Object – oriented Steel Member Design Framework

C.G. Hewetson

13180738

Thesis presented in partial fulfilment of the requirements for the degree of Master of Civil Engineering at the University of Stellenbosch.

Study leader: Dr G.C. van Rooyen

December 2005

Declaration

I, the undersigned, hereby declare that the work contained in this thesis is my own original work and has not previously in its entirety or in part been submitted at any university for a degree.

Ek, die ondergetekende, verklaar hiermee dat die werk gedoen in hierdie tesis my eie oorspronklike werk is wat nog nie voorheen gedeeltlik of volledig by enige universiteit vir ‘n graad aangebied is nie.

Signature: Date: 24 November 2005

Synopsis

Adequate member design is a vital part of structural design. Current design software automates the design process by making use of the finite element model to create a design model. Although this is time effective, the engineer has limited control over the factors and procedures that are used for design. This leads to a lack of confidence in the eventual design results.

This thesis concentrates on developing a model for designing steel members with the emphasis on control over the model, its components and the design procedures. Methods for structural steel design are developed according to the new South African design code, namely *SANS 10162: Code of Practice for the Structural use of Steel: Part1: Limit States Design of hot – rolled steelwork – 2005*.

An object oriented framework for structural steel member design, including graphical user interface, is developed and implemented. The implemented framework:

- Implements the design paradigm of the new South African code for structural steel design.
- Builds on an existing architecture that allows for structural analysis, structural connection design and distributed collaboration in the design process

Opsomming

Voldoende ontwerp van struktuur onderdele is van die uiterste belang vir strukturele ontwerp. Huidige ontwerpsagteware vergemaklik die ontwerpprosedure deur gebruik te maak van die eindige element model om die ontwerp model te skep. Alhoewel dié proses tyd - effektief is, het die ingenieur min beheer oor die faktore en prosedures wat nodig is vir ontwerp. Dit lei tot ’n vermindering in vertroue in die finale ontwerpresultate.

Hierdie tesis fokus daarop om ‘n model te ontwikkel vir die ontwerp van staalstrukture met die klem op beheer oor die model, model samestelling en ontwerpprosedures. Metodes vir strukturele staalontwerp is ontwikkel volgens die nuwe Suid Afrikaanse ontwerpkode, naamlik *SANS 10162: Code of Practice for the Structural use of Steel: Part1: Limit States Design of hot – rolled steelwork – 2005*.

’n Objek-orienteerde raamwerk en ’n grafiese gebruikersoppervlak is ontwikkel en geimplimenteer vir strukturele staalontwerp. Die geimplimenteerde raamwerk:

- Gebruik die nuwe Suid Afrikaanse ontwerpkode vir strukturele staal, as ontwerp basis.
- Bou op ‘n bestaande argitektuur wat stukturele analise, strukturele verbindingsontwerp en verspreide samewerking in die ontwerpproses toelaat.

Table of Contents

List of Figures

List of Tables

Glossary

ϕ	= resistance factor for structural steel
A	= the gross area of the profile cross section
C_e	= the Euler buckling strength
C_r	= factored compressive resistance of a member
C_u	= the ultimate compressive force in a member; ultimate axial load
C_y	= the axial compressive force in a member at yield stress
E	= the elastic section modulus of steel
f_u	= the specified ultimate tensile stress
f_y	= the yield stress
h	= the height of a steel section
h_w	= the clear depth of web between flanges
I	= the moment of inertia (subscripts refer to axes)
J	= St. Venant torsion constant of a cross section
K	= effective length factor (subscripts refer to axes)
L_{eff}	= effective length
L	= gross length of a member
M_{cr}	= the critical elastic moment of a laterally unbraced beam
M_p	= the plastic moment
M_r	= the factored moment resistance of a member
M_u	= the ultimate bending moment in a member
M_y	= the yield moment
r	= radius of gyration
T_r	= the factored tensile resistance of a member
T_u	= the ultimate tensile force
t_f	= the flange thickenss
t_w	= the web thickness
U_1	= the factor to account for moment gradient and for second order effects of an axial force acting on the deformed member
W	= width to thickness ration of a cross section
Z_e	= the elastic section modulus of a steel section
Z_{pl}	= the plastic section modulus of a steel section

Acknowledgements

I am thankful to Dr. G.C. van Rooyen for his guidance, support and teachings throughout my years of study. It was a privilege and experience working with him.

Many thanks to Prof P.E. Dunaiski for his guidance with the specification of structural steel member design. I am grateful to have worked with him.

Many thanks to Bertie Olivier for the time spent in assisting with the integration with the finite element model. I am grateful for his patience and tireless efforts.

Thanks go to Eike Tauscher who assisted with the 3D modelling of members.

1 Introduction

General: Adequate member design and control over structural member properties is an important part of the structural design task. Member design and proportioning of elements follows on structural analysis. For the purposes of analysis a finite element model is created which provides member forces and the translations and rotations of the model’s degrees of freedom. However, the finite element model does not provide sufficient information for the design of structural members. Design of structural members is a complex procedure and should be performed with all the necessary information available, as required by the various design procedures.

1.1 Structural Design

General: Structural design is an iterative process of applying engineering mechanics and knowledge of the surrounding environment to create a functional and safe structure. By making use of structural analysis techniques and relating the results to specific design code criteria, a solution is obtained about the adequacy of a proposed structural makeup. Structural analysis techniques are employed to compute the forces, stresses, rotations and displacements of a structure. The analysis provides solutions to the behaviour of a structure under certain load conditions. The behaviour of the structure, due to the forces and stresses at work within a structure, is then related to appropriate design specifications. In this process the structural members and their connections are tested for their conformity with strength and serviceability standards as set by the design code.

Requirements: For adequate structural member design, the design paradigms require certain knowledge about the individual members that comprise a structure along with the structure itself. This information is used in the design procedures stipulated in the various design codes to test for structural member strength and serviceability. Typical information required during the design process is the following:

- The physical length of the structural members. This physical length depends on the geometry and topology of the structure under design.
- Complete knowledge of the external loading, the internal forces and internal stresses occurring along the length of a structural member.
- Complete knowledge of the translation and rotation of the structural members during loading. This data is used to test for serviceability adequacy of a structural member.

- The type of material of the structural member. This determines the type of design code and thus the design procedures used for testing.
- The cross sectional geometry of a structural member.
- The end conditions of a structural member. Depending on the end conditions of a structural member, whether they are connected to other members or have free ends, etc, determine effective lengths of the structural member itself, influencing its stability and strength.
- Knowledge of the stability and behaviour of the structure as a whole.

1.2 Computational Structural Design

General requirements: Design of structural members is an intricate process and efficient design is beneficial for effective engineering practise. By applying the processing power of computers in a computational design framework, the lengthy procedures involved in structural analysis and design can be reduced.

In order to model the steps in the overall design process of a structure, separate models dealing with the finite element analysis, connection design and the member design should be developed. The remainder of this chapter will develop the reasoning behind a separate structural member design model.

A finite element model's primary function is to analyse a structure subject to a given load condition. This model thus provides the stresses and strains at work within a structure. Each member in the structure is modelled with numerous finite elements. This model, and thus the finite elements, does not provide sufficient control and information as is required by the various design codes and processes involved with structural member design. To solve this problem, a separate member design model should be implemented. This model should represent a structure and its members from a design perspective, thus containing all the required design information as stipulated by the appropriate design codes. Although this design model is separate, all data pertaining to the finite element model shall be linked to the model.

The components of the design model are special two dimensional elements. These elements will be specialized in their functionality, which is structural member design. These special elements, or design elements, should not be dependent on the topology of the finite elements, meaning that the length and end points of the design elements need not coincide with that of the finite elements. This would result in greater control over the design model. The overall geometry and shape of the structure should remain the same as is stipulated by the finite element model.

Depending on the material type of the structural members, e.g. steel, concrete, timber etc, the appropriate design procedures can be implemented in the design model. The design elements should then provide the additional design information as is required with different material types.

Structural Steel Design Framework: For the purpose of this thesis a design model adapted for the design of structural steel members is developed. The general structural member design model shall be developed as mentioned earlier. This will allow for easy adaptation for design with any material type. This model shall then be adapted for the design of hot rolled structural steel members. The South African code *SANS 10162: The Structural use of Steel: Part 1: Limit States Design of hot – rolled steelwork – 2005* represents procedures used for the design of hot – rolled steel members. These design procedures shall be analysed with the aim of creating an object orientated computational framework and specialized design model for hot – rolled steel members.

Structural Steel Design requirements: The additional requirements for a structural member design framework aimed specifically at hot – rolled steel are as follows:

- The computational design framework implements the design paradigm as employed by the South African code for hot rolled structural steel.
- All the design types of steel members will be implemented in the framework and associated with their specific design procedures as stipulated by the South African code.
- Each design procedure implemented by the design framework shall easily be modified to allow for additional design procedures or changes in design type.
- An extensive collection of hot – rolled steel profiles must be made available for use in the design framework without fixing any specific parameters. Various grades of steel must be made available to represent the different steel profiles. This requirement shall be met by creating a database of steel profile properties, cross sectional properties and material properties. These databases can easily be modified to include or exclude specific steel profiles and or material types.
- The structural steel member model must be built on an existing architecture that allows for structural finite element analysis, structural connection design and distributed collaboration in the design process. The existing structure supports heterogeneous multimodels in an application by allowing for finite element analysis models, connection design models, member design models and the seamless transfer of information between models. The

member design model can for example obtain forces from the finite element analysis model and end conditions from the connection design model.

Overview: The thesis begins with a brief description of the existing architecture in chapter 2. The important concepts and relevance of the classes used are discussed.
The sign convention and axis system used to represent the members in space is described in chapter 3. This includes the global axis system, local axis system of the finite element and member design model as well as internal forces sign convention.
The detailed design specification for the steel members is described in chapter 4. The implemented specification is based on the new South African Code released in 2005, namely *SANS 10162: The Structural Use of Steel: Part 1: Limit States Design of hot – rolled steelwork – 2005*.
The various types of hot – rolled steel profiles and their adaptation to the design framework are described in chapter 5.
The properties of specialized elements used for design, namely Design Elements, are discussed in chapter 6 along with specialized Design Elements used for structural steel design, namely SSDesign Elements.
Chapter 7 describes the concepts and advantages of dividing a structure into "representative sets" or "Design Sets".
The development and implementation of the steel member computational design framework is then discussed and illustrated in chapter 8. The thesis concludes with illustrated examples from the computational framework and verification with hand calculations of the design for each example, in chapter 9.

2 Brief Background on Existing Architecture

Existing architecture: The creation of a software architecture for the integration of the various analysis and design models is currently in progress and is the subject of another study. The aim of the programming architecture is to provide an object oriented application environment that allows multiple models and model types to exist inside an application. The model types include e.g. finite element models, steel member design models and steel connection design models. The seamless integration of these models into a complete framework is the focus of that study. A structural steel member design model suitable for inclusion into the currently researched architecture is developed in this thesis.

Object oriented programming: The fundamental difference between using an object oriented framework and procedural approach is that data has meaning outside the scope of a specific algorithm. This enables the sharing of data at object level which enhances collaboration possibilities and transfer of information between heterogeneous models.
An object is an instance of a class and is described by its attributes and methods. The methods of an object are normally referred to as its functionality. A class therefore serves as a mould for objects of the class.

2.1 Basic Structure

The software architecture used at the time of this thesis is described in the thesis *Object – Oriented Steel Connection Design FrameWork by G.E. Willemse, section 2.* The software architecture is called Juma. This is illustrated in Figure 2-1. The structure is divided into the following basic folders which are also known as packages. Each model contributing to the collaboration of analysis and design has its own package in the structure.

- classes

  This package contains all the compiled files which are used by the computer to run the application.
- doc

  This package contains the java documentation of the application
- component

  This package contains the components of each model type separated in their own sub packages, e.g. *fe* for finite element components and *ssMem* for member design components. This package contains the class `AppObject` which is used by all components.

- interface

This package is divided into the same sub packages as the components package. Each sub package is contains the interfaces of the different model types, e.g. finite element model and member design model. This package also contains two interfaces, namely `IAppObject` and `IModel`. This `IModel` interface is used by all model objects while the `IAppObject` interface is used by all the components.

- model

This package contains the model of each model type. This package is divided up into each model's sub package.

- service

This package contains all the analysis and design classes associated with each model type. Class Application forms the basis and allows for multiple model types.

- gui

This package contains all the graphical user interface components of each model type.

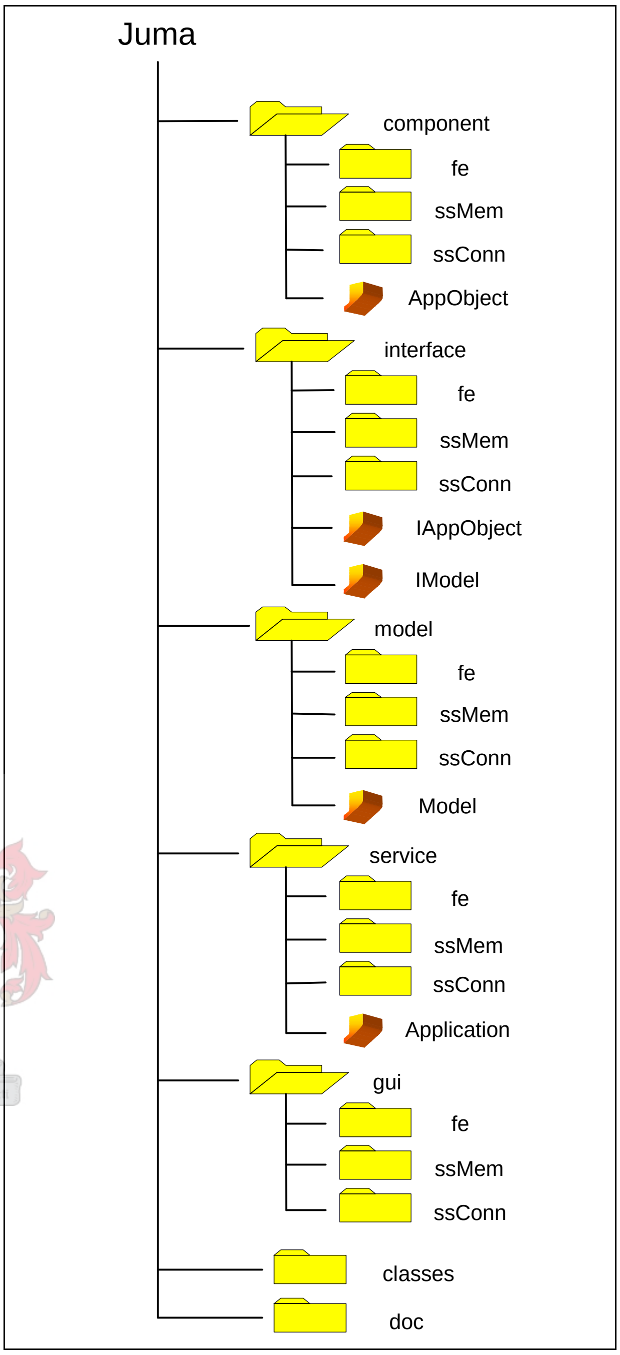


Figure 2-1 The Juma structure

The remainder of this chapter will briefly discuss the relevance of the important classes used in the structure of Juma. A more detailed description is provided in the Java Documentation of the application.

2.2 Important classes

The important classes and interfaces used for the basic structure of the design framework are discussed:

2.2.1 `IAppObject`

This interface prescribes the basic functionality of all application objects (`AppObjects`) whether they are finite element components, steel member components or steel connection components. This interface ensures all `AppObjects` have the ability to return the name of the `AppObject` when requested. The interface further enforces that all `AppObjects` have the ability to activate the references of associated `AppObjects` via their identifiers, i.e. there persistent identifiers. The relevance of interfaces is discussed in chapter 8.

2.2.2 `AppObject`

This class is the basic class in applications structured around persistently identified objects. For this purpose it implements interface `IAppObject` and therefore contains all the methods prescribed by this interface. All objects that represent components such as finite element components and steel member design components are persistently identified and are able to activate the references of their associated `AppObjects` via their names. This class further allows for the auto naming of components.

2.2.3 `IModel`

The `IModel` interface describes the functionality of `Model` objects. A `Model` object encapsulates all the components used in the framework and can be seen as a set of component objects. This interface provides the functionality for components to be added and removed from the model. Components can be obtained from the model object via their names or references.

2.2.4 `Model`

An object of class `Model` contains a single set that contains the components of a specific model type, for example a structural steel member design model. The components may be of any class and implement any interface. Typical components of a structural steel member model are steel members, steel profiles, internal restraints, restraints and internal elements.
The single set that an object of class `Model` contains is a special set. This set is filterable. In other words any object of any class with any attribute conditions specific to that object can be navigated to and obtained from the set by applying an appropriate filter procedure.

This filterable set contains all the components relevant to the application, in this case all structural steel member components. Each component within this set can be obtained by applying the appropriate filter procedure to the component set.

2.2.5 `Application`

Class `Application` provides the functionality for the existence of multiple models in the application by maintaining references to all the existing models. This class allows for control over the various models by allowing for the addition of models, the removal of models and the control over which model is currently active

3 Sign Convention

The definition of the global and local axis convention used in the framework as well as the sign convention adopted for the loading on the elements is developed in this chapter.

3.1 Axis Systems

The global axis is defined according to the right hand coordinate system and is shown in Figure 3-1. This system is used in the Design Model as well as in the underlying Finite Element model.

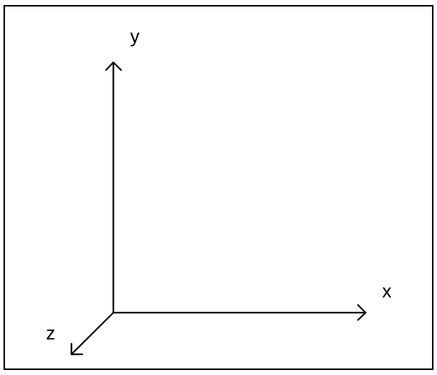


Figure 3-1 Axis definition of the global axis vectors

The global directional vectors are as follows:

- X direction: [1; 0; 0]
- Y direction: [0; 1; 0]
- Z direction: [0; 0; 1]

The local axis system of individual elements is defined in accordance with the right hand coordinate system with the local x-axis running along the length of the element. All local axes of design elements are described with reference to these global direction vectors.

The orientation of an element's local axis in comparison to the global axis is shown in Figure 3-2, with the local y = y' and local x = x'axes as indicated.

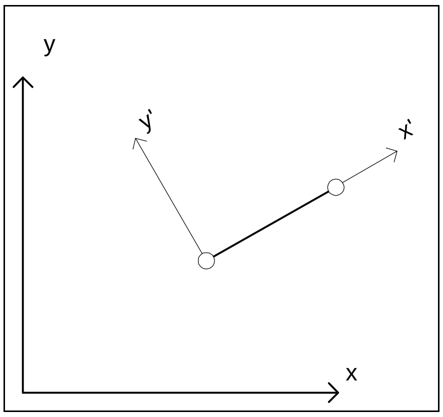


Figure 3-2 Local axis of an element

3.2 External- and Internal Force Sign Convention

Fixed end forces: The sign convention of the internal and external forces of an element is developed as follows.

The (external) end forces of an element are defined as positive if they are applied in the direction of the positive local axes of the element. This is described as follows:

- All fixed end shear forces are taken as positive if they are in the positive direction of the local y and z – axes of the element.
- All axial forces are taken as positive if they are in the positive direction of the local x – axis of the element.

- All bending and torsional moments are taken as positive if the moment vector corresponds with the direction of the local x, z and y – axes according to the right hand rule.

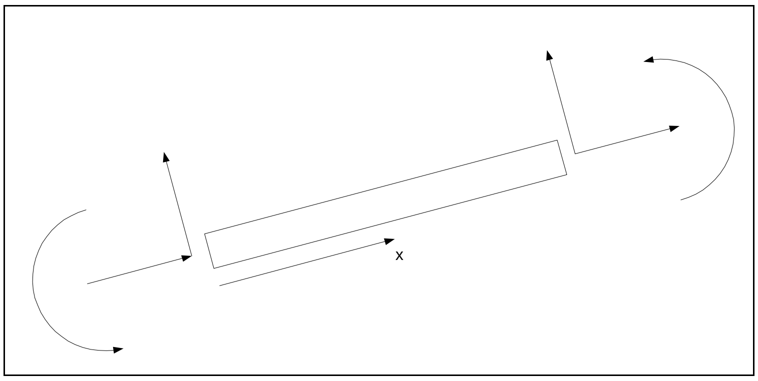


Figure 3-3 Positive end forces of an element

Figure 3-3 shows an element in space with positive end forces at its ends with its local x – axis as indicated.

Internal forces: The chosen convention for internal forces is shown in Figure 3-4. M indicates bending moment about the elements local z – axis, V indicates shear force in the elements local y – axis direction and N indicates an axial force in the local x – axis direction of the element. The directions indicated were chosen as positive. This convention leads to positive stresses on positive parts of a cross – section with a positive normal vector. The internal forces are calculated from the end forces in combination with the internal loading, if any, of the member. The method used is described below with the help of Figure 3-5

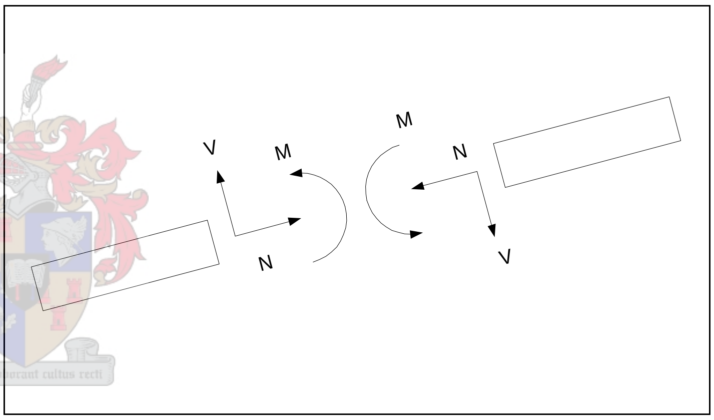


Figure 3-4 Positive end forces

Sign convention: As can be seen from Figure 3-5, the contribution of the fixed end forces at A to the internal forces at C result in a negative axial force (N), a negative shear force (V) and a positive bending moment (M). The contribution of the fixed end forces at B cause a positive axial force (N), a positive shear force (V) and a negative bending moment (M). This procedure is maintained for all elements in the finite element model.

