	Fz	X	Y	Z
0	0	0	100	1	0	0

Warping	ß	dß/dx	d2ß/dx
	0	0	0

Figure 5.14 Beam GUI page loaded with values from BOX.TWS


Computer Aided Design Environment For Composites (Copyright Ever Barbero 1998)
Navigation Help
Segment Stiffness per Unit Length
Seg Ai Bi Di Fi Ci Hi
Main
Chapter
Back
Exit
Java Applet Window

```
Panel  Ai       Bi     Di     Fi       Ci      Hi
 1 0.2102E+06  0.000   2802.  0.3925E+05  0.000   523.3
```

Figure 5.15 Results page for Segment Stiffness per Unit Length using new CADEC software.

Chapter 6. Conclusions and Future Work

6.1. Conclusions:

Example problems were being chosen from each chapter of the textbook and the problem is solved using the software. The results obtained from the software are a precise match with the results obtained using the textbook. Based on the result values graphs have also been successfully implemented for some material properties. The current version of the software proves that the CADEC software in VB can be ported to java very easily and efficiently, and thereby elimination the limitations posed by the CADEC software.

6.2. Future Work:

The scope of this computer software can be extended so as to enhance the features of this tool. Some of the features, which can be included, are as follows

- To include an Input/output dialog box enabling the user to save or load the saved data files thereby saving the precious time of reentering the data manually.

- Previously an application called converter was included in the old version of the software, which would aid the user to covert the material properties from one Unit system to another. Implementing such a feature in the current version of the software would enhance the scope of the tool.
- To provide the user with the option for printing the outputs and graphs

Bibliography

1. Hortan Ivor, “Beginning Java 2”, Wrox Press Limited, pp.15-30, 2000.

2. Milton, G.W. University of Utah, The Theory of Composites. Available [On-Line].
   http://assets.cambridge.org/0521781256/sample/0521781256WS.pdf

3. The Laminator (classical Analysis of composite Laminates). Available [On-Line]
   http://www.thelaminator.net/

4. Preliminary Design composite Materials and Structures(PDCMS) software Available [On-Line]
   http://www.eng.hawaii.edu/~nejhad/CompDesignSoft/composite.html

5. Automated System for Composite Analysis (ASCA) software Available [On-Line].
   http://composite.about.com/gi/dynamic/offsite.htm?site=http%3A%2F%2F www.adtechsystems.com%2Fasca.htm

6. Computer Aided Design Environment for Composite Materials.

   CADEC (On-Line) Available:

   http://www.mae.wvu.edu/~ejb

7. Barbero, E. J. (1999) “Introduction to Composite Materials Design”, Taylor & Francis, pp. 61-223.

8. A Java Matrix Package [JAMA] Available [On-line]

   http://math.nist.gov/javanumerics/jama/

9. Socket programming with TCP Available [On-line]

   http://www.seas.upenn.edu/~ross/book/apps/sockettcp.htm

10. All About Sockets Available [On-line]

    http://java.sun.com/docs/books/tutorial/networking/sockets/

11. Program : Available [On-line]

    http://www.webopedia.com/TERM/P/program.html

12. Assembly Language.

    http://networking.webopedia.com/TERM/A/assembly_language.html

13. Brewing Java : A Tutorial. Available [On-line]

http://www.ibiblio.org/javafaq/javatutorial.html

14. Object – Oriented Programming, Available [On-line]

http://www.webopedia.com/TERM/o/object_oriented_programming_OOP.htm
l

15. Object-Oriented Programming Concepts. Available [On-line]

http://java.sun.com/docs/books/tutorial/java/concepts/

16. Objected-Oriented Programming Concepts Available [On-line]

http://java.sun.com/docs/books/tutorial/java/concepts/class.html

17. Sestoft, P. (2002) "Java Precisely", The MIT press, pp. 26-28.

18. Lesson 2 Building Applications. Available [On-line].

http://www.developeriq.com/java/tutorials/chap2.php3

http://www.developeriq.com/java/tutorials/chap4.php3

19. Trail: Custom Networking, Available [On-line]

http://java.sun.com/docs/books/tutorial/networking/index.html

http://java.sun.com/docs/books/tutorial/networking/sockets/definition.htm

20. Socket Programming in Java : A tutorial. Available [On-line].

    http://www.javaworld.com/javaworld/jw-12-1996/jw-12-sockets.html

21. Socket Programming with TCP. Available [On-line].

    http://www.seas.upenn.edu/~ross/book/apps/sockettcp.htm