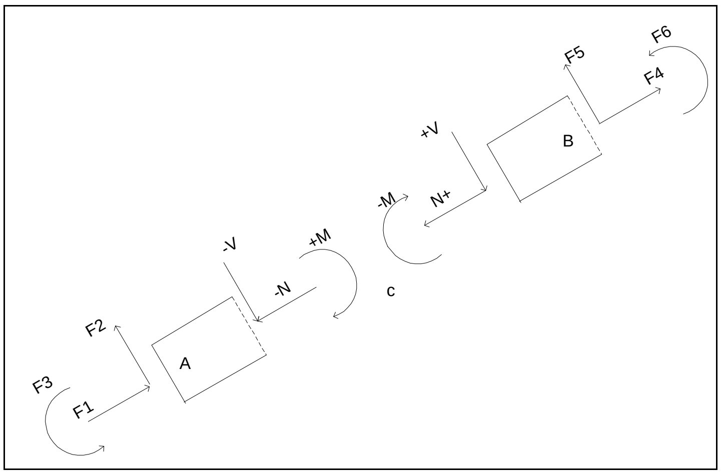


Figure 3-5 Internal forces resulting from end forces

The internal forces are then calculated within the finite element model by relating the fixed end forces to the internal forces as follows:

- $\begin{bmatrix} N \\ V \\ M \end{bmatrix} = \begin{bmatrix} -F_1 \\ -F_2 \\ +F_3 \end{bmatrix}$
- $\begin{bmatrix} N \\ V \\ M \end{bmatrix} = \begin{bmatrix} +F_4 \\ +F_5 \\ -F_6 \end{bmatrix}$

Once this is done, the contribution of any element loading is applied to the internal forces. This approach can be applied to three dimensional loading.

This sign convention is maintained within the design model.

4 Member Design Specification

The specification for designing steel members is developed according to the new South African code *SANS 10162 2005*: *Code of Practice for the Structural use of Steel Part1: Limit States Design of Hot-Rolled steelwork.* All the steel member design types were included.

4.1 Pure Flexural Members

The following types of bending were considered:

- Uni-axial strong axis bending.
- Uni-axial weak axis bending
- Bi-axial bending.

The requirements for design of such members are given in the following clauses of *SANS 10162: Part1:*

- 10 Design lengths and slenderness ratios
- 11 Width-thickness ratios
- 13.5 Bending: Laterally supported members
- 13.6 Bending: Laterally unsupported members

Doubly symmetric sections of class 1, 2 and 3 as well as Channel profiles were implemented in the design framework of this thesis.

The design procedure is implemented through the following steps:

4.1.1 Determining the effective lengths of flexural members

4.1.1.1 Simply supported beams

Table 4-1 is an excerpt of effective length factors taken from *SANS 10162 Part1: 10.2.1. Table 1.*

For beams supported at both ends where no lateral restraint of the compression flange along the beam is provided but where each end of the beam is retrained against torsion, the effective length factor K to be used shall be given in Table 1. [SANS 10162 Part1: 10.2.1]

The effective length factors indicated in this table determine the lateral torsional buckling length of the member. Members that are continuously braced along its length do not require an effective length factor, as bending resistance is calculated on cross sectional resistance.

Table 4-1 Effective length factors for simply supported beams

1	2	3
Restraint against lateral bending at supports	**Effective length factor K**	
	Loading condition	
	Normal	**Destabilizing**
Unrestrained (i.e. free to rotate in plane)	1.0	1.2
Partially restrained (i.e. positive connection by flange cleats or end plates)	0.85	1.0
Practically fixed (i.e. not free to rotate in plan)	0.7	0.85

Where beams have no restraint against torsion, the values of the effective length factor K in Table 4-1 shall be increased by 20%.

The destabilizing loading condition applies when the load is applied to the compression flange of the beam and both the load and the flange are free to move laterally.

For beams that are provided with members giving effective lateral restraint to the compression flange at intervals along the span, in addition to the torsional restraint as required above, the effective length shall be taken as the distance, centre to centre, between the restraint members.

<u>Implementation aspects:</u>

For effective design purposes, software should support such functionality to have control over the definition and assignment of these effective length factors. Software should further allow for the functionality to provide additional lateral support to a member, without having to redefine the member's layout as spanning between the lateral supports.

The prototype application allows for the user to assign these effective length factors to the steel members by manipulating the restraint conditions at the ends of the member, resulting in the calculation the effective length used for lateral torsional buckling resistance of the member.

The application provides further functionality for the user to stipulate any additional support along the length of the member, thus reducing effective lateral torsional buckling lengths.

For the purpose of applying different restraint conditions at each end of a member, a conservative approach is taken by using the largest effective length factor provided in Table 4-1.

4.1.1.2 Cantilever Beams

The effective length factor K to be used in design is given in *SANS 10162 Part 1: 10.2.2 Table 2*. Table 4-2 shows an excerpt of Table 2.

Table 4-2 Effective length factor for cantilever beams

1	2	3	4
Restraint Condition		**Effective length factor K**	
		Loading Condition	
At Support	**At tip**	**Normal**	**Destabilizing**
Built in laterally and torsionally	Free	0.8	1.4
	Lateral restraint only (at compression flange)	0.7	1.4
	Torsional restraint only	0.6	0.6
	Lateral and torsional restraint	0.5	0.5
Continuous with lateral and torsional restraint	Free	1.0	2.5
	Lateral restraint only (at compression flange)	0.9	2.5
	Torsional restraint only	0.8	1.5
	Lateral and torsional restraint	0.7	1.2
Continuous with lateral restraint only	Free	3.0	7.5
	Lateral restraint only (at compression flange)	2.7	7.5
	Torsional restraint only	2.4	4.5
	Lateral and torsional restraint	2.1	3.6

For the case of a cantilever beam, the destabilizing loading condition applies when the load is applied to the tension flange of the beam and both the load and the flange are free to move laterally. This effective length factor is used, as in the case of simply supported beams, to determine the effective length of the member for the calculation of the member's resistance against lateral torsional buckling. The effective length factor only has an effect on laterally unbraced members.

4.1.2 Determining the factored moment of resistance

4.1.2.1 Bending-Laterally supported members

All members that have continuous lateral support along its entire length fall under this section. Members that are unbraced but are subjected to weak axis bending alone are considered here too. Design is then based on cross sectional strength of the section under consideration.

For sections of class 1 and 2, the factored moment of resistance is calculated as follows:

$M_r = \Phi \cdot Z_{pl} \cdot f_y$ [SANS 10162 Part1: 13.5(a)]

where

M_r = the factored moment resistance

Φ = 0.90

Z_{pl} = plastic section modulus of steel section

f_y = minimum yield stress of steel section

For section of class 3, the factored moment of resistance is calculated as follows:

$M_r = \Phi \cdot Z_e \cdot f_y$ [SANS 10162:Part1: 13.5(b)]

where

M_r = the factored moment resistance

Φ = 0.90

Z_e = elastic section modulus of steel section

f_y = minimum yield stress of steel section

4.1.2.2 Bending: Laterally unsupported members

The factored moment of resistance of members that do not have continuous lateral support of their compression flange is calculated as follows:

For doubly symmetric sections of classes 1 and 2:

- When $M_{cr} \geq 0.67 M_p$

$M_r = 1.15 \cdot \Phi \cdot M_p \left(1 - 0.28 \dfrac{M_p}{M_{cr}}\right)$, but not exceeding $\Phi \cdot M_p$

- When $M_{cr} \leq 0.67 \cdot M_p$

$M_r = \Phi \cdot M_{cr}$

with

$$M_p = Z_{pl} \cdot f_y$$

$$M_{cr} = \frac{\omega_2 \cdot \pi}{K \cdot L} \sqrt{E \cdot I_y \cdot G \cdot J + \left(\frac{\pi \cdot E}{K \cdot L} \right)^2 I_y \cdot C_w}$$

For doubly symmetric sections of class 3 or Channel profiles:

- When $M_{cr} \geq 0.67 M_y$

$M_r = 1.15 \Phi M_p \left(1 - 0.28 \frac{M_y}{M_{cr}} \right)$, but not exceeding $\Phi \cdot M_y$

- When $M_{cr} \leq 0.67 M_y$

$$M_r = \Phi \cdot M_{cr}$$

with

$$M_p = Z_{pl} \cdot f_y$$

$$M_{cr} = \frac{\omega_2 \cdot \pi}{K \cdot L} \sqrt{E \cdot I_y \cdot G \cdot J + \left(\frac{\pi \cdot E}{K \cdot L} \right)^2 I_y \cdot C_w}$$

where

M_r = the factored moment resistance

M_{p} = the plastic moment

M_y = the yield moment

M_{cr} = the critical elastic moment of an unbraced member.

$\Phi = 0.9$

κ = the ratio of the smaller to the larger ultimate end moment at opposite ends, positive for double curvature and negative for single curvature.

I_y = the second moment of inertia of the section about its weak axis.

E = elastic modulus of steel

G = shear modulus of steel.

J = St. Venant torsion constant.

C_w = warping torsion constant, equal to 0 for structural hollow sections.

K = effective length factor.

L = unbraced length of member.

ω_2 = coefficient to account for increased moment resistance of a laterally unsupported beam segment when subjected to a moment gradient.

4.1.2.3 Determining the value of ω_2

The value of ω_2 is determined by *SANS 10162 Part1: 13.6 (a).* $\omega_2 = 1.75 + 1.05 \cdot \kappa + 0.3 \cdot \kappa^2 \leq 2.5$

The value of ω_2 is equal to 1.0 when the bending moment at any point within the unbraced length of the compression flange is larger than the larger end moment or when there is no effective lateral support for the compression flange at one of the ends of the unsupported length.

4.1.2.4 Determining the value of κ

The value of κ is equal to the ratio $\frac{M_1}{M_2}$, with M_1 being the smaller end moment and M_2 being the larger end moment. This results in a negative value for single curvature and a positive value for double curvature. If either of the end moments is equal to zero, the value of κ is taken as 0.

4.1.2.5 Bi-Axial Bending

In addition, for bi-axial bending, the member shall meet the following criteria:

$$\frac{M_{ux}}{M_{rx}} + \frac{M_{uy}}{M_{ry}} \leq 1.0 \qquad \text{[SANS 10162 Part1: 13.6(e)]}$$

where

M_{ux} = the ultimate moment about the x axis

M_{rx} = the factored moment of resistance about the x axis.

M_{uy} = the ultimate moment about the y axis

M_{ry} = the factored moment of resistance about the y axis.

4.1.2.6 Design procedure

The value of the factored moment of resistance, M_r, of a steel member as calculated in either 4.1.2.1 or 4.1.2.2 is then compared to the value of the maximum factored moment (M_u) occurring along the length of the member. If M_u is smaller than the value calculated for M_r, the profile is deemed adequate for the design. The procedure in 4.1.2.5 follows the same pattern.

4.2 Columns

Members subjected only to compressive forces are considered in this section.
The requirements for design of such members are given in the following clauses of *SANS 10162: Part1:*

- Annex E Effective lengths of columns.
- 10 Slenderness ratios

- 11 Width-thickness ratios – Elements in compression
- 13.3 Axial Compression.

4.2.1 Maximum Slenderness Ratios

According to *SANS 10162 Part1: 10.4.2.1,* the maximum slenderness ratio of a member in compression shall not exceed 200.
The slenderness ratio is calculated as follows:

$$\frac{KL}{r} \leq 200$$ [SANS 10162 Part1: 10.4]

where,

K = the effective length factor of the member.

L = the actual length of the member.

r = the appropriate radius of gyration for the particular axis under consideration.

4.2.2 Effective length factors

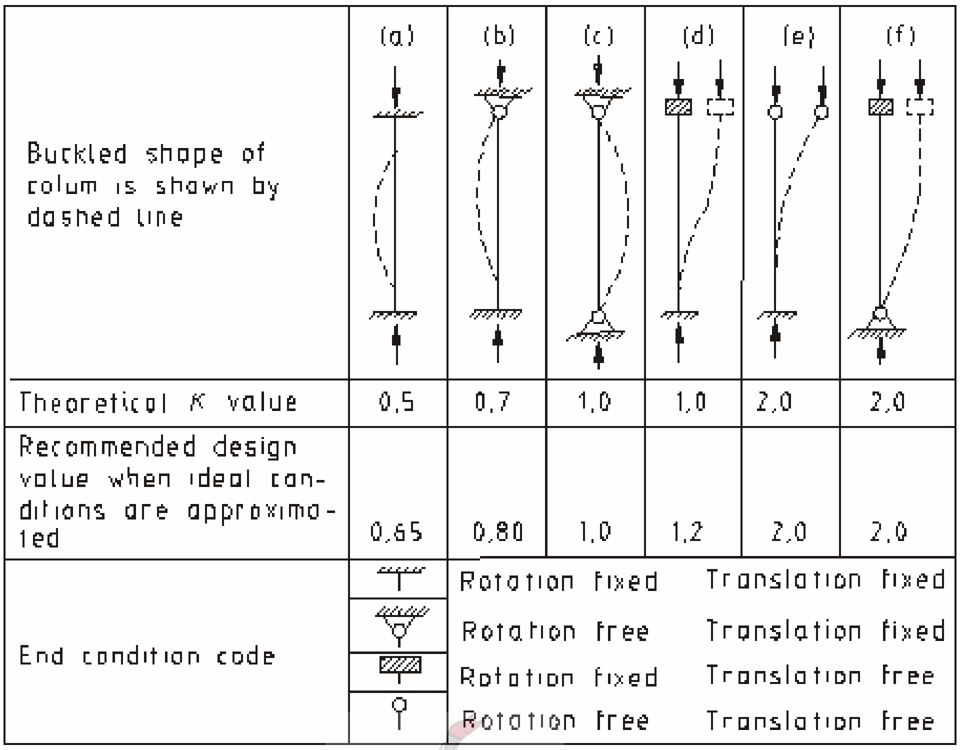
Table 4-3 is an excerpt of effective length factors taken from SANS *10162 Part1: 2005: Annex E. Figure E1 Idealized cases.* For columns with effective points of lateral bracing along its length, the effective length is taken as the distance between the centre points of these bracings.

Within the scope of this thesis, Table 4-3 was used to calculate the effective length of columns to determine their resistance to out of plane (weak axis) Euler buckling.
For in plane buckling (strong axis buckling) the effective length factor is taken as 1.0.

Implementation aspects:

As stated in section 4.1.1.1, it is advantageous for effective design that design software allow for the functionality to have control over these effective length factors.
In the prototype design application, the user has the option to toggle the restraint conditions in Table 4-3 to apply the effective length factors to the steel member.
Furthermore, the user has the option to allow for any additional lateral support along the length of the member, if applicable, to reduce the Euler buckling length of a particular column.
The results taken from the finite element model are assumed to be of a 2 dimensional second order analysis. This results in an effective length factor for in plane buckling to be taken as 1.0. This is stipulated in *SANS 10162 Part1: 10.3.2*.

Table 4-3 Idealized effective length factors



	(a)	(b)	(c)	(d)	(e)	(f)
Buckled shape of colum is shown by dashed line						
Theoretical K value	0,5	0,7	1,0	1,0	2,0	2,0
Recommended design value when ideal conditions are approximated	0,65	0,80	1,0	1,2	2,0	2,0

End condition code			
	⊥	Rotation fixed	Translation fixed
	▽	Rotation free	Translation fixed
	▨	Rotation fixed	Translation free
	○	Rotation free	Translation free

Profiles of class 1, 2 and 3 were considered in this section being either bisymmetric, asymmetric or monosymmetric. The design procedure followed was to test all members for flexural buckling and torsional flexural buckling resistance and thus select the lowest compressive resistance. The maximum slenderness ratio of the member was used in this procedure. The design procedure is as follows:

4.2.3 Compressive resistance for flexural buckling mode

All bi – symmetric sections of class 1, 2 or 3 are considered in this section. The compressive resistance, based on Euler buckling, is calculated as follows:

$$C_r = \Phi \cdot A \cdot f_y \cdot \left(1+\lambda^{2n}\right)^{\frac{-1}{n}}$$ [SANS 10162 Part1: 13.3.1]

where

C_r = the factored compressive resistance

A = the cross sectional area

f_y = the yield stress of steel

$\Phi = 0.9$

n = 1.34 for hot-rolled, fabricated structural sections, and hollow structural sections manufactured according to SANS 657-1 (cold-formed non-stress-relieved) n = 2.24 for doubly symmetric welded three – plate members with flange edges oxy flame cut and hollow structural sections manufactured according to ISO 657-14 (hot-formed or cold-formed stress relieved)

$$\lambda = \frac{K \cdot L_x}{r_x}\sqrt{\frac{f_y}{\pi^2 \cdot E}} = \sqrt{\frac{f_y}{f_e}} \qquad \text{[SANS 10162:Part1: 13.3.1]}$$

where,

K = the effective length factor

L = the unbraced length of the member (L_x, L_y)

r = the appropriate radius of gyration for the buckling axis

f_y = the yield stress of steel

E = Young's modulus for steel

4.2.4 Compressive resistance for torsional flexural buckling mode

Asymmetric and monosymmetric profiles of class 1, 2 and 3 that were not covered under 4.2.3 are considered in this section. The compressive resistance is calculated as follows:

4.2.4.1 Singly symmetric sections

The compressive resistance of all singly symmetric profiles is taken as the lesser value of f_{ex} and f_{eyz}.

In this case the y-axis refers to the axis of symmetry of the profile.

$$f_{ey} = \frac{{r_y}^2 \cdot \pi^2 \cdot E}{K^2 \cdot {L_y}^2} \qquad \text{[SANS 10162 Part1: 13.3.2b)]}$$

$$f_{ex} = \frac{{r_x}^2 \cdot \pi^2 \cdot E}{K^2 \cdot {L_x}^2}$$

$$f_{eyz} = \frac{f_{ey} + f_{ez}}{2 \cdot \Omega}\left(1 - \sqrt{1 - \frac{4 \cdot f_{ey} \cdot f_{ez} \cdot \Omega}{\left(f_{ey} + f_{ez}\right)^2}}\right) \text{ with}$$

$$\Omega = 1 - \left(\frac{{x_o}^2 + {y_o}^2}{\bar{r}_o^2}\right) \text{ and}$$

$$\bar{r}_o^2 = {r_x}^2 + {r_y}^3 + {x_o}^2 + {y_o}^2$$

$$f_e = \min\left\{f_{ex}, f_{eyz}\right\}$$

$$\therefore \lambda = \sqrt{\frac{f_y}{f_e}}$$

$$C_r = \Phi \cdot A \cdot f_y \cdot \left(1+\lambda^{2\cdot n}\right)^{\frac{-1}{n}}$$

where

r = the radius of gyration of the appropriate axis

E = Young's modulus for steel

K = effective length factor of the member

L_y = the buckling length about the y-axis of the member

x_o = the principal x coordinate of the shear centre of the profile in respect to the centroid of the section

y_o = the principal y coordinate of the shear centre of the profile in respect to the centroid of the section

The compressive resistance is then calculated using the formula in 4.2.3. The value of f_e used in 4.2.3 is then taken as the smaller of the two values calculated, f_{ex} and f_{eyz}.
For the case of equal leg angle profiles, the u-u axis of the profile is taken as the axis of symmetry, namely y-y, while the v-v axis of the profile is used as the x-x axis in the above formulae. This is illustrated by Figure 4-1

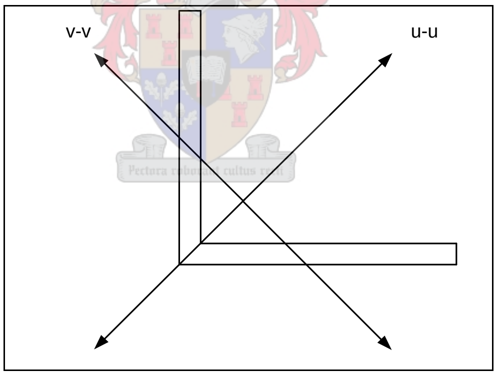


Figure 4-1 Local axis system of equal leg angle profile

4.2.4.2 Asymmetric sections

For all asymmetric sections, the value of compressive resistance is calculated as follows:

$$\left(f_e - f_{ex}\right)\left(f_e - f_{ey}\right)\left(f_e - f_{ez}\right) - f_e^{\,2}\left(f_e - f_{ey}\right)\left(\frac{x_o}{\bar{r}_o}\right)^2 - f_e^{\,2}\left(f_e - f_{ex}\right)\left(\frac{y_o}{\bar{r}_o}\right)^2 = 0 \qquad \text{[SANS 10162 Part1: 13.3.2c)]}$$

with,

$$f_{ex} = \frac{r_x^{\,2} \cdot \pi^2 \cdot E}{K^2 \cdot L_x^{\,2}}$$

$$f_{ey} = \frac{r_y^{\,2} \cdot \pi^2 \cdot E}{K^2 \cdot L_y^{\,2}}$$

$$f_{ez} = \left(\frac{\pi^2 \cdot E \cdot C_w}{K_z^{\,2} \cdot L_z^{\,2}} + G \cdot J \right) \cdot \frac{1}{A \cdot \bar{r}_o^{\,2}}$$

$$\bar{r}_o^{\,2} = r_x^{\,2} + r_y^{\,2} + x_o^{\,2} + y_o^{\,2}$$

where

r = the radius of gyration for the particular axis

L = the appropriate buckling length

K = the effective length factor of the member

E = Young’s modulus for steel

G = shear modulus of steel

J = St. Venant torsion constant

A = cross sectional area of the profile

C_W = warping constant for the section, 0 for hollow sections

x_o = the principal x coordinate of the shear centre of the profile in respect to the centroid of the section

y_o = the principal y coordinate of the shear centre of the profile in respect to the centroid of the section

The above formula in *SANS 10162 Part1 13.3.2 c)* is then solved for f_e. The smallest root (f_e) is then used in 4.2.3 to calculate the compressive resistance of the member.

4.3 Beam-Columns

All members subjected to flexural bending as well as compressive axial forces are considered in this section. In this clause a distinction is made between braced and unbraced frames. A frame with direct acting bracing is classified as braced when its sway stiffness is at least five times that of the frame without direct acting bracing.

Implementation aspects:

In the prototype application, the designer has the option to state whether a particular member of a larger structure, i.e. frame, is braced against sway effects or not provided the bracing is adequate according to design parameters. The application does not test the adequacy of bracing.

The following clauses from *SANS 10162 Part1* are used primarily in the design of members under axial compression and bending:

- 13.8 Axial compression and bending
- 13.5 Bending-Laterally supported members
- 13.6 Bending-Laterally unsupported members
- 13.3 Axial compression
- 10.2 Effective lengths of flexural members
- 10.3 Members in compression

Implementation aspects:

It was decided for the case of beam-column design to assume that a second order analysis was used in determining the forces and end moments of the member. This results in, as stated in *SANS 10162 Part1 10.3.2*, that the effective length factor used for determining the in plane Euler buckling length is to be taken as 1.0.
The effective length factor that is used in the determination of the lateral torsional buckling length and out of plane Euler buckling length is still left up to the designer to decide. These factors are determined by the choice of end constraints that the designer chooses. These end constraints are synonymous with the end constraints available with beam and or column design. Only I and H - profiles of class 1, 2 and 3 as well as Channel profiles were considered in the prototype application. Furthermore, only uniaxial strong axis bending is considered for design in the prototype application. All the aforementioned assumptions and limitations to the design procedure implemented are illustrated in more detail in the following sections.

4.3.1 Maximum slenderness ratios

According to *SANS 10162 Part1 10.2,* the maximum slenderness ratio of a member in compression shall not exceed 200.
The slenderness ratio is calculated as follows:

$$\frac{KL}{r} \leq 200 \qquad \text{[SANS 10162 Part1: 10.4]}$$

where

K = the effective length factor of the member.

L = the actual length of the member.

r = the appropriate radius of gyration for the particular axis under consideration.

4.3.2 Effective length factors

The calculation of the effective length factor for lateral torsional buckling of the member is identical to that mentioned in 4.1.1.1 or 4.1.1.2 respectively, depending on whether the beam column is simply supported or of a cantilever type.
The calculation of the effective length factor for out of plane Euler buckling is identical to that mentioned in 4.2.2.

4.3.3 Member strength and stability of class 1 and 2 I shaped sections

For the case of class 1 and 2 I and H sections, design is based on a formula that describes the interaction of axial compression and bending. This formula is commonly known as the interaction formula. The interaction formula, applicable for uniaxial strong axis bending, for such profiles is as follows:

$$\frac{C_u}{C_r} + \frac{0.85 \cdot U_{1x} \cdot M_{ux}}{M_{rx}} \leq 1.0 \qquad \text{[SANS 10162 Part1: 13.8.2]}$$

The values C_u and M_{ux} refer to the ultimate factored axial compression and bending moment respectively that occur within the member.
The factor U_{1x} is to account for moment gradient and for second-order effects of axial force acting on the deformed member.
As stated in *SANS 10162 Part1: 13.8* the member under design has to be examined for 3 cases of strength and these cases applied to the interaction formula as previously mentioned.

These cases are briefly described as follows:

- **Cross sectional strength**

  For this case, the parameters of the interaction formula are based on the pure cross sectional strength of the profile under design. This test is only applicable for members in braced frames.
- **Overall member strength**

  In this case, the resistance due to axial compression is based on Euler buckling whilst the resistance to bending is calculated on the cross sectional strength of the profile.

  In the first part of the interaction formula ($\frac{C_u}{C_r}$), the value of C_r is based on *SANS 10162 Part1 13.3: Axial Compression.* The latter part of the interaction formula refers to the bending resistance of the member, $\frac{0.85 \cdot U_{1x} \cdot M_{ux}}{M_{rx}}$. The value of M_{rx} is calculated as stated in *SANS*

10162 Part1: 13.5 Bending: laterally supported members irrespective of whether the member is laterally supported or not.

- **Lateral torsional buckling strength**

In this case, the value of M_{ux} used in the latter half of the interaction formula, $\frac{0.85 \cdot U_{1x} \cdot M_{ux}}{M_{rx}}$, is calculated as stated in *SANS 10162 Part1: 13.6 Bending-laterally unsupported members.* For the first part of the interaction formula, $\frac{C_u}{C_r}$, the value of C_r is calculated as stated in the clause *SANS 10162 Part1: 13.3 Axial compression,* and is based on either weak axis buckling (*SANS 10162 Part1: 13.3.1 Flexural buckling*) or torsional flexural buckling (*SANS 10162 Part1: 13.3.2 Torsional or Torsional- flexural buckling*). In the case that the member is continuously laterally supported along its weak axis, this test for buckling strength in the design is not applicable.

Each step in the implementation of the design is described in more detail in the following sections.

4.3.3.1 Cross sectional strength

In this section the cross sectional strength of the member is tested. Only members that are part of a braced frame are considered in this section. All values calculated must in the end satisfy the interaction formula below.

$$\frac{C_u}{C_r} + \frac{0.85 \cdot U_{1x} \cdot M_{ux}}{M_{rx}} \leq 1.0$$

The value of C_u is taken as the maximum factored compressive axial force occurring along the length of the member. The value of C_r is calculated according to *SANS 10162 Part1: 13.3 Axial compression* with the value of λ taken equal to 0. This is illustrated as follows:

$$C_r = \Phi \cdot A \cdot f_y \cdot \left(1+\lambda^{2n}\right)^{\frac{-1}{n}}$$

with $\lambda = 0$

$$\therefore C_r = \Phi \cdot A \cdot f_y$$

where,

C_r = the compressive resistance of the member.

Φ = 0.9

A = the cross sectional area of the member.

f_y = the yield stress of steel

The value of M_{ux} is taken as the maximum factored bending moment about the strong axis of the member. The value of M_{rx} is calculated according to *SANS 10162 Part1: 13.5 Bending-laterally supported members,* irrespective of whether the member is laterally supported or not. This is illustrated as follows:

$M_{rx} = \Phi \cdot Z_{plx} \cdot f_y$ (for class 1 and 2 sections)

The value of U_{ix} is calculated according to *SANS 10162 Part1: 13.8.4 Value of U_1* but not taken as less than 1.0.

$$U_{ix} = \frac{\omega_1}{1 - \frac{C_u}{C_{ex}}}$$

The value of C_u is the same value used in the interaction formula, namely the maximum factored compressive axial force within the member.

C_{ex} is the Euler buckling strength of the member about the strong axis. This is described as follows:

$$C_{ex} = \frac{\pi^2 \cdot E \cdot I_x}{K \cdot L_x}$$

where,

E = Young's modulus for steel

I_x = the moment of inertia for the particular section

L_x = the unbraced portion of the length of the member

K = the effective length factor of the member

The value of ω_1 is defined according to *SANS 10162 Part1: 13.8.4 Values of* ω_1.

a) **ω_1 for members not subjected to transverse loads between supports**

$\omega_1 = 0.6 - 0.4 \cdot \kappa \geq 0.4$ [SANS 10162 Part1: 13.8.4 a)]

The value of κ is defined in *SANS 10162 Part1: 13.6) Bending-Laterally unsupported members.* It is the ratio of the smaller to the larger end moment at opposite ends of the unbraced length, positive for double curvature and negative for single curvature.

b) **ω_1 for members subject to distributed loads or a series of point loads between supports**

$\omega_1 = 1.0$ [SANS 10162 Part1: 13.8.4 b)]

c) **ω_1 for members subject to a concentrated load or moment between supports**

$\omega_1 = 0.85$ [SANS 10162 Part1: 13.8.4 c)]

4.3.3.2 Overall member strength

In this section the overall member strength of the member is tested. All values calculated must in the end satisfy the interaction formula below.

$$\frac{C_u}{C_r} + \frac{0.85 \cdot U_{1x} \cdot M_{ux}}{M_{rx}} \leq 1.0$$

The value of C_u is taken as the maximum factored compressive axial force occurring along the length of the member. The value of C_r is calculated according to *SANS 10162 Part1: 13.3 Axial compression* with the value of K = 1.0 for the case of bi-axial bending or weak axis bending. In the case of strong axis uni-axial bending, $C_r = C_{rx}$.

$$C_r = \Phi \cdot A \cdot f_y \cdot \left(1 + \lambda^{2n}\right)^{\frac{-1}{n}}$$

where

A = cross sectional area of the profile

$\Phi = 0.9$

f_y = yield stress for steel

n = 1.34 for hot-rolled, fabricated structural sections, and hollow structural sections manufactured according to SANS 657-1 (cold-formed non-stress-relieved)

with

$$\lambda = \frac{K \cdot L}{r}\sqrt{\frac{f_y}{\pi^2 \cdot E}} = \sqrt{\frac{f_y}{f_e}}$$

where

K = the effective length factor

L_x = the unbraced length of the member

r_x = the radius of gyration for the x- axis

f_y = the yield stress of steel

E = Young’s modulus for steel

The effective length factor *K* is taken as 1.0 due to the assumption of a second order analysis.

The value of M_{ux} is taken as the maximum factored bending moment about the strong axis of the member. The value of M_{rx} is calculated according to *SANS 10162 Part1: 13.5 Bending-laterally supported members,* irrespective of whether the member is laterally supported or not. This is the same as is in the previous section, 4.3.3.1.

$$M_{rx} = \Phi \cdot Z_{plx} \cdot f_y$$

The value of U_{ix} is calculated according to *SANS 10162 Part1: 13.8.4 Value of U_1* for members in braced frames. If the member is part of an unbraced frame, the value of U_{ix} is taken as 1.0. For the case of a braced frame, U_{ix} is calculated as in 4.3.3.1

The exception in this section is that the value calculated can be less than 1.0. The procedure is as follows:

$$U_{ix} = \frac{\omega_1}{1 - \frac{C_u}{C_e}}$$

The value of C_u is the same value used in the interaction formula, namely the maximum factored compressive axial force within the member.

C_e is the Euler buckling strength of the member. This is described as follows:

$$C_e = \frac{\pi^2 \cdot E \cdot I_x}{K \cdot L_x}$$

where

E = Young's modulus for steel

I_x = the moment of inertia for the particular section

L_x = the unbraced portion of the length of the member

K = the effective length factor of the member

The value of K in the formula is taken as 1.0 based on the assumption that a second order analysis has been done on the structure.

The value of ω_1 is defined according to *SANS 10162 Part1: 13.8.4 Values of* ω_1.

a) **ω_1 for members not subjected to transverse loads between supports**

$\omega_1 = 0.6 - 0.4 \cdot \kappa \geq 0.4$ [SANS 10162 Part1: 13.8.4 a)]

The value of κ is defined in *SANS 10162 Part1: 13.6 Bending-Laterally unsupported members.* It is the ratio of the smaller to the larger end moment at opposite ends of the unbraced length, positive for double curvature and negative for single curvature.

b) **ω_1 for members subject to distributed loads or a series of point loads between supports**

$\omega_1 = 1.0$ [SANS 10162 Part1: 13.8.4 b)]

c) **ω_1 for members subject to a concentrated load or moment between supports**

$\omega_1 = 0.85$ [SANS 10162 Part1: 13.8.4 c)]

4.3.3.3 Lateral torsional buckling strength (if applicable)

In this section the lateral torsional buckling strength of the member is tested. If the member is laterally supported along its entire length, this section does not apply. All values calculated must in the end satisfy the interaction formula below.

$$\frac{C_u}{C_r} + \frac{0.85 \cdot U_{1x} \cdot M_{ux}}{M_{rx}} \leq 1.0$$

The value of C_u is taken as the maximum factored compressive axial force occurring along the length of the member. The value of C_r is based on:

- Weak axis buckling according to *SANS 10162 Part1: 13.3.1 Flexural buckling,*

The compressive resistance is calculated according to this clause. This is illustrated as follows:

$$C_{rFBweak} = \Phi \cdot A \cdot f_y \cdot \left(1+\lambda^{2n}\right)^{-\frac{1}{n}}$$, with

$$\lambda = \frac{K \cdot L_{weak}}{r_{weak}} \sqrt{\frac{f_y}{\pi^2 \cdot E}} = \sqrt{\frac{f_y}{f_e}}$$

where

$C_{rFBweak}$ = the flexural buckling compressive resistance of the member

A = the cross sectional area of the profile

Φ = 0.9

n = 1.34 for hot-rolled, fabricated structural sections, and hollow structural sections manufactured according to SANS 657-1 (cold-formed non-stress-relieved)

f_y = the yield stress of steel

K = 1.0

L_{weak} = the longest unbraced length of the weak axis of the member

r_{weak} = the radius of gyration about the weak axis of the profile

E = Young's modulus for steel

The value of M_{ux} is taken as the maximum factored bending moment about the strong axis of the member. The value of M_{rx} is calculated according to *SANS 10162 Part1 13.6 Bending-laterally unsupported members.* This is different from sections 4.3.3.1 and 4.3.3.2.

The plastic yield moment of class 1 and 2 profiles is taken as $M_p = f_y \cdot Z_{plx}$.

- When $M_{cr} \geq 0.67M_p$

$M_{rx} = 1.15\Phi \cdot M_p \left(1 - 0.28\dfrac{M_p}{M_{cr}}\right)$, but not exceeding $\Phi \cdot M_p$

- When $M_{cr} \leq 0.67M_p$

$M_{rx} = \Phi \cdot M_{cr}$

The critical moment is calculated as follows, $M_{cr} = \dfrac{\omega_2 \cdot \pi}{K \cdot L}\sqrt{E \cdot I_y \cdot G \cdot J + \left(\dfrac{\pi \cdot E}{K \cdot L}\right)^2 I_y \cdot C_w}$

with $\omega_2 = 1.75 + 1.05 \cdot \kappa + 0.3 \cdot \kappa^2 \leq 2.5$

where

M_r = the factored moment resistance

M_p = the plastic moment

M_y = the yield moment

M_{cr} = the critical elastic moment of an unbraced member.

$\Phi = 0.9$

κ = the ratio of the smaller to the larger ultimate end moment at opposite ends, positive for double curvature and negative for single curvature.

I_y = the second moment of inertia of the section about its weak axis.

E = elastic modulus of steel

G = shear modulus of steel.

J = St. Venant torsion constant.

C_w = warping torsion constant, equal to 0 for structural hollow sections.

K = effective length factor.

L = unbraced length of member.

The value of U_{ix} is calculated according to *SANS 10162 Part1: 13.8.4 Value of U_1* for members in braced frames. If the member is part of an unbraced frame, the value of U_{ix} is taken as 1.0. For the case of a braced frame, U_{ix} is calculated as in 4.3.3.1 but not less than 1.0

The procedure is as follows:

$$U_{ix} = \frac{\omega_1}{1 - \dfrac{C_u}{C_e}}$$

The value of C_u is the same value used in the interaction formula, namely the maximum factored compressive axial force within the member.

C_e is the Euler buckling strength of the member. This is described as follows:

$$C_e = \frac{\pi^2 \cdot E \cdot I_x}{K \cdot L_x}$$

where

E = Young's modulus for steel

I_x = the moment of inertia for the particular section

L_x = the unbraced portion of the length of the member

K = the effective length factor of the member

The value of *K* in the formula is taken as 1.0 based on the previous assumption made that a second order analysis has been done on the structure.

The value of ω_1 is defined according to *SANS 10162 Part1: 13.8.4 Values of* ω_1.

a) **ω_1 for members not subjected to transverse loads between supports**

$\omega_1 = 0.6 - 0.4 \cdot \kappa \geq 0.4$ [SANS 10162 Part1: 13.8.4 a)]

The value of κ is defined in *SANS 10162 Part1: 13.6 Bending-Laterally unsupported members.* It is the ratio of the smaller to the larger end moment at opposite ends of the unbraced length, positive for double curvature and negative for single curvature.

b) **ω_1 for members subject to distributed loads or a series of point loads between supports**

$\omega_1 = 0$ [SANS 10162 Part1: 13.8.4 b)]

c) **ω_1 for members subject to a concentrated load or moment between supports**

$\omega_1 = 0.85$ [SANS 10162 Part1: 13.8.4 c)]

4.3.3.4 Moment Interaction

Apart from the interaction formula stated in 4.3.3.3, the member must comply to the following moment interaction formula, as stipulated in *SANS 10162 Part1: 13.8.2*. This is as follows:

$\frac{M_{ux}}{M_{rx}}$ [SANS 10162 Part1: 13.8.2]

where

M_{ux} = the ultimate moment present in the member

M_{rx} = the moment of resistance of the member, defined as stipulated in either 4.1.2.1 or 4.1.2.2 depending on whether the member is laterally braced or not.

4.3.4 Member strength and stability of class 3 I – shaped sections and Channel sections

For the case of class 3 I and H sections as well as Channel sections, design is applied according to *SANS 10162 Part1: 13.8.3 Member strength and stability-All classes except class 1 and 2 I shaped sections.* The design procedure is similar to the design procedure in 4.3.3 except the interaction formula used is slightly modified. This is shown below:

$$\frac{C_u}{C_r} + \frac{U_{1x} \cdot M_{ux}}{M_{rx}} \leq 1.0 \quad \text{[SANS 10162 Part1: 13.8.2]}$$

For the calculation of the compressive resistance of Channel profiles, the procedure mentioned in 4.2.4.2 is followed.

Another difference occurs in the calculating of the moment of resistance as the formulae $M_{rx} = \Phi \cdot Z_{plx} \cdot f_y$ and $M_p = Z_{plx} \cdot f_y$ are replaced by $M_{rx} = \Phi \cdot Z_{ex} \cdot f_y$ and $M_y = Z_{ex} \cdot f_y$ (*SANS 10162 Part1: 13.5 Bending-Laterally supported members*) respectively.

4.4 Axial tension and bending

The design of a member subjected to combined bending and axial tension shall be done according to *SANS 10162 Part1: 13.9 Axial tension and bending.* Within the scope of this thesis, the calculation of tensile strength of a member was limited solely to the strength of the member itself. The strength of the member due to the connections at its ends was not included in the design.

4.4.1 Maximum slenderness ratios

According to *SANS 10162 Part1 10.2.2*, the maximum slenderness ratio of a member in tension shall not exceed 300.

The slenderness ratio is calculated as follows:

$$\frac{K \cdot L}{r} \leq 300 \qquad \text{[SANS 10162: Part1: 10]}$$

where

K = the effective length factor of the member.

L = the actual length of the member.

r = the appropriate radius of gyration for the particular axis under consideration.

4.4.2 Effective length factors

This set of effective length factors is used in the lateral torsional buckling part of the design of tension beams. Where tension beams have no restraint against torsion, the values of the effective length factor K in Table 4-4 shall be increased by 20%.

For tension beams that are provided with members giving effective lateral restraint to the compression flange at intervals along the span, in addition to the torsional restraint as required above, the effective length shall be taken as the distance, centre to centre, between the restraint members.

Implementation aspects:

As stated in the previous sections, the prototype application allows for the user to stipulate any additional support along the length of the member in addition to the effective length factors that are determined by the end conditions of the member, thus reducing effective lengths for lateral torsional buckling resistance.

Table 4-4 Effective length factors

1	2	3
Restraint against lateral bending at supports	**Effective length factor K**	
	Loading condition	
	Normal	**Destabilizing**
Unrestrained (i.e. free to rotate in plane)	1.0	1.2
Partially restrained (i.e. positive connection by flange cleats or end plates)	0.85	1.0
Practically fixed (i.e. not free to rotate in plan)	0.7	0.85

4.4.3 Axial tension and bending design

Members are proportioned according to the interaction formula as follows:

$$\frac{T_u}{T_r} + \frac{M_u}{M_r} \leq 1.0$$ [SANS 10162 Part1: 13.9]

where

T_u = the maximum factored tensile force occurring along the member

Implementation aspects:

The calculation of tensile strength of a member by the prototype application was limited solely to the strength of the member itself. The strength of the member due to the connections at its ends was not included. The tensile resistance is therefore calculated in the following sections.

$T_r = \Phi \cdot A_g \cdot f_y$ [SANS 10162 Part1: 13.2a subclause i)]

where

T_r = the tensile resistance of the member

Φ = 0.9

A_g = the gross area of the member, i.e. the area of the entire cross section of the member.

f_y = the yield stress of steel

The value of M_r is calculated according to *SANS 10162 Part1: Bending-laterally supported members.*

M_r for profiles of class 1 and 2:

$M_r = \Phi \cdot Z_{pl} \cdot f_y$

where

M_r = the factored moment of resistance

Φ = 0.9

Z_{pl} = the plastic section modulus of a section

f_y = the yield stress of steel

M_r for profiles of class 3:

$M_r = \Phi \cdot Z_e \cdot f_y$

where

M_r = the factored moment of resistance

Φ = 0.9

Z_e = the elastic section modulus of a section

f_y = the yield stress of steel

Furthermore, the member shall resist the following interaction criteria as well.

For profiles of class 1 and 2, the member shall be proportioned such that,

$$\frac{M_u}{M_r} - \frac{T_u \cdot Z_{pl}}{M_r \cdot A} \leq 1.0$$

where

M_u = the maximum factored moment occurring along the length of the member

T_u = the maximum factored tensile force occurring within the member

A = the gross area of the section

The value of M_r is calculated according to either *SANS 10162 Part1:13.5 Bending-laterally supported members* or *SANS 10162 Part1: 13.6 Bending-laterally unsupported members,* depending on whether the member is laterally supported along its length or not.

a) Laterally supported members

$$M_r = \Phi \cdot Z_{pl} \cdot f_y$$

where

M_r = the factored moment of resistance

$\Phi = 0.9$

Z_{pl} = the plastic section modulus of a section

f_y = the yield stress of steel

b) Laterally unsupported members

When $M_{cr} > 0.67\ M_p$

$$M_r = 1.15 \cdot \Phi \cdot M_p \left(1 - 0.28 \frac{M_p}{M_{cr}}\right)$$, but not exceeding $\Phi \cdot M_p$

When $M_{cr} \leq 0.67\ M_p$

$$M_r = \Phi \cdot M_{cr}$$

with

$$M_p = Zpl \cdot f_y$$

$$M_{cr} = \frac{\omega_2 \cdot \pi}{K \cdot L} \sqrt{E \cdot I_y \cdot G \cdot J + \left(\frac{\pi \cdot E}{K \cdot L}\right)^2 I_y \cdot C_w}$$

$$\omega_2 = 1.75 + 1.05 \cdot \kappa + 0.3 \cdot \kappa^2 \leq 2.5$$

where

M_r = the factored moment resistance

M_p = the plastic moment

M_y = the yield moment

M_{cr} = the critical elastic moment of an unbraced member.

$\Phi = 0.9$

κ = the ratio of the smaller to the larger ultimate end moment at opposite ends, positive for double curvature and negative for single curvature.

I_y = the moment of inertia of the section about its weak axis.

E = elastic modulus of steel

G = shear modulus of steel.

J = St. Venant torsion constant.

C_w = warping torsion constant, equal to 0 for structural hollow sections.

K = effective length factor.

L = Unbraced length of member.

For profiles of class 3, the member shall be proportioned such that,

$$\frac{M_u}{M_r} - \frac{T_u \cdot Z_e}{M_r \cdot A} \leq 1.0$$

where

M_u = the maximum factored moment occurring along the length of the member

T_u = the maximum factored tensile force occurring within the member

A = the gross area of the section

The value of M_r is calculated according to either *SANS 10162 Part1:13.5 Bending-laterally supported members* or *SANS 10162 Part1: 13.6 Bending-laterally unsupported members,* depending on whether the member is laterally supported along its length or not.

a) Laterally supported members

$$M_r = \Phi \cdot Z_e \cdot f_y$$

where

M_r = the factored moment of resistance

$\Phi = 0.9$

Z_e = the elastic section modulus of a section

f_y = the yield stress of steel

b) Laterally unsupported members

When M_{cr} > 0.67 M_y

$$M_r = 1.15\Phi \cdot M_y \left(1 - 0.28\frac{M_y}{M_{cr}}\right)$$, but not exceeding $\Phi \cdot M_y$

When $M_{cr} \leq 0.67 M_y$

$$M_r = \Phi \cdot M_{cr}$$

with

$$M_y = Z_e \cdot f_y$$

$$M_{cr} = \frac{\omega_2 \cdot \pi}{K \cdot L}\sqrt{E \cdot I_y \cdot G \cdot J + \left(\frac{\pi \cdot E}{K \cdot L}\right)^2 I_y \cdot C_w}$$

$$\omega_2 = 1.75 + 1.05 \cdot \kappa + 0.3 \cdot \kappa^2 \leq 2.5$$

where

M_r = the factored moment resistance

M_y = the yield moment

M_{cr} = the critical elastic moment of an unbraced member.

Φ = 0.9

κ = the ratio of the smaller to the larger ultimate end moment at opposite ends, positive for double curvature and negative for single curvature.

I_y = the second moment of inertia of the section about its weak axis.

E = elastic modulus of steel

G = shear modulus of steel.

J = St. Venant torsion constant.

C_w = warping torsion constant, equal to 0 for structural hollow sections.

K = effective length factor.

L = Unbraced length of member.

4.5 Tension members

The factored tensile resistance of a member subjected to tensile loading is calculated in this section. It was decided to limit the design of members under axial tension to cross sectional strength, and not connection strength due to the conditions at the members ends. The design was then simplified to *SANS 10162 Part1: 13.2 Axial compression and connection strength a) (i).*

The tensile resistance is then calculated as follows:

$$T_r = \Phi \cdot A_g \cdot f_y$$

where

T_r = the tensile resistance of the member

$\Phi = 0.9$

A_g = the gross area of the cross section of the profile

f_y = the yield stress of steel

Implementation aspects:

The prototype application only calculates the general cross sectional tensile resistance of the steel member. The end connections are not considered.

4.6 Shear resistance

The shear resistance of members is determined according to *SANS 10162 Part1: 13.4 Shear*.
The shear resistance, based on elastic analysis, of a member having two flanges is as follows:

$V_r = \phi \cdot A_v \cdot f_s$ [SANS 10162 Part1: 13.4.1.1]

where

V_r = the factored shear resistance of the member

$\Phi = 0.9$

A_v = the effective shear area

f_s = the ultimate shear strength of the section.

The value of f_s is taken as 0.66 f_y due to the assumption made that there are no web stiffners present.
The effective shear area is taken as the area of the web of the profile, $h(t_w)$.
For sections having only one flange, the shear stress at any point on the cross section of the profile may not exceed $0.66 \cdot \Phi \cdot f_y$.

The shear stress of the cross section at a particular cut is calculated by the formula

$$\tau = \frac{V \cdot A^* \cdot \bar{y}}{A \cdot I}$$

where

τ = the shear stress in the section

V= the applied shear force

A^* = the partial area of the cross section at the particular cut

A = the total area of the cross section

I = moment of inertia of the total cross section

5 Structural steel sections

The term “structural steel” is used to define steel members whose primary purpose is to support the loads or resist the forces which act on a structure. The main aim of this thesis was to develop a framework for the design of hot rolled steel sections according to *SANS 10162-1:2004: The structural use of steel Part1: Limit states design of hot-rolled steelwork.* Thus the types of rolled steelwork used in the design process developed in this thesis are:

- Hot rolled steel sections
- Structural hollow sections manufactured according to *SABS 657-1 (cold formed, non-stress relieved).*

The following sections describe the types of steel profiles used in the prototype application and the classification of these profiles for effective design.

5.1 Classification of steel sections

General: For the purpose of the design of elements in compression in *SANS 10162 Part1*, the structural steel sections used have to be designated as class 1, 2, 3 or 4. Members in compression, whether axial or flexural compression, are subject to certain limits on width – thickness ratios of their cross sections to ensure that they will not buckle locally under load. The behaviour of a member, within a structure, is dependent on its classification. This is done according to *SANS 10162 Part1: Width thickness ratios – Elements in compression* where the classification of elements is dependent on their respective maximum width – thickness ratios.

Classification: The classification system implemented divides steel profiles into certain classes describing their cross sectional local buckling behaviour under load. Once a profile is classified, the appropriate design procedure is used for that specific class of steel profile.
The classes are defined as follows:

a) **Class 1** sections (plastic design sections) will permit attainment of the plastic moment and subsequent redistribution of the bending moment.
b) **Class 2** sections (compact sections) will permit attainment of the plastic moment but need not allow for subsequent redistribution of the bending moment.
c) **Class 3** sections (non – compact sections) will permit attainment of the yield moment.
d) **Class 4** sections (slender sections) will generally have local buckling of elements in compression at the limit state of structural resistance.

The classification process is dependent on cross sectional geometry of the steel profile. The determination of the type of class a steel profile belongs to is determined by the width to thickness ratio of portions of a steel profile's cross section.

The value of the maximum width – thickness ratio (*W*) is calculated by the formula:

$$W = \frac{b}{t}$$

where

W = the width – thickness ration of the specific section

b = the nominal width of the section

t = the nominal thickness of the section

The value of W is calculated for both the web and the flange of the respective steel section. The definition of b and t is defined differently for the web and the flange of the steel profile's cross section. The value of the nominal width (b) of the steel section is defined in either *SANS 10162 Part1: 13.3.1 or 13.3.2,* depending on the layout of the aforementioned cross section.

Below is an extract from *SANS 10162 Part1: 11.3 Width and thickness.*

For elements supported along only one edge parallel to the direction of the compressive force, the value of b is calculated as follows:

- *For legs of angles, flanges of channels and stems of tees: the full nominal dimension.*
- *For flanges of beams and tees: one half of the full nominal dimension*

For elements supported along two edges parallel to the direction of the compressive force, the width b shall be defined as follows:

- *For flanges of rectangular hollow sections, the clear distance between webs less the inside corner radius on each side.*

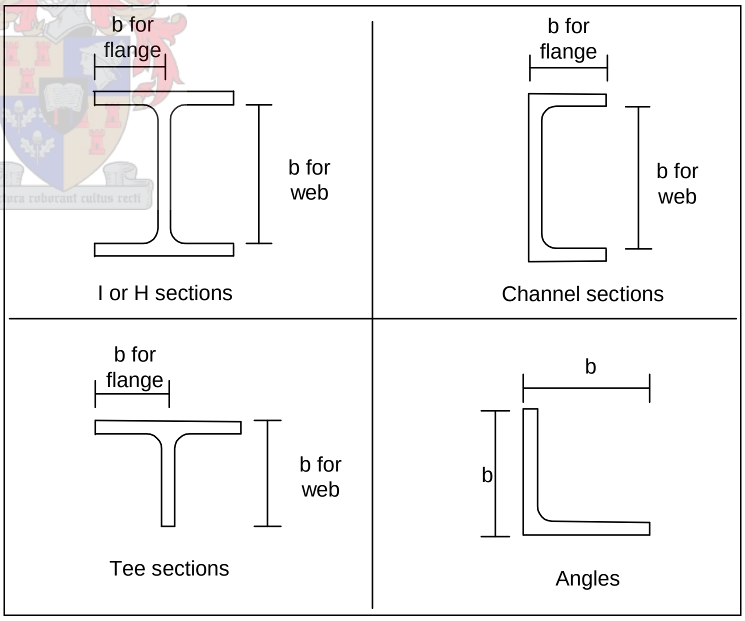


Figure 5-1 Width values used for some common steel profiles

Figure 5-1 shows the value of b for both the web and the flanges of some common steel cross sections. For steel profiles such a these, the overall classification is based on the highest class obtained between the web and the flange.

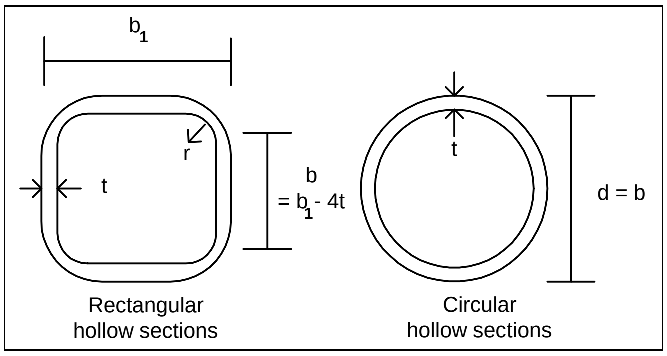


Figure 5-2 Values of b and t for some hollow sections

The value of t in the width – thickness relationship is the nominal thickness of the web or flange of the section. This value is dependent on the type of steel profile. For tapered flanges of rolled sections, the thickness *t* is taken as the nominal thickness halfway between the free edge and the corresponding face of the web. Values of b and t for some typical structural hollow sections are illustrated in Figure 5-2.

Class definitions: For steel sections that have both web and flanges the maximum compression class calculated is taken as governing. Structural steel sections are classified according to the type of compression present in the member. This compression is either axial compression, flexural compression or combined axial and flexural compression. The class definitions are defined in *SANS 10162: Part1 Table 3 Maximum width to thickness ratios – Elements in compression* as well as *SANS 10162: Part1 Table 4 Maximum with to thickness ratios – Elements in flexural compression.*

Axial compression: Table 5-1 illustrates the limits that determine the class of a section that is subjected to axial compression using the maximum width – thickness ratio calculated. All profiles that have width – thickness ratios that fall within the limits of the inequalities shown are designated as class 3 profiles. All profiles that have width – thickness ratios that exceed the following inequalities are taken as class 4 profiles. The values of f_y in the formulae is the ultimate yield stress of the applicable steel profile or section.

Table 5-1 Width-thickness ratios: Elements in compression

1	2
Description of element	**Maximum width-to-thickness ratio W**
Elements supported along one edge Flanges of I-sections, T-sections, and channels Legs of angles, plate girder stiffners	$\frac{b}{t} \leq \frac{200}{\sqrt{f_y}}$

1	2
Description of element	**Maximum width-to-thickness ratio W**
Stems of T-sections	$\frac{b}{t} \leq \frac{340}{\sqrt{f_y}}$
Flanges of rectangular hollow sections Flanges of box sections,	$\frac{b}{t} \leq \frac{670}{\sqrt{f_y}}$
Flange cover plates, and diaphragm plates between lines of fasteners or welds Webs supported on both edges	
Perforated cover plates	$\frac{b}{t} \leq \frac{840}{\sqrt{f_y}}$
Circular hollow sections	$\frac{d}{t} \leq \frac{23000}{f_y}$

Flexural compression: Table 5-2 is used to determine the class of a section that is subjected to flexural compression and or combined flexural and axial compression. For steel profiles with both a web and a flange, classification is done according to the individual classification of both the web and the flange. The maximum between these two values is taken as the overall classification of the steel profile.

Table 5-2 Width - thickness ratios: Elements in flexural compression

1	2	3	4
Section classification			
Description of element in compression	**Class 1**	**Class 2**	**Class 3**
Flanges of I-sections or T-sections Plates projecting from compression elements Outstanding legs of pairs of angles in continuous contact with an axis of symmetry in the plane of loading	$\frac{b}{t} \leq \frac{145}{\sqrt{f_y}}$	$\frac{b}{t} \leq \frac{170}{\sqrt{f_y}}$	$\frac{b}{t} \leq \frac{200}{\sqrt{f_y}}$

1	2	3	4
Section classification			
Description of element in compression	**Class 1**	**Class 2**	**Class 3**
Stems of T-sections	$\frac{b}{t} \le \frac{145}{\sqrt{f_y}}$	$\frac{b}{t} \le \frac{170}{\sqrt{f_y}}$	$\frac{b}{t} \le \frac{340}{\sqrt{f_y}}$
Flanges of rectangular hollow sections	$\frac{b}{t} \le \frac{420}{\sqrt{f_y}}$	$\frac{b}{t} \le \frac{525}{\sqrt{f_y}}$	$\frac{b}{t} \le \frac{670}{\sqrt{f_y}}$
Flanges of box sections Flange cover plates and diaphragm plates between lines of fasteners or welds	$\frac{b}{t} \le \frac{525}{\sqrt{f_y}}$	$\frac{b}{t} \le \frac{525}{\sqrt{f_y}}$	$\frac{b}{t} \le \frac{670}{\sqrt{f_y}}$
Webs	$\frac{h_w}{t_w} \le \frac{1100}{\sqrt{f_y}}\left(1-0,39\frac{C_u}{\Phi \cdot C_y}\right)$	$\frac{h_w}{t_w} \le \frac{1700}{\sqrt{f_y}}\left(1-0,61\frac{C_u}{\Phi \cdot C_y}\right)$	$\frac{h_w}{t_w} \le \frac{1900}{\sqrt{f_y}}\left(1-0,65\frac{C_u}{\Phi \cdot C_y}\right)$
Circular hollow sections	$\frac{d}{t} \le \frac{13\,000}{f_y}$	$\frac{d}{t} \le \frac{18\,000}{f_y}$	$\frac{d}{t} \le \frac{66\,000}{f_y}$

Steel sections that exceed the limits for class 3 sections were designated as class 4. The value of f_y in the inequalities in Table 5-2 is taken as the ultimate yield strength of the steel profile under consideration. The value of Φ, the resistance factor of steel, is taken as 0.9 for the inequalities in Table 5-2.

Implementation aspects: With the classification process defined it was decided to further subdivide the steel sections into groups depending on their cross sectional symmetry. This was done due to the fact that many of the design procedures implemented in *SANS 10162 Part1* require information about cross sectional symmetry of the steel members. The groups are defined as follows:

a) **Bi – symmetric** sections are sections with two axis of symmetry
b) **Mono – symmetric** sections are sections with one axis of symmetry
c) **A – symmetric** sections are sections with no axis of symmetry

This further classification was used in the design process that was implemented in the framework of this thesis. Furthermore, steel profiles classified as greater than class 3 profiles by Table 5-1 and Table 5-2 were not considered.
When selecting a particular profile, the user has the option to select the type of steel grade that is applicable. Any steel grade may be defined, depending on the preference of the user

6 Design Elements

In this chapter the use and creation of specialized 2 dimensional elements used for the design of structural members is developed.

6.1 Concept of Design Elements

Design Elements: A Design Element is a specialized element used to represent a particular structural member for design purposes. These specialized elements are created to provide adequate design information for the design process. A finite element model of a structure does not contain sufficient information to execute a proper design on the members it represents. Performing a design directly on an analysed finite element model causes uncertainty about the design parameters used given the boundary conditions used. By combining the information provided by the finite elements within the finite element model with the design information provided by the Design Elements, a more sufficient and controlled design can be achieved.
A Design Element forms the link between the design process and the finite element model, allowing for the design processes to interpret the finite element model in a manner that suits the appropriate design procedure.

Composition: In order to create the link between the design model that consists of Design Elements, and the finite elements in the finite element model, each Design Element needs to have information about which portion of a finite element model it represents.
A Design Element can consist of one or many Euler beam finite elements provided they form a continuous, straight line macro element. A Design element can therefore span over a number of finite elements and be viewed as a single member as shown in Figure 6-1. Shown in the figure is a single Design Element consisting of three finite elements, $\{d\} = \{e_2, e_3, e_4\}$. The finite elements included provide the Design Element with their loading information, internal forces as well as the displacements and rotations of their degrees of freedom

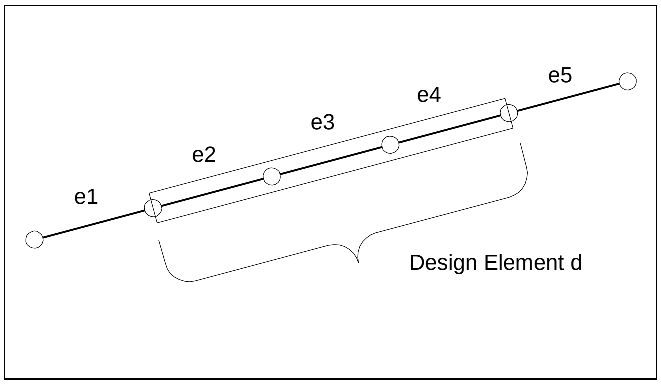


Figure 6-1 Composition of a Design Element

A structural member that is modelled by a number of finite elements is now represented by a single element, namely a Design Element, which in turn has reference to all the appropriate finite elements. A Design element consists of a start finite element, an end finite element and internal finite elements. This is shown in Figure 6-2. The start and element is interchangeable depending on the local x-x axis of the Design Element. The axis system of a Design Element is discussed later.

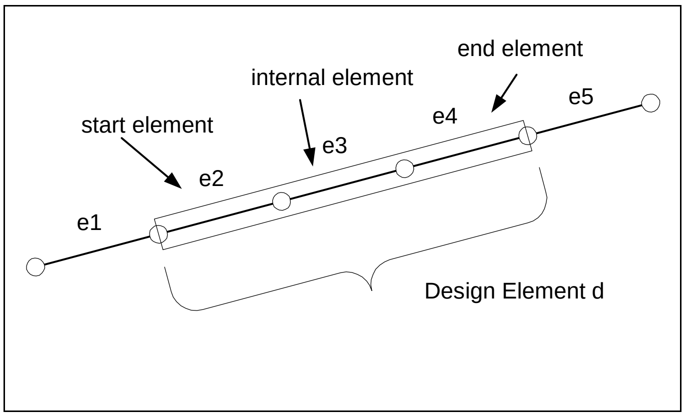


Figure 6-2 Finite elements of a Design Element

A Design Element can contain portions of finite elements as well as entire finite elements. This allows for the design model to be independent on the topology of the finite element model, while remaining dependent of the geometry of the finite elements of that same model. By stipulating an offset along the start or end finite elements, a Design Element can have its end points independent of the nodes of the included finite elements. Thus a Design Element is capable of including portions of finite elements. This functionality further allows for greater control over the design model, providing the ability to control any differences in member composition between the finite element model and the design model. As illustrated in Figure 6-3, Design Element d consists of five finite elements, $\{e_1, e_2, e_3, e_4, e_5\}$. The start offset is equal to Δ_s and the end offset is equal to Δ_e. These offsets, $\{\Delta_s, \Delta_e\}$, are measured from the furthest nodes of the start and end finite element of the Design Element. The terminal elements of the Design Element are not affected by the presence of the offsets. These offsets stipulate the end points of the Design Element relative to the start and end finite elements.

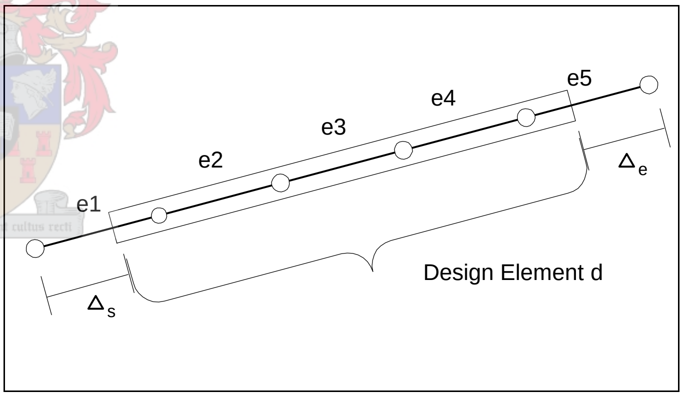


Figure 6-3 Offsets in a Design Element

Sector Points: The end points of a Design Element are called Sector Points. A Design Element includes two Sector Points. These are a start Sector Point and an end Sector Point. Sector Points can be placed at any place along the length of a finite element. This allows for the end points of a Design Element to be independent of the end points of a finite element as mentioned earlier. A Sector Point has a global x and y coordinate that is used to position the Sector Point along the length of a finite element. A Sector Point has no effect on the behaviour of the Design Element or on the design process. A Sector Point merely stipulates the start and end positions of a Design Element, as shown in Figure 6-4. The global coordinates of a Sector Point are calculated using linear interpolation along the length of the terminal (start and end) finite elements. For the example shown in Figure 6-4, with the offset, Δ_s measured in the direction shown, the coordinates of the start Sector Point are calculated as follows: $x_{\text{sector point}} = x_{n1} \cdot \Delta_s + x_{n1} \cdot (1 - \Delta_s)$ and $y_{\text{sector point}} = y_{n1} \cdot \Delta_s + y_{n1} \cdot (1 - \Delta_s)$. A more detailed description of the axis system implemented by the Design Elements is described in the following subsection.

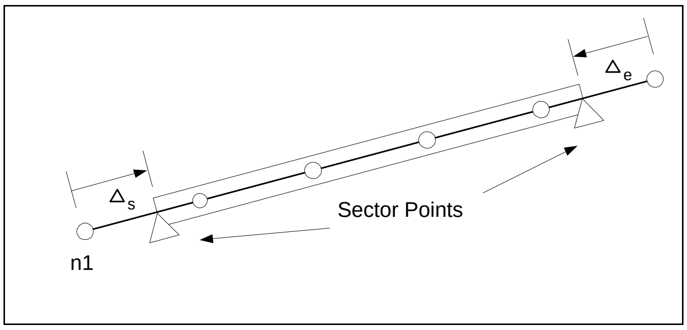


Figure 6-4 Sector Points

Coordinate System: The coordinate system of the Design Element is a the right hand system. This was discussed in chapter 3. The local x-x axis runs along the length of the element. Its direction depends on the start and end Sector Points of the Design Element, with the local x axis vector running form the start Sector Point to the end Sector Point. This is illustrated in Figure 6-5. The coordinate system of a Design Element is defined in a 3x3 coordinate matrix. This matrix contains the local axes of a Design element as unit vectors in the global coordinate system. The global coordinate system is described in chapter 3. Each column of this coordinate matrix is a 3 dimensional unit vector representing the direction and orientation of a particular local axis. These vectors are the local x axis vector, local y axis vector and local z axis vector of the particular Design Element.

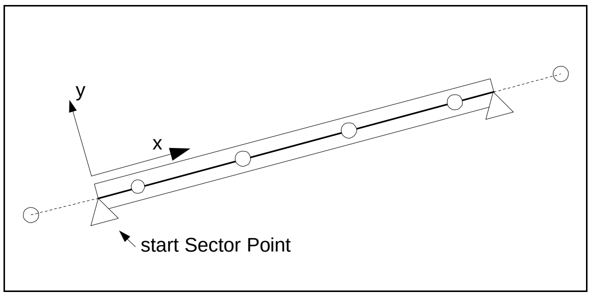


Figure 6-5 Local x axis

The coordinate matrix, K, of a general Design Element oriented in space is as follows:

$$K = \begin{pmatrix} x_x & x_y & x_z \\ y_x & y_y & y_z \\ z_x & z_y & z_z \end{pmatrix}$$

As can be seen in the coordinate matrix, the local x axis is represented globally in the first column, the local y axis in the second column and the local z axis in the last column. This coordinate matrix is essential for the correct transformation of forces, rotations and displacements from the local coordinate system of a finite element to that of the local coordinate system of a Design element. This ensures correct mapping of all the force data provided from the included finite elements. The orientation of the local axes of the included finite elements does not necessarily coincide with the local axes of the containing Design Element. This is illustrated in Figure 6-6.

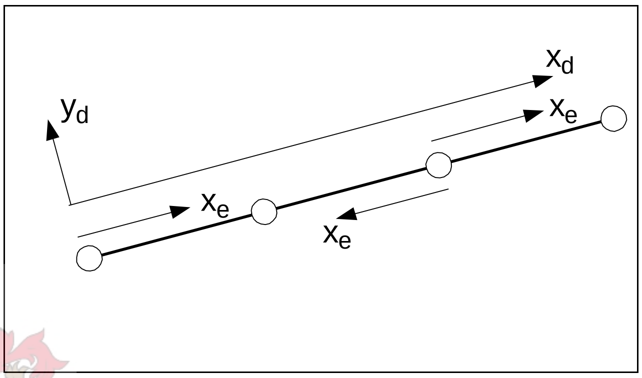


Figure 6-6 Local x axes of included elements

By transforming the force, rotational and translational data from the element's local axis system to the local axis system of the Design Element, the directions and positions of all the data are correctly mapped to the Design Element. This transformation is achieved through the use of the Design Element's coordinate matrix, K.

The transformation of the internal forces within a finite element is illustrated as follows:
The local forces of a finite element are transformed to equivalent values in the global axis system by use of the finite element's own coordinate matrix, K_e. This force vector is then transformed from the global axis system to the local axis system of the Design Element. This is illustrated by the following equations:

$$\{F_{global}\} = [K_e] \cdot \{F_e\}$$

$$\{F_{d'}\} = [K_d] \cdot \{F_{global}\}$$

$$\therefore \{F_{d'}\} = [K_d]^T \cdot [K_e] \cdot \{F_e\}$$

where

$\{F_{global}\}$ = the global internal force vector.

$[K_e]$ = the coordinate matrix of the finite element.

{F_e} = the internal force vector of a finite element.

[K_d] = the coordinate matrix of the Design Element.

{$F_{d'}$} = the untransformed internal force vector of the Design Element.

The local x – axis vector of the finite elements compared to the local x – axis vector of the Design Element is co – linear by a factor of 1 or -1. The internal force vector calculated, {$F_{d'}$}, of the Design Element has the correct orientation but the incorrect direction. A further factor is introduced to transform the internal force vector of the Design element to ensure each internal force has the correct sign. This is the dotK factor. The calculation of the dotK factor is the dot product between the local x vector of the Design Element and the local x vector of a particular included finite element. This dot product will either be equal to 1 or - 1. This value is calculated due to the fact that internal forces do not transform in the same manner as external forces. Internal forces have opposite signs depending on whether they are calculated on a positive or negative normal vector. This is illustrated in Figure 6-7. To account for this change in sign, the calculated forces, $\left\{F_{d'}\right\}$, are multiplied with the calculated dot product between the local x – axis of the Design Element and a particular included finite element. Each finite element included in a Design element has its internal forces transformed in this manner. This is illustrated by the following equations:

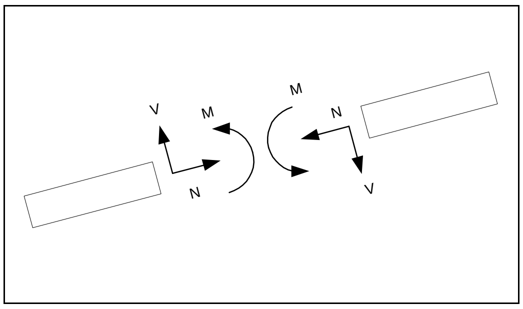


Figure 6-7 Internal forces

$$\left\{F_d\right\} = \left\{F_{d'}\right\} \cdot \text{dotK}$$

with

$$\text{dotK} = \begin{bmatrix} K_{d_{11}} \\ K_{d_{21}} \\ K_{d_{31}} \end{bmatrix}^T \begin{bmatrix} K_{e_{11}} \\ K_{e_{21}} \\ K_{e_{31}} \end{bmatrix}$$

where

{F_d} = the internal force vector of the Design Element.

{$F_{d'}$} = the untransformed internal force vector of the Design Element.

[K_e] = the coordinate matrix of the finite element.

[K_d] = the coordinate matrix of the Design Element.

dotK = the dot product between the local x – axis of the Design Element and a particular included finite element

As a result, a complete set of internal force data is mapped to the Design Element allowing for the determination of an internal force at any point along the length of the Design Element.
The transformation of displacements and rotations from the included finite elements to the Design Element is as follows.

$$\{d_d\} = \left([K_d]^T \cdot [K_e] \cdot \{d_e\}\right)$$

where

$\{d_d\}$ = the displacement vector of the Design Element

$[K_d]$ = the coordinate matrix of the Design Element

$[K_e]$ = the coordinate matrix of a finite element

$\{d_e\}$ = the displacement vector of a finite element

The transformation of the displacement vector of the finite elements is similar to that of the internal force transformation, except for the absence of the dotK factor. This is due to the fact that the displacements do not differ in sign on either a positive or negative face of an element.

As a result, all the information calculated by a finite element is passed on to the containing Design Element, while maintaining sign convention and direction. By creating Design Elements and a separate design model, a finite element model does not have to conduct the design procedures with limited and often envisaged design information.

Specializations: The Design Element discussed is a general representation of a structural member. These elements can represent a variety of structural members, e.g. steel, concrete, timber etc provided they provide the sufficient information required in the design process. A specialized Design Element, namely a Structural Steel Design Element is developed and discussed in the next section

6.2 Structural Steel Design Elements

Definition: A Structural Steel Design Element is a specialized Design Element that is used for the design of steel members. A Structural Steel Design Element provides additional functionality for design as implemented by *SANS 10162 Part1: Limit states design of hot rolled steelwork*. These elements have the same standard functionality of Design Elements with additional functionality for the design of steel members. These specialized steel elements are referred to as SSDesign Elements for the remainder of this chapter. The additional functionality is described as follows.

Design Type: Design types are representative types of steel members that govern the procedures used to perform a design as stipulated in *SANS 10162 Part1-The structural use of steel*. A steel member is assigned a certain design type according to the type or types of internal forces present in the member. These internal forces are the bending moments, shear forces and axial forces (compressive or tensile). Each SSDesign Element is assigned a certain design type that best represents the status of internal forces present. By assigning a particular design type to an SSDesign Element, more control is placed over the implemented design procedures. This allows for control over which design procedures from *SANS 10162 Part1-The structural use of steel* are executed on a steel member.

The various design types and their descriptions are as follows:

- BEAM

  This design type represents a steel member that has only bending moments and shear forces present internally. The steel member is designed mainly according to bending resistance.
- COLUMN

  This design type represents a steel member that has only a compressive axial force present within the member. The steel member is designed mainly according to compressive resistance.
- BEAM COLUMN

  This design type represents a steel member that has combined bending with compressive axial forces present within the member. The steel member is design mainly according to combined bending and compressive resistance.
- CANTILEVER

  This design type represents a steel member that has only bending moments and shear forces present internally. This member has one of its ends free from any vertical restraint. The steel member is designed mainly according to bending resistance.
- TENSION

  This design type represents a steel member that has only axial tensile forces present within the member. The steel member is designed mainly according to tensile resistance.
- TENSION BEAM

  This design type represents a steel member that has combined bending with axial tensile forces present within the member. The steel member is design mainly according to combined bending and tensile resistance.

The designer has the choice to assign one of the above design types to a particular steel member. These design types do not affect the composition of the SSDesign Element, but instead just determine the design procedures used.

Rotation: An SSDesign Element has the additional ability to rotate about it's local x – axis. This rotation angle, θ, orientates the cross section of the steel profile that is used to represent the SSDesign Element. This allows for the steel member to be rotated relative to the forces present in the member. As can be seen in Figure 6-8 (a), the steel profile used to represent a particular SSDesign Element has a rotation angle, θ, equal to 0. By rotating an SSDesign Element about it's local x – axis the chosen steel profile can be orientated accordingly to this angle. This is shown in Figure 6-8 (b) with θ equal to 90^o.

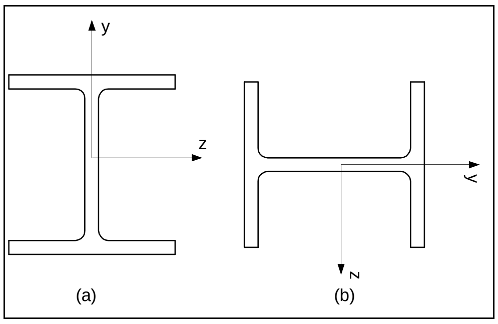


Figure 6-8 Rotation of elements

This rotation ranges from:

$0^o \leq \theta \leq 360^o$

where .

$\theta = 90 \cdot n$ with $n = \{0,1,2,3,4\}$

The steel member is rotated relative to the loading present, i.e. the external loading and thus the internal loading remains stationary relative to the steel member. In effect, only the SSDesign Element (steel member) is rotated.

In order to rotate an SSDesign Element, a rotation matrix is developed based on the rotation angle θ. The rotation matrix is multiplied by the transpose of the original coordinate matrix of the SSDesign Element. This procedure then creates a new coordinate matrix for the rotated SSDesign Element. This is shown as follows:

$$R = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & \sin\theta \\ 0 & -\sin\theta & \cos\theta \end{pmatrix}$$

$$[K_{new}] = [R] \cdot [K_{old}]^T$$

where

θ = the rotation angle about the local x – axis of the member.

K_{new} = the new coordinate matrix of the element.

R = the rotation matrix of the element

K_{old} = the old rotation matrix of the element.

Once this new coordinate matrix is calculated, the internal forces of the SSDesign Element are orientated about their new axes accordingly.
By providing this functionality, greater control is placed within the design model that is not provided by the finite element model. Complete control over cross section orientation relative to loading orientation is now provided.

Internal Restraints: Internal restraints can be viewed as a lateral support on a steel member. Internal restraints can be applied at points along the length of an SSDesign Element, providing lateral support for the steel member at those points. These lateral supports can be applied in any number along the length of an SSDesign Element. They are not present at the ends of an SSDesign Element. Figure 6-9 illustrates an SSDesign Element with two internal lateral restraints present along its length.

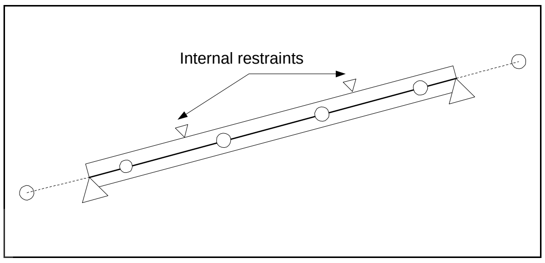


Figure 6-9 SSDesign Element with internal restraints

The concept of an internal support allows for the functionality to reduce buckling lengths of steel members as needed or to investigate the effects of adequate cross bracing within a structure. Internal restraints are ridged. They only provide information to the design model, and thus the design process, that a lateral support is present at a particular point. These restraints always provide adequate lateral support to a steel member as their resistance to lateral buckling is always sufficient.

Internal restraints provide support along either the weak and or strong axis of a steel member. Strong axis internal restraints reduce the strong axis Euler buckling length of a member. Weak axis internal restraints reduce either the weak axis Euler buckling length or lateral torsional buckling length of a member.

Apart from the positions along the length of a steel member, an internal restraint has a cross sectional position in a member. These restraints can be applied to the top or bottom flanges of a steel member or on both flanges simultaneously. Figure 6-10 illustrates the different positions of an internal restraint on a steel cross section.

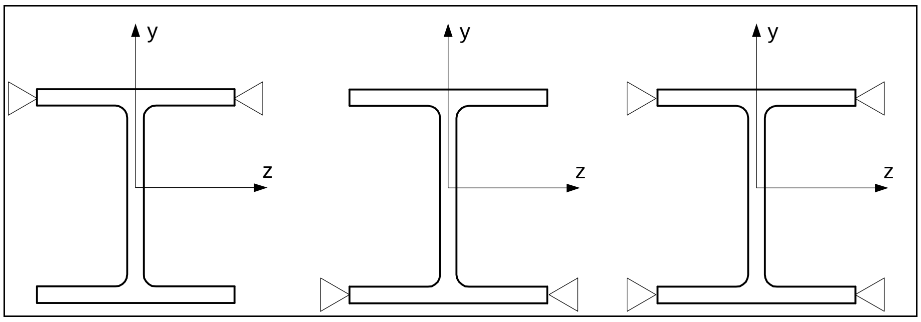


Figure 6-10 The various positions of an internal restraint

The top or bottom flange of a steel member is determined by the position of the member's cross section relative to the local y – axis of the member. The purpose of placing an internal restraint on either of the flanges of a steel member stems from the implemented design stipulations that lateral restraint is only effective on the compressive flange of a profile for determination of the lateral torsional buckling resistance of a member. Lateral restraints placed on non compressive flanges have thus no effect on the lateral torsional buckling lengths of the SSDesign Element and are ignored.

Apart from placing individual internal restraints the entire length of a member can be braced against lateral buckling. An SSDesign Element provides the ability to continuously laterally brace its entire length against weak axis buckling. As a result the steel members represented are designed accordingly to the bracing present.

Loading: The condition of loading on an SSDesign Element has the ability to be changed depending on the current design parameters. The load on an SSDesign Element can take on two distinct conditions. The loading condition can be toggled between a condition of stabilized loading or destabilized loading. The availability of this load state information is vital for effective implementation of the design process. These load states apply to flexural members, i.e. members that are subjected to bending.

For the case of a flexural member supported at both ends, a destabilising loading condition applies when the load is applied to the compression flange of the steel beam and both the load and flange are free to move vertically. For the case of cantilevers, a destabilising loading condition applies when the load is applied to the tension flange of the beam and both the load and the flange are free to move laterally. These conditions applied to the loading do not affect the status of the analysed structure in any way. A finite element model does not convey the information as to how a load is applied to a member. An SSDesign Element, by utilizing the information provided by the finite elements, attaches this additional loading information to create a more comprehensive element for use in the design process.

Implementation aspects: As discussed earlier, the additional functionality of specifying a design type for a SSDesign Element is critical for the correct design procedure to be executed. Although these design types are dependent on internal force type and presence, the procedure of design type association is not automated. In the pilot implementation, the user has to stipulate manually a design type of an SSDesign Element. This leaves the choice of design procedure to the user and not the implementing framework. By alleviating this functionality from the pilot framework, a more conservative approach is taken in the design process.
The choice of loading conditions on an SSDesign Element, whether it be stabilised or destabilised, is not automated by the pilot implementation due to this information not being available from the finite element model. This choice is given to the user to apply the loading conditions accordingly.

6.2.1 Structural Steel Restraints

In order for the correct calculation of effective lengths of structural steel members, along with various other design factors, according to *SANS 10162: Part1* the concept of structural steel restraints was developed. These restraints are unlike the restraints used in the finite element analysis of a structure. Finite element restraints are synonymous with the fixing of certain finite element degrees of freedom, such as translation and rotation of the nodes. The structural steel restraints developed for use with SSDesign Elements provide information about the degree of fixity of the end points of an SSDesign Element, as viewed from a steel design perspective. Structural steel restraints are passive in a sense that the do not affect the behaviour of a member under loading. They merely provide conditions that prescribe certain stages in the design process.

Each SSDesign Element has one structural steel restraint at its ends. Depending on the design type of the SSDesign Element, these restraints have different constraints that can be applied. The constraints available to a restraint are dependent on specific design code criteria, in this case *SANS 10162: Part1.* The various types of restraints and their constraints that can be applied to a design element due to the design type are as follows:

Flexural members (supported at both ends): The constraints in this category can be applied to either end of an SSDesign Element. The design type(s) that implement such conditions are the BEAM and BEAM – COLUMN design type.

All the constraint options refer to the degree of lateral restraint of the compression flange of the steel member and are taken from *SANS 10162 Part1: 10.2.1 Table 1.* They are as follows:

- Unrestrained (i.e. free to rotate in plan)
- Partially restrained (i.e. positive connection by flange cleats or end plates)
- Practically fixed (i.e. not free to rotate in plan)

One further constraint is one of torsional support, which can be applied by the user if necessary. This constraint is can be applied in conjunction to the other constraints.

Cantilevers: The constraints in this category depend on the ends of an SSDesign Element. The design type(s) that implement such conditions is the CANTILEVER design type.
These constraints are taken from *SANS 10162 Part 1: Table 2.*
The constraint conditions available at the support or start of an SSDesign Element are as follows:

- Built in laterally and torsionally
- Continuous, with lateral and torsional support
- Continuous, with lateral support only

The constraint conditions available at the end or tip of an SSDesign Element are as follows:

- Free
- Lateral restraint only (at compression flange)
- Torsional restraint only
- Lateral and torsional restraint

Axial compression (pure columns in trusses): The constraints in this category can be applied to either end of an SSDesign Element. The design type(s) that implement such conditions is the COLUMN design type.
These values are tabulated in *SANS 10162 Part1: Annex E Figure E.1.*

- Pinned
- Fixed
- Free
- Roller

By creating additional restraint conditions other than the mathematical restraints provided by the included finite elements more control is placed over the design conditions of a member.

6.2.2 Internal Elements

Definition: An internal element is simply a portion of an SSDesign Element. Internal elements are identical to SSDesign Elements in terms of functionality and design use. An internal element is created from an SSDesign Element. There are two distinct types of internal elements that are spawned by an SSDesign Element. They are:

- Weak axis internal elements
- Strong axis internal elements

These internal elements are designed in the same manner as the SSDesign Element that created them. The purpose of internal element is to create control over design lengths as well as effective lengths of steel members. As is often the case in effective design, strength of a member about its strong and weak axis, or a combination thereof, is required for determination of member adequacy. The strong axis of an element refers to the local x – y plane of the element while the weak axis refers to the local x – z plane of the element. Figure 6-11 illustrates the two buckling planes of an SSDesign Element. The internal elements of an SSDesign Element contains and controls all the information regarding the strength of a member about the strong and weak axes. These elements represent the structure of an SSDesign Element at a weak and strong axis level. Each internal element represents an unbraced portion of the parent element's strong and weak axis buckling length. Every SSDesign Element has one strong axis internal element and at least one weak axis internal element. All design calculations referring to the strong axis length of an SSDesign Element are executed on the element's strong axis internal element. All design calculations referring to the unbraced weak axis length of an SSDesign Element are executed on the element's weak axis internal element(s). Internal elements obtain all the necessary data, such as internal forces etc., from the parent SSDesign Element which in turn obtains the data from the included finite elements. Internal elements span between points of internal restraint present along the length of an SSDesign Element. For this purpose, the restraints present at the ends of an SSDesign Element are viewed as internal restraints as well.

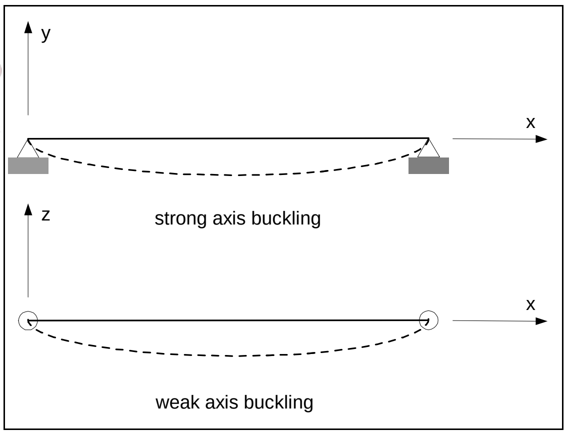


Figure 6-11 Buckling axes of a member

Strong axis internal elements: Every SSDesign Element has a single strong axis internal element. This element is identical to the parent SSDesign Element and spans from one end of the SSDesign Element to the other. This element represents an SSDesign Element's strong axis. An SSDesign Element's strong internal element is illustrated in Figure 6-12. This internal element spans the entire length of the SSDesign Element. The strength of the strong axis of an SSDesign Element is calculated from the attributes of its strong axis internal element.

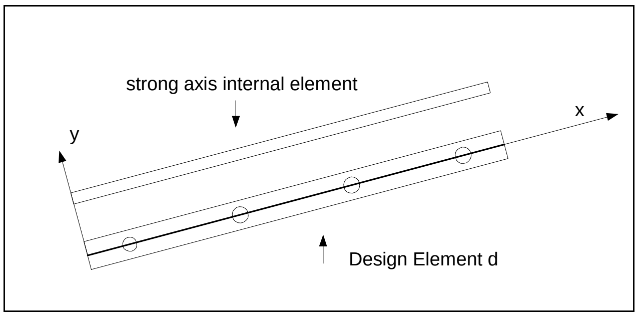


Figure 6-12 A strong internal element of an SSDesign Element

Weak axis internal elements: Weak axis internal elements represent the unbraced portions of an SSDesign Element's weak axis. In the implemented design procedures for steel members (SSDesign Elements), members that are provided with effective lateral support at the compression flange at intervals along the span, the length of the member shall be taken as the distance, centre to centre, between the lateral supports. Each portion of a member is designed as a separate member with all relevant design data taken from the specific portion. In order to maintain correct record for the various effective lengths or portions of members that are to be tested in the design process, weak axis internal elements are created. Weak axis internal elements represent the effective lengths (portions) of a member that are created due to lateral support at points along the span.

Weak axis internal elements are created between points of lateral restraint along the length of an SSDesign Element. In the case where no internal restraint is present or applicable, these internal elements span the entire length of the SSDesign Element. Each internal element has its own set of loading data that is relevant to that particular segment (portion) of an SSDesign Element. Each internal element can be

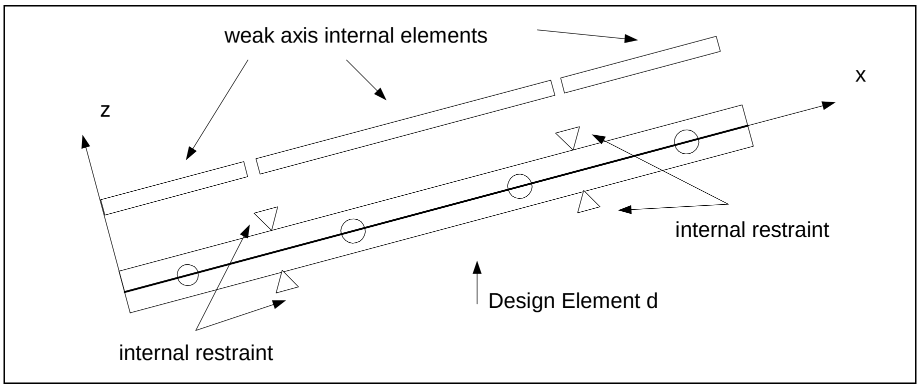


Figure 6-13 Weak axis internal elements of an SSDesign Element

viewed as a structural steel member on its own. As can be seen in Figure 6-13, an SSDesign Element with two internal restraints creates three internal elements. The adequacy of an SSDesign Element is about its weak axis is determined by the member strength of these internal elements.

Implementation aspects: In the prototype application, the design procedures are performed on the internal elements that an SSDesign Element produces, and not on the SSDesign Element itself. An SSDesign Element is used as the connection between the finite elements and the member it represents. All internal elements are created automatically in the prototype application by an SSDesign Element. If all the internal elements of an SSDesign Element are deemed adequate by the design process, the SSDesign Element as a whole discrete member is taken as adequate.

7 Design Set

In this section the concept of a Design Set is introduced. Their function in design in general as well as in the prototype application is described.

7.1 Concept of Design Sets

General: The concept of a design set was developed to provide uniformity of chosen profile types for common, repetitive member geometry of a structure. By dividing a structure into representative parts, members with similar functions in a structure can be designed as a single entity. These representative parts of a structure can be viewed as structural component sets as they from sets of common structural members. Each structural component set is designed with a particular profile type. For example, all members that are used as columns in a particular structure can be grouped together to form a structural component set of columns of a structure. This set of members can be designed together with a particular profile type preferred by the designer for use in columns. These "structural component sets" are called Design Sets.

Design Sets: Design Elements that represent particular members with similar functionality and layout within a structure can be grouped together to form Design Sets. A Design Set allows for each part of a structure to be designed separately from the whole structure. The mathematical representation of a Design Set is as follows:

$$\text{Design Set} \qquad S = \{\text{Design Element E} \mid \text{E is an element of a structural component set}\}$$
$$= \{E_1, E_2, E_3, E_4, \ldots\ldots, En\}$$

Each Design Set and its components, namely Design Elements, are designed with a particular profile type relevant to that part of the structure which the Design Set represents. Each of the Design Elements present in the Design Set is designed accordingly with the particular profile type of the Design Set. Design Sets allow for only the relevant parts of a structure to be designed with a particular profile, with other parts of a structure to be designed with other suitable profiles. This creates a more cost effective design procedure by eliminating redundant design on members with profiles that are better suited to other parts of a structure. Design Sets allow for more control over the types of profiles and their association with particular members in a design. Design Sets are independent on the type of member, i.e. steel, concrete, timber etc, as they are independent on the design procedures implemented.

Implementation aspects: In the prototype application, each Design Set is assigned a set of profiles by the designer. The members included in a Design Set are designed accordingly to these profile sets. Each profile set contains one particular type of profile; however profile types between a number of Design Sets may differ. This creates the possibility of multiple profile types present in a design.

An example of two Design Sets is illustrated in Figure 7-1. As can be seen from the figure, two Design Sets, namely Beams (A) and Columns (B), group together parts of a structure with similar purpose. Design Set A can be assigned a set of H-profiles to which a design can be performed while Design Set B can be assigned a set of Channel profiles. This is an example of multiple profile types being available in the application.

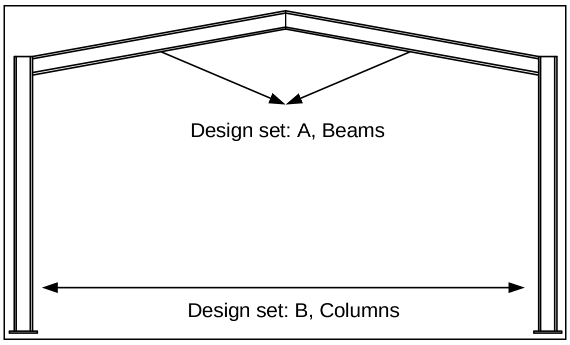


Figure 7-1 Design Sets

Each Design Element created in the design model is assigned to a particular Design Set depending on the preference of the user. All the Design Elements within a particular Design Set are designed accordingly with the profile type chosen for the enclosing Design Set. Once the design procedures are complete, each Design Element in a Design Set has a set of adequate profiles taken from the profile set of the Design Set. This is represented mathematically as follows:

Profile set of Design Element i (PS_{Ei}):

$$PS_{Ei} = \{p \in \text{ set of profile types AND p is adequate}\}$$

$$p = \text{ a profile}$$

To create uniformity within a Design Set, the intersection of all the adequate profile sets of the included Design Elements is calculated and passed on as the overall adequate profile set of the entire Design Set. This is represented as follows:

Profile set of Design Set s (PS_s)

$$PS_s = PS_{E1} \cap PS_{E2} \cap PS_{E3} \cap .... \cap PS_{En}$$

where

PS_{En} = the adequate profile set of the nth Design Element

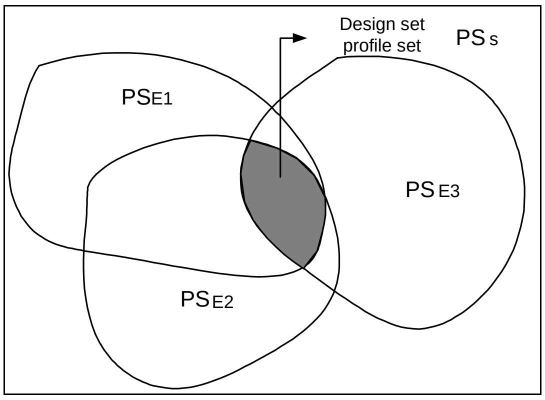


Figure 7-2 Design Set consisting of 3 Design Elements

Figure 7-2 illustrates the intersection of the profile sets of 3 Design Elements in a Design Set. The shaded area represents the intersection of the adequate profile sets of all Design Elements.

By creating the intersection of the profile sets of each Design Element, an overall adequate profile set is created, i.e. this set of profiles that are adequate for all the members in a particular Design Set.

8 The Development and Implementation of the Computational Framework

An object oriented framework for the design of structural steel members is developed and implemented for all types of prescribed steel members according to *SANS 10162: The structural use of steel Part1: Limit states design of hot rolled steelwork – 2005.*

The scope of the design framework does not include the structural analysis of the building under design consideration. A structural analysis framework has been developed in a separate study, as was a framework for steel connection design. The seamless integration of the various frameworks is the scope of future study. All the frameworks have a basic structure that makes them fundamentally compatible. For the development of the design framework it was decided to transfer analysis results and data in a useful format which implements an interface specified by the design framework. The analysis results are contained in .model files. These files contain information about the analysed structure that is to be designed. These .model files are used by the framework to both display and obtain all the relevant analysed results of the finite element models. The details of these model files are shown in Appendix A.

A further requirement for the framework is access to the range of hot rolled steel profiles and their cross sectional properties. This is achieved through the use of an external database that stores all the relevant properties about the available profile types and is read during the relevant part of execution of the framework. The external database used is a Microsoft Access database. This database contains the following tables:

- `Angul`

  This table contains all the necessary properties of unequal leg angle profiles
- `Angel`

  This table contains all the necessary properties of equal leg angle profiles
- `Chanpf`

  This table contains all the necessary properties of parallel flange channel profiles
- `Chantf`

  This table contains all the necessary properties of tapered flange channel profiles
- `CircHS`

  This table contains all the necessary properties of circular hollow sections

- Ø `Hsecpf`

  This table contains all the necessary properties of parallel flange H – profiles.
- Ø `Isecpf`

  This table contains all the necessary properties of parallel flange I – profiles.
- Ø `Isectf`

  This table contains all the necessary properties of tapered flange I – profiles.
- Ø `RecHS`

  This table contains all the necessary properties of rectangular hollow profiles.
- Ø `SqHS`

  This table contains all the necessary properties of square hollow profiles.
- Ø `TfrmH`

  This table contains all the necessary properties of T – profiles that were cut from H – profiles.
- Ø `TfrmI`

  This table contains all the necessary properties of T – profiles that were cut from I – profiles.
- Ø `SteelGrades`

  This table contains all the necessary properties of available types of steel.

The complete tables of the database are shown in Appendix B.

The remainder of this chapter will briefly discuss the member design model and the different packages that provide for the integration of the design process to the design model. This includes packages such as interface, component, service, model, gui and gui3D.

8.1 Interfaces

Interfaces are used to define the required functionality of objects in the framework. By implementing the interfaces, the various classes provide the functionality defined by the interfaces. A class represents an abstraction of a real object e.g. a steel profile.

Advantages of using interfaces: Instances of all classes that implement a specific interface are equivalent at the functional level defined by that interface. An interface is consequently a mechanism for establishing an equivalence relation in an object model, which has the advantage that the elements of the relation can be dealt with in an identical way. For example, when using different steel profiles, all steel profiles implementing the `ISteelProfile` interface will be able to provide its first moment of area when asked by the `Profile` object.

8.1.1 Interface Hierarchy

Figure 8-1 shows the interface hierarchy that was developed for the steel member design framework. The remainder of this chapter will briefly discuss the details of each interface. In some cases only the more important methods of the interfaces will be described. Methods required for internal functionality are not discussed here.

A more detailed description of all the classes and interfaces is described in Java Documentation of the application.

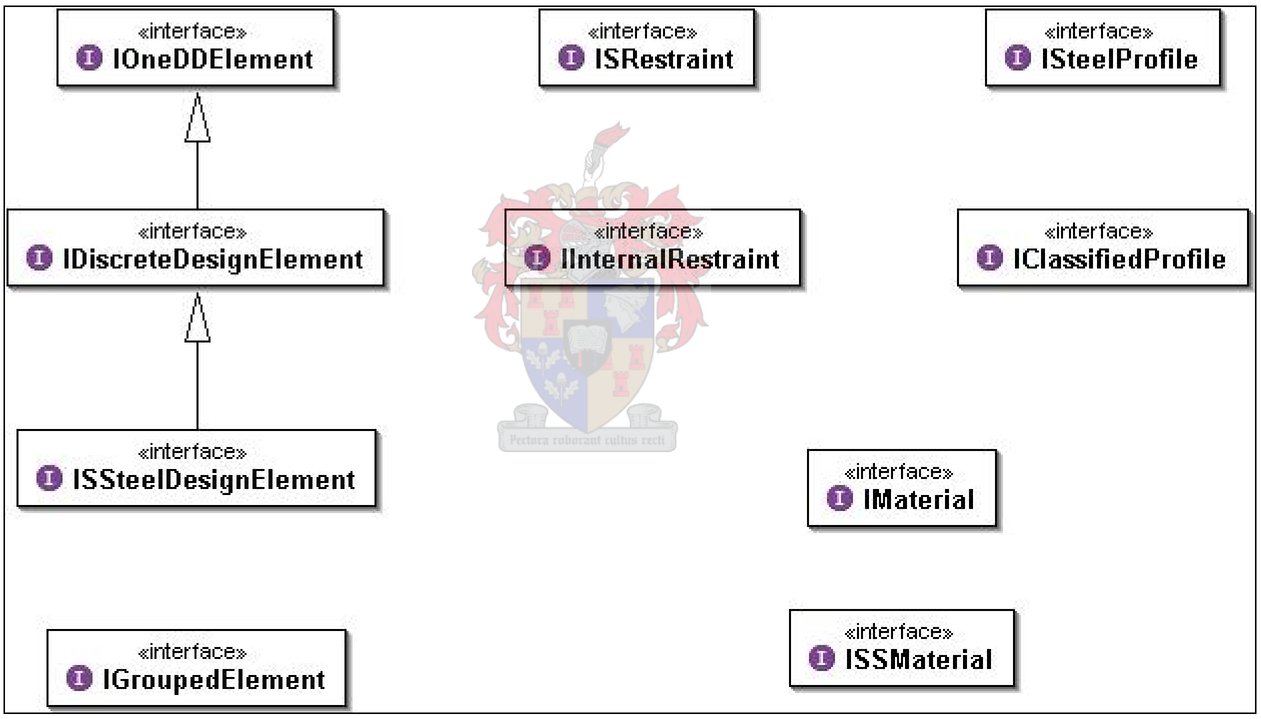


Figure 8-1 Interface hierarchy

8.1.2 Interface Descriptions

General design elements: All objects that implement the following interfaces form general design elements. These elements are not specifically aimed at any design procedures or structural type. These elements can represent any type of structural member of an entire structure, e.g. concrete, timber, steel etc. These interfaces provide the general functionality for all structural members that are to be used in a design model. Such general functionality include the orientation of an element in

2 dimensional space, the acquisition of internal forces along the length of an element, information on external loading, positions of any extreme forces within an element and physical properties of an element. This functionality provides the basis for any Design Element as discussed in chapter 6.
Two main interfaces prescribe the functionality of general Design Elements. The `IOneDElement` prescribes the orientation of a Design Element. This interface provides all the functionality to position a Design Element in a two dimensional plane. The second interface, namely the `IDiscreteDesignElement`, provides all the general functionality for the calculation of internal forces and the acquisition of external loading information from the included finite elements of the Design Element. A third interface, namely `IGroupededElement`, exists to provide the functionality for a particular Design Element to belong to a Design Set. The concept of a Design Set is discussed in chapter 7.
A brief description of these interfaces and their methods is given below.

8.1.2.1 `IOneDDElement`: Orientation of an element

The `IOneDDElement` interface prescribes the functionality of general `DesignElement` objects. An `IOneDDElement` object represents a general Design Element with all the necessary attributes for orientation in a two dimensional plane. Methods prescribed by this interface are:

- `public double length();`

  This method returns the length of the design element.
- `public double[][] getCoordinateSystem();`

  This method returns the coordinate system matrix of the design element as described in chapter 6.
- `public void setRotationAngle();`

  This method sets the rotation angle about the local x – axis of the design element. This rotation angle is obtained from the underlying finite elements.
- `public double getRotationAngle();`

  This method returns the rotation angle of the design element about its x – axis.
- `public void setRotationMatrix();`

  This method sets the complete rotation matrix of the design element
- `public double[][] getRotationMatrix(double[][] crds);`

  This method returns the complete rotation matrix of the design element.

8.1.2.2 `IDiscreteDesignElement:` Forces in an element

The `IDiscreteDesignElement` interface extends the `IOneDDElement` and therefore prescribes additional functionality for `DesignElement` and `InternalElement` objects. This additional functionality is to provide methods for acquiring the internal forces and external loading information of the discrete `DesignElement` or `InternalElement`. Methods prescribed by this interface are:

- `public double getInternalForce(byte type, double offset);`

  This method returns an internal force of a design element, depending on the type, e.g. shear force or bending moment that is required at a provided offset along the length of the element.
- `public void getInternalForceDistribution(byte type, double[] values);`

  This method sets the values of a given array to the values of the given type of internal forces along the length of a design element.
- `public double[] getExtremeInternalForces(byte type);`

  This method returns a double array of size 4. The first entry and third entry in the array is the position of the smallest and largest internal force of the given type respectively. The second and fourth entry of the array is the smallest and largest internal force values respectively.
- `public double[] getAbsExtremeInternalForces(byte type);`

  This method returns an array of length 4 of the absolute values of the largest and smallest internal forces of a given type. The first entry and third entry in the array is the position of the smallest and largest absolute internal force of the given type respectively. The second and fourth entry of the array is the absolute smallest and absolute largest internal force values of the given type respectively.
- `public double getMaxAxial();`

  This method returns the maximum axial force in a design element.
- `public double getMaxShearY();`

  This method returns the maximum shear force in the local y axis of a design element.
- `public double getMaxShearZ();`

  This method returns the maximum shear force in the local z axis of a design element.
- `public double getMaxMoment_zz();`

  This method returns the maximum bending moment about the local z axis of a design element.
- `public double getMaxMoment_yy();`

This method returns the maximum bending moment about the local y axis of a design element.

- `public double getMaxTorsion();`

  This method returns the maximum torsional moment of a design element.
- `public double[] getEndForces();`

  This method returns a vector of end forces of a design element.

8.1.2.3 `IGroupedElement`: Grouping elements in a Design Set

The `IGroupedElement` interface prescribes the functionality of `DesignElement` and `InternalElement` objects that are grouped together for particular design reasons. These objects contain all the possible methods required for design within a set. Methods prescribed by this interface are:

- `public void setDesignSet(DesignSet designSet);`

  This method adds the design element object to a specific `DesignSet` object. The concept of a Design Set is discussed in chapter 7.
- `public DesignSet getDesignSet();`

  This method returns the `DesignSet` object of a design element object.

Structural steel elements: The following interfaces all provide the additional functionality for structural steel design. All elements that implement the following interfaces have the necessary functionality to represent steel members in the design model along with the general functionality as described by the previous interfaces for general design elements. Additional counterparts for use in steel design along with the steel members as described in chapter 5 are developed by the following interfaces. These include the restraints of a member, internal elements, internal lateral restraints and any other design information that is required for steel design.

The `ISSteelDesignElement` interface provides all the additional functionality to design steel members. This interface specialises a general Design Element to form a steel design member, namely an SSDesign Element. This specialized design element is discussed in chapter 6.

The `ISRestraint` interface provides the functionality to create the end restraints of a steel member that is used to define the end conditions of an SSDesign Element. These steel restraints are discussed in chapter 6.

The `IInternalRestraint` defines the functionality of internal restraints that may be applied along the length of an SSDesign Element to form lateral support, as discussed in chapter 6.

A brief description of these interfaces and their methods are described next.

8.1.2.4 `ISSteelDesignElement:` Steel design properties of an element

The `ISSteelDesignElement` interface extends the `IDiscretDesignElement` and therefore prescribes additional functionality for `DesignElement` and `InternalElement` objects. Only important methods of this interface are described here.

- `public double getEffLengthCrx();`

  This method returns the effective length of a design element for in plane Euler buckling.
- `public double getEffLengthCry();`

  This method returns the effective length of a design element for out of plane Euler buckling.
- `public double getEffLengthMcr();`

  This method returns the effective length of a design element for lateral torsional buckling.
- `public double getK();`

  This method returns the effective length factor of a design element
- `public void setKCrx(double k);`

  This method sets the effective length factor of a design element for in plane Euler buckling.
- `public void setKCry(double k);`

  This method sets the effective length factor of a design element for out of plane Euler buckling.
- `public void setKMcr(double k);`

  This method sets the effective length factor of a design element for in lateral torsional buckling.
- `public void setStartRestraint(IRestraint r);`

  This method sets the start restraint object of a design element.
- `public void setEndRestraint(IRestraint r);`

  This method sets the end restraint object of a design element.
- `public Restraint[] getRestraints();`

  This method returns an array of the restraints of a design element.
- `public void setBracedStructure(boolean braced);`

  This method sets the status of a design element to whether it is part of a braced structure or not. The parameter is set to `true` if the structure is braced and `false` if not.
- `public boolean getBracedStructure();`

  This method returns the braced structure status of a design element.

- `public void setBraced(boolean braced);`

  This method sets the braced status of the design element. The braced status is set to `true` if the design element is continuously laterally braced along its length and `false` if not.
- `public boolean getBraced();`

  This method returns the braced status of the design element
- `public void setLoadStablized(boolean stabilized);`

  This method sets the load stabilized status of the design element. The load stabilized status is set to `true` if the load is stabilized and `false` otherwise.
- `public boolean getLoadStabilized();`

  This method returns the load stabilized status of the design element.

8.1.2.5 `ISRestraint` End restraints of a steel member

The `ISRestraint` interface prescribes the functionality of `Restraint` objects used in a steel member design. A `Restraint` object represents applicable restraint of a steel member and is applied at the end points of a member. Important methods prescribed by this interface are as follows:

Methods applicable to beam and beam – column design:

- `public void setSSRestAgainstTorsion(boolean val);`

  This method sets whether a beam or beam – column is restrained against torsion at its end supports or not. `True` if the beam/ beam – column is restrained against torsion and `false` if not. For simply supported beams or beam columns.
- `public void setUnRestrainedSupport(boolean val);`

  This method sets whether a beam or beam – column is free to rotate in plan at its ends or not (i.e. free to rotate about its local y – axis). `True` if the ends of the beam/ beam – column are free to rotate in plan and `false` if not.
- `public void setPartRestrainedSupport(boolean val);`

  This method sets whether a beam or beam – column is partially allowed to rotate in plane at its ends or not (i.e. partially free to rotate about its local y – axis). `True` if the ends of the beam/ beam – column are partially free to rotate in plane and `false` if not.
- `public void setPracFixedSupport(boolean val);`

  This method sets whether a beam or beam – column is practically built in at its ends or not (i.e. not free to rotate about its y – axis). `True` if the ends of the beam/ beam – column are practically fixed and `false` if not.

Methods applicable to cantilever beam design:

- `public void setBuiltInSupport(boolean val);`

  This method sets whether a cantilever beam has its supported end built in laterally and torsionally or not. `True` if the supported end of the cantilever is supported laterally and torsionally, and `false` if not. Figure 8-3 illustrates a simple example of this support.

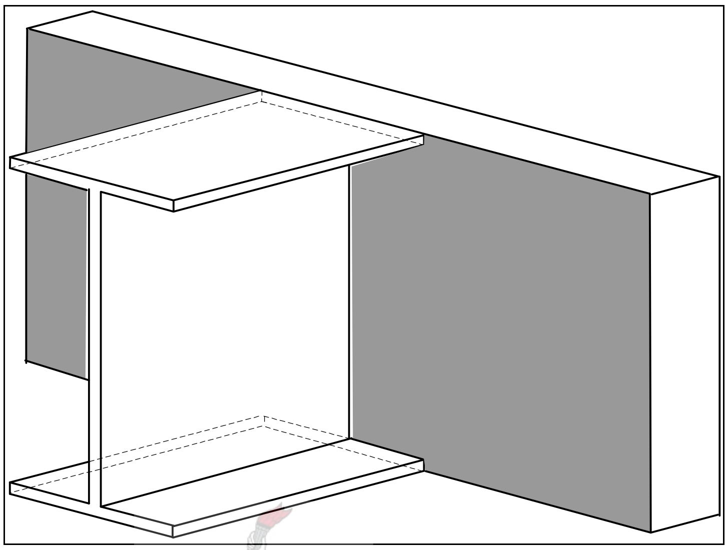


Figure 8-2 Built in support

- `public void setContTorsionalRestrained(boolean val);`

  This method sets whether a cantilever beam has its supported end supported against torsion and lateral movement while being continuous. `True` if the continuous supported end of the cantilever is supported laterally and torsionally, and `false` if not. Figure 8-3 illustrates a simple example of this support.

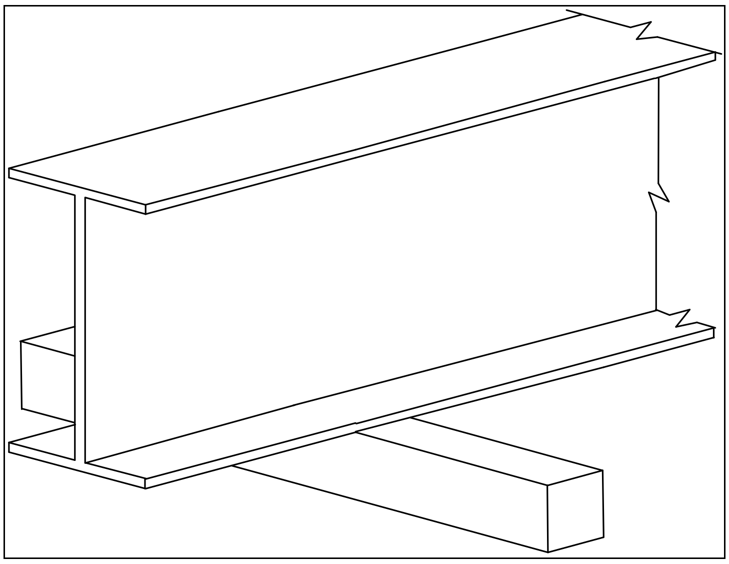


Figure 8-3 Continuous support

- `public void setContLateralRestrained(boolean val);`

  This method sets whether a cantilever beam has its supported end supported against lateral movement only while being continuous. `True` if the continuous supported end of the cantilever is supported only laterally, and `false` if not.
- `public void setFreeEnd(boolean val);`

  This method sets whether a cantilever beam has its free end unsupported both laterally and torsionally or not. `True` if the free end of the cantilever is completely unsupported, and `false` if not.
- `public void setLateralRestraint(boolean val);`

  This method sets whether a cantilever beam has its free end supported against lateral movement or not. `True` if the free end of the cantilever is laterally restrained, and `false` if not.
- `public void setTorsionalRestraint(boolean val);`

  This method sets whether a cantilever beam has its free end supported against torsion or not. `True` if the free end of the cantilever is torsionally restrained, and `false` if not.
- `public void setLatTorRestraint(boolean val);`

  This method sets whether a cantilever beam has its free end simultaneously supported against both torsion and lateral movement or not. `True` if the free end of the cantilever is both torsionally and laterally restrained, and `false` if not.

Methods applicable to pure column design:

- `public void setFixed(boolean val);`

  This method sets whether a column has its end restrained against translation and rotation. `True` if the end of the column is fixed against rotation and translation and `false` if not.
- `public void setPinned(boolean val);`

  This method sets whether a column has its end restrained against translation only or not. `True` if the end of the column is restrained against translation only and `false` if not.
- `public void setRoller(boolean val);`

  This method sets whether a column has its end restrained against rotation but is free to translate. `True` if the end of the column is restrained against rotation while being free to rotate and `false` if not.
- `public void setFree(boolean val);`

This method sets whether a column has its end unrestrained. `True` if the end of the column is unrestrained and `false` if not.

8.1.2.6 `IInternalRestraint` Internal lateral support of a steel member

The `IInternalRestraint` interface prescribes the functionality of `InternalRestraint` objects used in the design process. An `InternalRestraint` object represents applicable internal restraint conditions of a steel member. Important methods prescribed by this interface are as follows:

- `public void setPositiveRestraint(boolean val);`

  This method sets whether a steel element is laterally restrained at a point on its positive flange. `True` if the steel element is laterally restrained at a point on its positive flange and `false` if not. A flange is positive if it lies on the positive local y – axis side of the steel profile.
- `public void setNegativeRestraint(boolean val);`

  This method sets whether a steel element is laterally restrained at a point on its negative flange. `True` if the steel element is laterally restrained at a point on its negative flange and `false` if not. A flange is negative if it lies on the negative local y – axis side of the steel profile.
- `public void setPosAndNeg(boolean val);`

  This method sets whether a steel element is laterally restrained at a point on both its positive and negative flanges. `True` if the steel element is laterally restrained at a point on both its positive and negative flange and `false` if not.
- `public void setStrong(boolean val);`

  This method sets whether an internal restraint is a strong axis internal restraint or not.
- `public void setWeak(boolean val);`

  This method sets whether an internal restraint is a weak axis internal restraint or not.

Steel Profiles: The interfaces `ISteelProfile` and `IClassifiedProfile` define the functionality of the steel profiles that are used to represent the members in the design model. The interface `ISteelProfle` provides the functionality to acquire the cross sectional and material properties of a hot rolled steel profile. These profiles determine the manner in which the design processes are executed on a steel member. The `IClassifiedProfile` interface provides the functionality to classify a steel profile according to the manner of attainment of the yield moment by the cross section. This classification is dependent on the type of compressive forces present in a member.

8.1.2.7 `ISteelProfile` Steel profile used for a particular member or group of members

The `ISteelProfile` interface prescribes certain functionality of steel `Profile` objects used in a design model. A `Profile` object the common attributes and methods that are needed for any profile element. Method descriptions used depend on the local axis system of a `Profile` object as shown in Figure 8-4. This axis system is similar to the one used in chapter 6. The only difference is that the local x and z axis are swapped for the profile's local axis system. This was done to maintain uniformity of the local axis systems between the `Profile` objects used in the application and the steel profiles defined in *SANS 10162 2005 Part1* and the *Southern African Steel Construction HandBook, fifth edition 2005*.

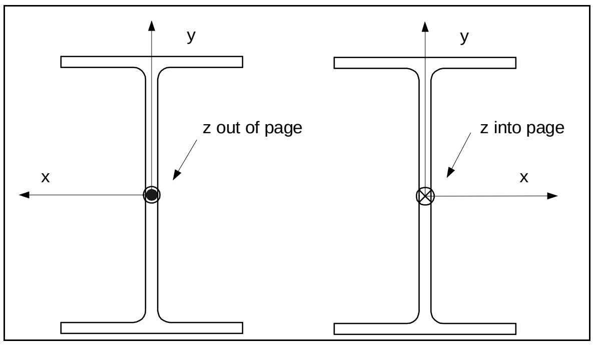


Figure 8-4 Local axis system

Important methods of this interface are as follows:

- `public double Ix();`

  This method returns the second moment of area about the local x – axis of the steel section.
- `public double Iy();`

  This method returns the second moment of area about the local y – axis the steel section.
- `public double rx();`

  This method returns the radius of gyration about the local x – axis of the steel profile,
- `public double ry();`

  This method returns the radius of gyration about the local y – axis of the steel profile,
- `public double getArea();`

  This method returns the gross cross sectional area of the steel profile.
- `public double getZplx();`

  This method returns the plastic section modulus of the steel section with respect to bending about the local x – axis.
- `public double getZply();`

  This method returns the plastic section modulus of the steel profile with respect to bending about the local y – axis.
- `public double getZex();`

This method returns the elastic section modulus of the steel profile with respect to bending about the local x – axis.

- `public double getZey();`

  This method returns the elastic section modulus of the steel profile with respect to the local y – axis.

- `public double[] getShearCentre();`

  This method returns the coordinates of the shear centre of the steel profile relative to the position of the centre of gravity of the profile's cross section.

- `public double getJ();`

  This method returns the value of the St. Venant torsional constant for the steel profile.

- `public double getCw();`

  This method returns the warping torsional constant of the steel profile. For structural hollow sections this value is 0.

- `public SSMaterial getMaterial();`

  This method returns the `Material` object that the steel profile consists of.

- `public SSMaterial setMaterial(SSMaterial material);`

  This method sets the `Material` object of a steel profile to the provided type.

8.1.2.8 `IClassifiedProfile` Steel profile classified according to the implemented design code

The `IClassifiedProfile` interface prescribes the functionality with respect to classification of `Profile` objects. Profile objects that implement this interface can be classified according to the classification system used in the design process.

Important methods prescribed by this interface are as follows:

- `public int getCompressionClass();`

  This method returns the compression class of the steel profile.

- `public int getFlexuralClass(double Cu);`

  This method returns the flexural or combined flexural and compression class of the steel profile.

Material: The interface `IMaterial` defines a general material object. This interface defines the general properties and functionality for a material object that is required to represent the composition of a Design Element. The `ISSMaterial` interface provides additional methods for the definition of hot rolled steel used for steel profiles.

8.1.2.9 `IMaterial` A general material that comprises a general structural member

The `IMaterial` interface prescribes the functionality of `Material` objects. A `Material` object represents a general material type with all the necessary attributes for design purposes. Important methods provided by this interface are as follows:

- `public double getE();`

  This method returns the value of Young's modulus for the particular material.
- `public double getG();`

  This method returns the value of the shear modulus for a particular material.
- `public double getNu();`

  This method returns the value of Poisson's ratio for a particular material.
- `public double getDensity();`

  This method returns the value of the density for a particular material.
- `public void setE(double E);`

  This method sets the value of Young's modulus.
- `public void setG(double G);`

  This method sets the value of the shear modulus.
- `public void setNu(double nu);`

  This method sets the value of Poisson's ratio.
- `public void setDensity(double density);`

  This method sets the value of the density.

8.1.2.10 `ISSMaterial` Structural steel material that comprises a steel member

The `ISSMaterial` interface prescribes the functionality of `SSMaterial` objects. An `SSMaterial` object represents a steel grade that is used in steel profiles with all the necessary attributes for design purposes. Important methods provided by this interface are as follows:

- `public double getFy();`

  This method returns the yield stress of the particular grade of steel.
- `public double getFu();`

  This method returns the ultimate strength (stress) of the particular grade of steel.
- `public void setFy(double fy);`

  This method sets the value of yield stress.
- `public void setFu(double fu);`

  This method sets the value of ultimate strength.

8.2 Components

The components of a structural steel member represent all the physical parts of the member as well as the influence of the surrounding steel structure. These include steel members, restraints, internal restraints, and the steel profiles of the members themselves. Some of these components are necessary for the creation of a specific steel element and some are not.

8.2.1 Component Hierarchy

General: Figure 8-5 shows the first part of the component hierarchy that was implemented for the framework. These components shown represent the physical objects that are used in the application of the design process. The remainder of this chapter will briefly describe the attribute and methods of each of these components. Due to the fact that some of the methods of these components pertain to the interface methods that they implement, which was already discussed in section 8.1, only methods that are not prescribed by the interfaces of each component will be briefly discussed. A more detailed description and explanation of each component is provided in the Java Documentation of the application.

Member components: The components represented in Figure 8-5 are physical objects that represent the various components used in structural steel design. The components represented are the SSDesign Elements (steel members) represented by class `SSDesignElement`, the end points of an SSDesign Element represented by class `SectorPoint`, the end restraints of a steel element represented by class `Restraint`, internal lateral internal support represented by class `InternalRestraint` and the internal elements of an SSDesign Element, the internal elements, represented by class `InternalElement`. These classes are responsible for the construction of the physical components in the design model. All the classes are required for the creation of a steel member apart from classes `Material`, `SSMaterial` and `InternalRestraint`. Objects of class `InternalRestraint` are only created when internal restraints are required at points along the length of a steel member. Objects of classes `Material` and `SSMaterial` are used to define the physical material properties of the various hot rolled steel profiles that provide physical form to the steel members or SSDesign Elements. No specific design procedures are implemented by these classes and any further functionality required from a design procedure can easily be implemented via the interfaces that prescribed the functionality of these components.
The hierarchy of these components allow for the possible inclusion of different design procedures on the components.

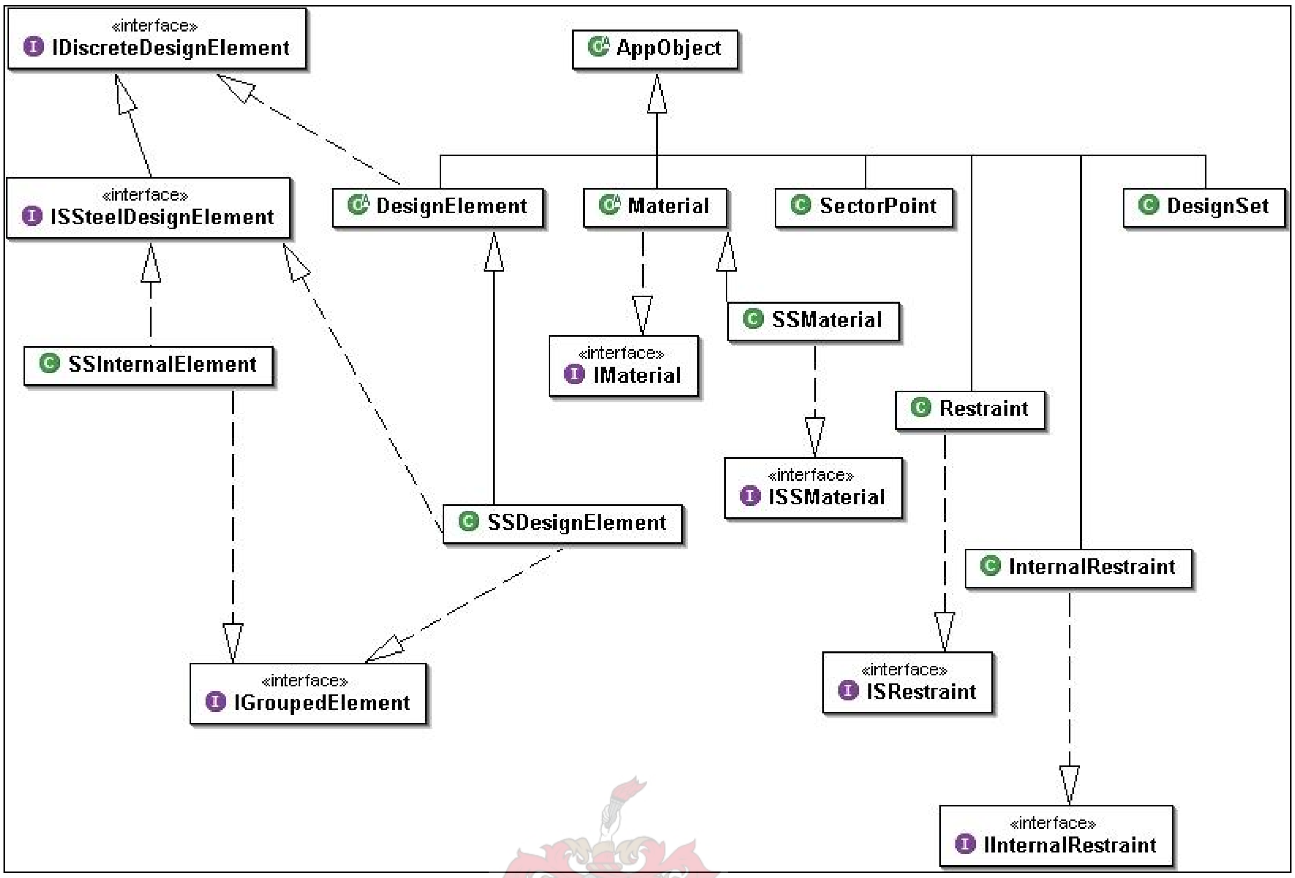


Figure 8-5 Component hierarchy (1)

Profiles: Figure 8-6 shows the second part of the component hierarchy that was implemented for the framework. These components represent a selection of hot rolled cold formed and structural hollow steel sections that were used in the application of the design process. Class `Profile` represents a physical general steel profile. Class `HRSteelProfile` represents a specific type of profile, namely a steel profile with all the necessary attributes and methods required to describe the physical properties of a steel profile.

The remainder of this chapter following the descriptions of the steel member components described in Figure 8-5 will briefly describe the attributes and methods of each of the steel profiles. Due to the fact that some of the methods of these components pertain to the interface methods that they implement, which was already discussed in section 8.1, only methods that are not prescribed by the interfaces of each component will be briefly discussed.
A more detailed description and explanation of each component is provided in the Java Documentation of the application.

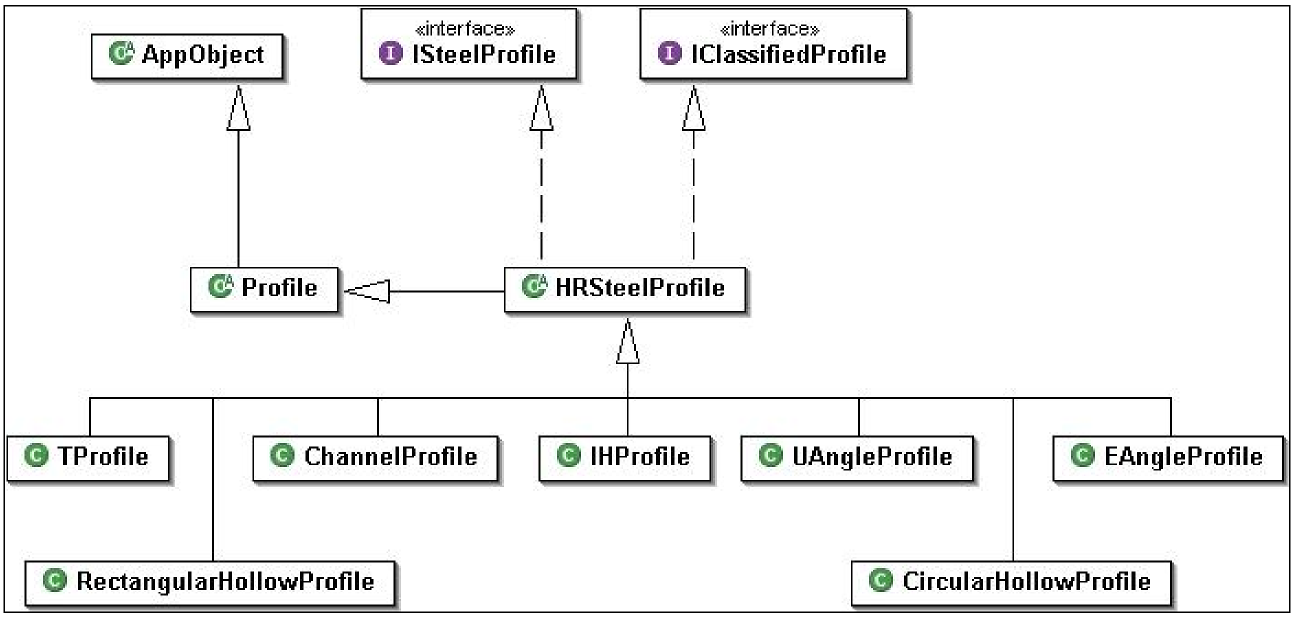


Figure 8-6 Component hierarchy (2)

The hierarchy of these components allows for the possible inclusion of different structural steel profile types.

8.2.2 Component descriptions

8.2.2.1 `DesignElement`

Class `DesignElement` implements the interface `IDiscreteDesignElement` and represents a general design element as discussed in chapter 6. This is illustrated in Figure 8-7.

`DesignElements` are the basic form of discrete structural elements that can be designed according to specific procedures depending on material type and type of design. These elements provide the basic ordered and structured transition from the discrete finite elements that are used in analysis of a structure and the design process. Design elements can be used to represent a number of different material designs, but only structural steel design elements are implemented here.

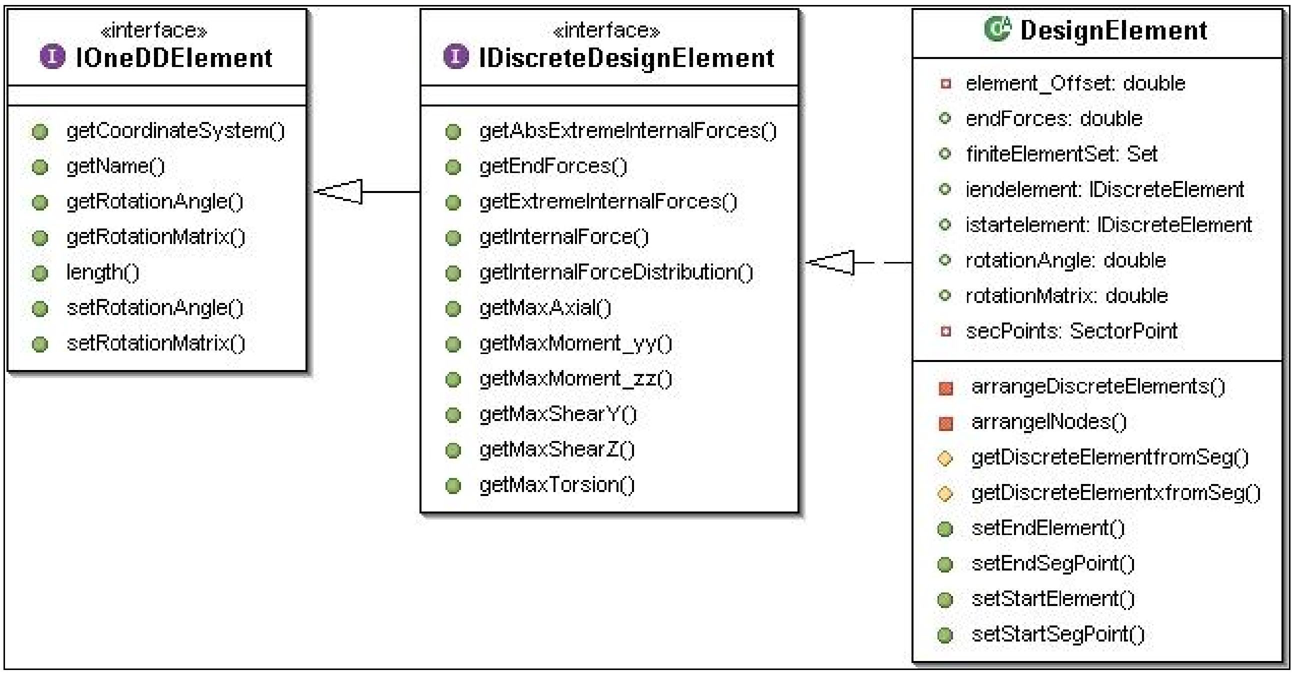


Figure 8-7 Design Element hierarchy

The attributes of an object of class `DesignElement` are:

- `public double[][] rotationMatrix;`

  The rotation matrix of the design element.

- `public double[] endForces;`

  The end forces of the design element.

- `public IDiscreteElement istartelement;`

  The finite element at the start of the design element.

- `public IDiscreteElement iendelement;`

  The finite element at the end of the design element.

- `public Set finiteElementSet;`

  The unordered set of finite elements comprising the design element.

- `private double[] element_Offset;`

  The offsets (if any) of the design element from the end nodes of the finite elements at the extremities. These offsets occur if the ends of a design element do not coincide with the ends of the finite elements at the extremities.

- `private SectorPoint[] secPoints;`

  The actual end points of a design element. They do not necessarily coincide with the ends of the included extreme finite elements

Additional methods implemented in class `DesignElement`:

- `public void setStartElement(IDiscreteElement iel)`

  This method sets the start finite element according to the chosen local x – axis of the design element.
- `public void setEndElement(IDiscreteElement iel)`

  This method sets the end finite element according to the chosen local x – axis of the design element.
- `private void arrangeDiscreteElements()`

  This method orders the included finite elements of the design element according to their sequence relative to the local x – axis of the design element.
- `private void arrangeINodes()`

  This method orders the nodes of the included finite elements according to their sequence relative to the local x – axis of the design element.
- `public void setStartSegPoint(SectorPoint p)`

  This method sets the start point of the design element for calculation of the local x – axis vector of the design element as well as for calculating the offsets (if any) of the design element.
- `public void setEndSegPoint(SectorPoint p)`

  This method sets the end point of the design element for calculation of the local x – axis vector of the design element as well as for calculating the offsets (if any) of the design element.
- `protected IDiscreteElement getDiscreteElementfromSeg(double x)`

  This method returns a finite element object corresponding to a given position along the local x – axis of the design element.
- `protected double getDiscreteElementxfromSeg(double x)`

  This method returns the local x – axis position of a corresponding finite element from a given position along the local x – axis of a design element.

8.2.2.2 SSDesignElement

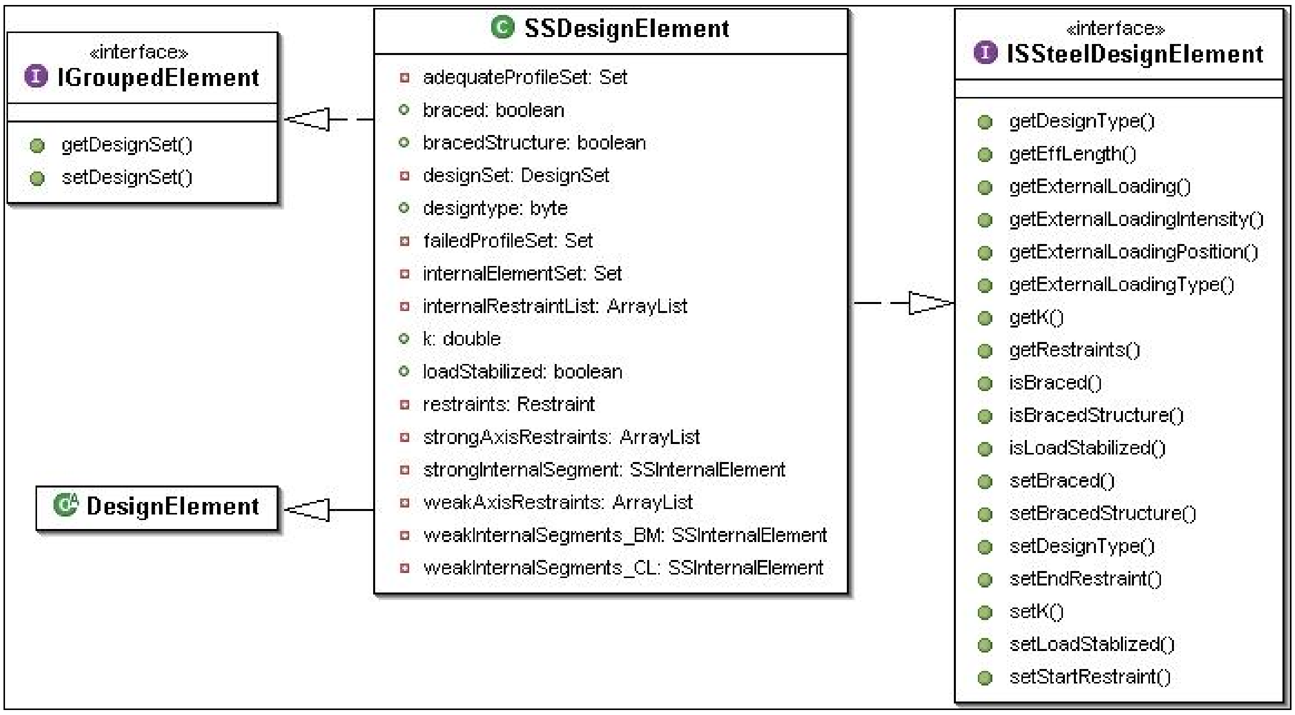


Figure 8-8 SSDesign Element hierarchy

Class `SSDesignElement` extends class `DesignElement` and implements `ISSteelDesignElement` as well as `IGroupedElement` and thus contains all the attributes and methods of an object of class `DesignElement`. This is illustrated in Figure 8-8. The separation is made for the reason of specializing in structural steel design. Structural steel design elements require certain information about itself that is different for other types of structural design. The calculation of effective lengths is one specialized function for structural steel elements. The effects of restraints as well as internal restraints on steel elements are different for other structural elements.

Objects of class `SSDesignElement` are commonly referred to in this thesis as **steel elements.**

The additional attributes of an object of class `SSDesignElement` are:

- `private Set adequateProfileSet;`

  The set of adequate steel sections for the structural element after a design is complete.

- `private Set failedProfileSet;`

  The set of failed steel sections for the structural element after a design is complete.

- `private Set internalElementSet;`

  The complete set of internal elements for the steel element. Internal Elements are discussed in chapter 6.

- `private SSInternalElement sInternalSegment;`

The strong axis internal element of the steel element which for the purposes of this framework is the steel element in its entirety.

- `private SSInternalElement[] weakInternalSegments_BM;`

  The array of weak axis internal elements that is applicable for the purposes of flexure design.

- `private SSInternalElement[] weakInternalSegments_CL;`

  The array of weak axis internal elements that is applicable for the purposes of axial compression or combined axial compression and flexure design.

- `private ArrayList internalRList;`

  The list of internal restraints for the steel element ordered according to their position along the length of the design element.

- `private ArrayList strongAxisRestraints;`

  The ordered (according to position along the local x – axis of the member) list of strong axis internal restraints for the steel elements.

- `private ArrayList weakAxisRestraints;`

  The ordered (according to position along the local x – axis of the member) list of weak axis internal restraints for the steel elements.

- `public byte designtype;`

  The design type of the steel element, e.g. BEAM, COLUMN etc.

- `public boolean braced;`

  The status of whether the steel element is continuously braced along its length or not.

- `private Restraint[] restraints;`

  The end restraints of the steel element.

- `private DesignSet designSet;`

  The design set to which the steel element belongs.

- `public boolean bracedStructure;`

  The status of whether the steel element is part of a structure that is braced or not.

- `public boolean loadStabilized;`

  The status of whether the steel element has its loading stabilized or not.

- `public double k;`

  The effective length factor of the steel element.

8.2.2.3 SSInternalElement

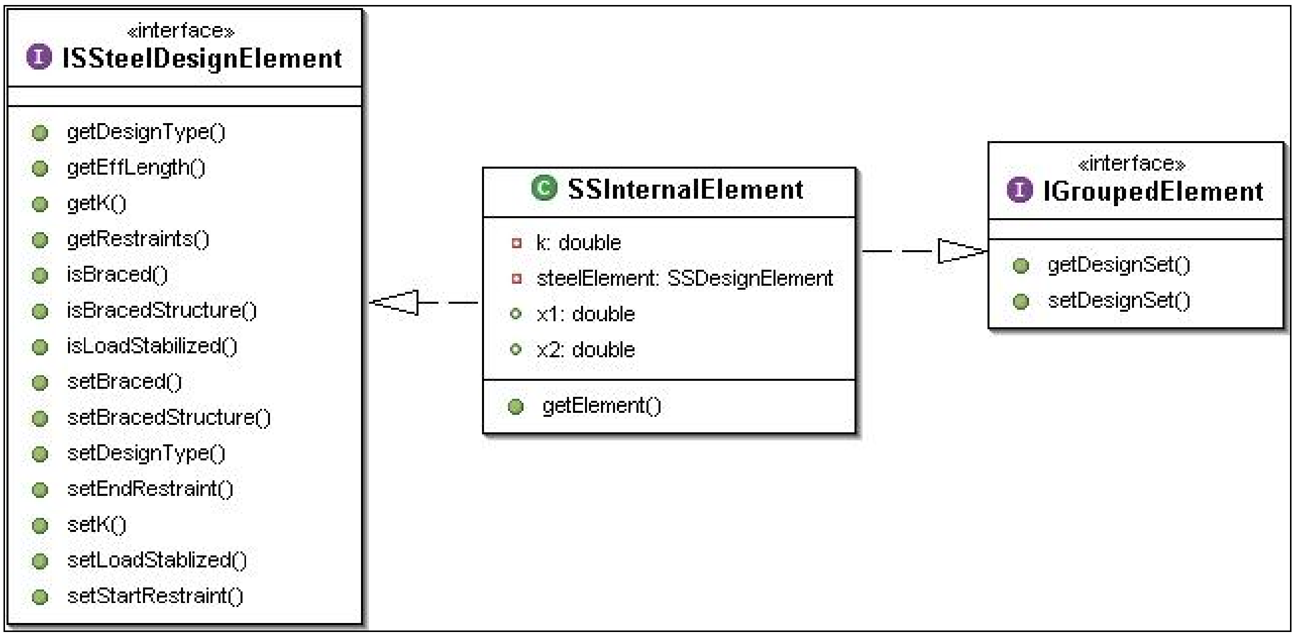


Figure 8-9 Internal Element hierarchy

Class `SSInternalElement` implements interfaces `ISSteelDesignElement` and `IGroupedElement` and thus has the all the functionality as prescribed by those interfaces. This is illustrated by the hierarchy represented in Figure 8-9. An internal element object represents a subdivision of a steel design element into smaller parts for design purposes. Internal elements have the same functionality as structural steel design elements (SSDesign Elements) and can be used in the same manner for design purposes, as they represent portions of the steel elements. The concept of an internal element was discussed in chapter 6. Special attributes of an object of class `SSInternalElement` are:

- `private SSDesignElement steelElement;`

  The parent steel element that generated the internal element object.
- `public double x1;`

  The start x – coordinate of the internal element.
- `public double x2;`

  The end x – coordinate of the internal element.
- `private double k;`

  The effective length factor of the internal element.
- `private double[] endforces;`

  The array of end forces for the internal element.

Additional methods implemented in class `SSInternalElement` are:

- `public ISSteelDesignElement getElement()`

This method returns the design steel element that spawned the internal element object, i.e., the parent steel design element.

- `public boolean isElement()`

  This method returns the status of whether the internal element spans the entire length of the parent design element

- `public boolean isInternalElement()`

  This method returns the status of whether the internal element does not occur at either of the ends of a design element and does not span the entire length of the design element

8.2.2.4 Restraint

Class `Restraint` implements the interface `ISteelRestraint` and represents a restraint element as discussed in chapter 6. Restraints contain the necessary information about the end conditions of a steel Design Element.

The attributes of an object of class `Restraint` are:

- `private boolean[] constraints;`

  The array of constraints of the restraint.

- `private byte type;`

  The design type of the restraint.

- `public boolean startCBR;`

  The indicator for a start restraint, applicable for cantilever beams only, as they have different end conditions.

- `public boolean endCBR;`

  The indicator for an end restraint, applicable for cantilever beams only, as they have different end conditions.

Additional methods not prescribed by the interface `ISteelRestraint`:

- `public boolean[] getConstraints()`

  This method returns all the toggled constraint values of the restraint object.

- `public void setConstraints(boolean[] constraintArray)`

  This method sets the constraint values of a restraint either on or off depending on the values specified from the given array. `True` if the constraint is on or `false` if the constraint is off.

- `public void setConstraintAt(int type, boolean val)`

  This method sets a specific constraint at a given position either on or off. `True` if the constraint is on and `false` if the constraint is off.

8.2.2.5 `InternalRestraint`

Class `InternalRestraint` implements the interface `IInternalRestraint` and thus has all the functionality as prescribed by the interface. An object of class `InternalRestraint` represents an internal restraint as discussed in chapter 6. These internal restraints are used in the steel design procedure and can be applied to any structural steel element in the application.

The attributes of an object of class `InternalRestraint` are:

- `public boolean topflange;`

  The status of whether the internal restraint is applied to the top flange of the steel section.

- `public boolean bottomflange;`

  The status of whether the internal restraint is applied to the bottom flange of the steel section.

- `public boolean weak;`

  The status of whether the internal restraint is applied to the weak axis of the steel section

- `public boolean both;`

  The status of whether the internal restraint is applied to both the bottom and top flanges of the steel section.

- `public boolean strong;`

  The status of whether the internal restraint is a strong axis internal restraint .

- `private double position;`

  The position of the internal restraint object along the length of the steel design element.

8.2.2.6 `SectorPoint`

Objects of class `SectorPoint` represent the end points of a Design Element. These objects, as discussed in chapter 6, provide the ability for design elements to have end points that do not necessarily have to coincide with the end points of the included finite elements. Thus, these objects represent the end points of a design element. The concept of Sector Points is discussed in chapter 6.

Methods implemented by an object of class `SectorPoint` are:

- `public void setCoordinates(double x, double y)`

  This method sets the global coordinates of the sector point.

- `public double[] getCoordinates()`

  This method returns the global coordinates of the sector point.

8.2.2.7 Material

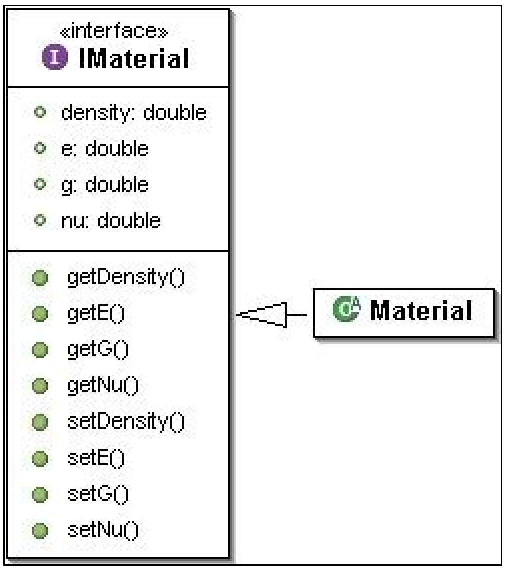


Figure 8-10 Material hierarchy

Objects of class `Material` implements interface `IMaterial` and represent general material objects in the application. These material objects are used to represent a general material type used in structural members. This allows for the ability to add additional material types to the application and thus allow for greater variation in design for different structural members. All the functionality of a general `Material` object is defined by the interface `IMaterial`. This is illustrated in Figure 8-10.

8.2.2.8 SSMaterial

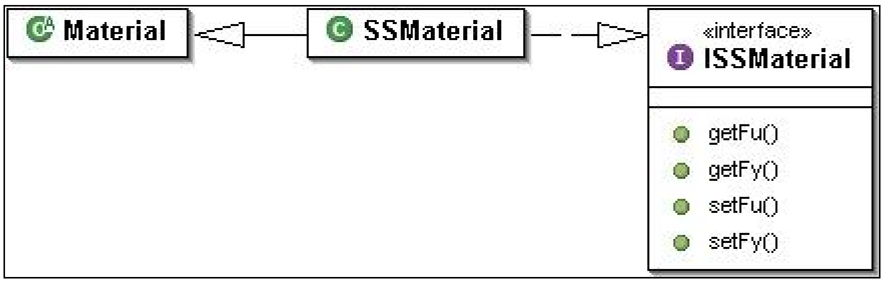


Figure 8-11 SSMaterial hierarchy

Class `SSMaterial` extends class `Material` and implements interface `ISSMaterial` and thus has all the functionality as prescribed by the interface. An object of class `SSMaterial` represents a steel material grade that is used for the design of structural steel members. All the necessary functionality of steel profiles is prescribed by interface `ISSMaterial`. This is illustrated in Figure 8-11. An object of class `SSMaterial` provides all the necessary factors of material strength for steel sections.

8.2.2.9 `Profile`

Class `Profile` implements interfaces `IClassifiedProfile` and `Comparable` and thus has all the functionality as prescribed by these interfaces. An object of class `Profile` represents a general profile used as a structural member. This allows the ability to add additional profile types for different structural members to the application.

The attributes of an object of class `Profile` are:

- `public String name;`

  The name of the profile object.

- `public double mass;`

  The mass per unit length of the profile object.

The methods implements by an object of class `Profile` are:

- `public double getMass()`

  This method returns the mass per unit length of the profile object.

- `public void setMass(double mass)`

  This method sets the mass per unit length of the profile object.

- `public String getDesc()`

  This method returns the name of the profile.

8.2.2.10 `HRSteelProfile`

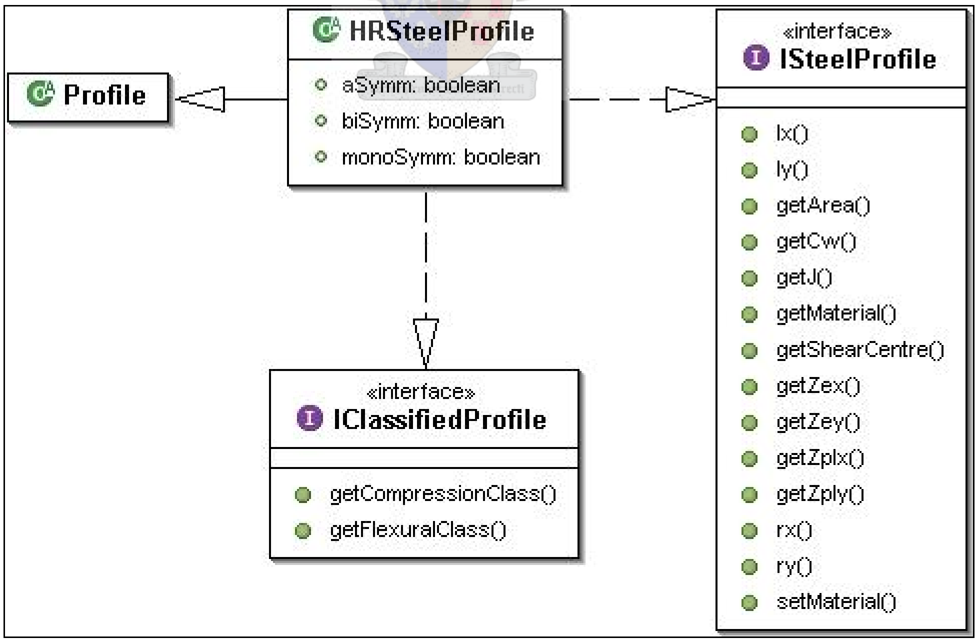


Figure 8-12 HRSteelProfile hierarchy

Class `HRSteelProfile` extends `Profile` and implements the interfaces `ISteelProfile`, `IClassifiedProfile` and thus has all the functionality as prescribed by these interfaces. An object of class `HRSteelProfile` represents a general steel section as used in a steel structural member. The hierarchy of an class `HRSteelProfile` is illustrated in Figure 8-12. This allows for the ability to add additional of steel profiles to the application.

The attributes of an object of class `HRSteelProfile` are:

- `public boolean biSymm;`

  The status of whether the steel section is bisymmetric.

- `public boolean aSymm;`

  The status of whether the steel section is asymmetric.

- `public boolean monoSymm;`

  The status of whether the steel section is monosymmetric.

There are no additional methods implemented in class `HRSteelProfile`.

8.2.2.11 IHProfile

Class `IHProfile` extends class `HRSteelProfile` and thus has all the functionality as prescribed by that class. An object of class `IHProfile` represents a specific profile type, namely an I – or H – section.

8.2.2.12 ChannelProfile

Class `ChannelProfile` extends class `HRSteelProfile` and thus has all the functionality as prescribed by that class. An object of class `ChannelProfile` represents a specific profile type, namely a channel section.

8.2.2.13 EAngleProfile

Class `EAngleProfile` extends class `HRSteelProfile` and thus has all the functionality as prescribed by that class. An object of class `ChannelProfile` represents a specific profile type, namely an equal leg angle section.

8.2.2.14 UAngleProfile

Class `UAngleProfile` extends class `HRSteelProfile` and thus has all the functionality as prescribed by that class. An object of class `UAngleProfile` represents a specific profile type, namely an unequal leg angle section.

8.2.2.15 `TProfile`

Class `TProfile` extends class `HRSteelProfile` and thus has all the functionality as prescribed by that class. An object of class `TProfile` represents a specific profile type, namely a tee section.

8.2.2.16 `RectangularHollowProfile`

Class `RectangularHollowProfile` extends class `HRSteelProfile` and thus has all the functionality as prescribed by that class. An object of class `RectangularHollowProfile` represents a specific profile type, namely a rectangular or square hollow section.

8.2.2.17 `CircularHollowProfile`

Class `CircularHollowProfile` extends class `HRSteelProfile` and thus has all the functionality as prescribed by that class. An object of class `CircularHollowProfile` represents a specific profile type, namely a circular hollow section.

8.2.2.18 `DesignSet`

Objects of class `DesignSet` are used to represent structural component (structural member) sets in a structure, as discussed in chapter 7.

Methods implemented by an object of class `DesignSet` are:

- `public void setCurrentDesignType(byte type)`

  This method sets a design type for all the design elements that are to be created. All design elements belonging to the DesignSet object will be created with such a design type.

- `public boolean getBracedState()`

  This method returns the braced state of the whole or portion of the structure that the DesignSet represents.

- `public void setDefaultRestraint()`

  This method sets the default restraints for the design set.

- `public Restraint[] getDesignSetRestraint(byte type)`

  This method returns the restraint array of the designset for a particular design type.

- `public void setChosenProfile(Profile selectedProfile)`

  This method sets the overall chosen profile for the DesignSet.

- `public Profile getChosenProfile()`

  This method returns the chosen profile for the DesignSet. This profile is then the representative profile that is to be used in the structure or portion of structure that the design set represents.

- `public List getPassedProfileList()`

  This method returns the overall ordered adequate profile list for the DesignSet. All the profiles in this list are adequate for every design element that has a reference to this DesignSet.
- `public Set getLoadedProfileSet()`

  This method returns the complete set of loaded profiles of the design set. This collection of profiles is then used in the design procedure.
- `public boolean getDesignState()`

  This method returns the status of whether the DesignSet has been designed or not. `True` if the DesignSet has been designed and `false` if not.
- `public void setDesignState(boolean designed)`

  This method sets the design state of the DesigSet.

8.3 Service Classes for Members

General: In the application, the objects that are to be designed and tested are the structural members themselves. Structural members are recognized and handled differently according to the different types of internal forces and loading that are present in the member itself. This results in a structural member being assigned a design type that is indicative of the procedures that must be followed in the design process. The design types implemented as discussed earlier are: BEAM, BEAM – COLUMN, CANTILEVER, COLUMN, TENSION – BEAM, and TENSION. Therefore, for all structural elements that are to be designed, each design type associated with that structural member has a different analyser or calculator that is responsible for dealing with the applicable design procedure. These different analyser or calculator classes reside in the `service` package for structural members. The specific calculator for each design type extends class `FactorCalculator` which will be described later in this chapter. The purpose of the calculators is to design the structural members according to the relevant design type as well as to display textual and graphical feedback of the design. The design procedures implemented are done according to *SANS 10162: Code of Practice for the Structural Use of Steel: Part 1: Limit states design of hot-rolled steelwork – 2005* as discussed in the specification in chapter 4. The calculator classes are discussed in sections 8.3.1 and 8.3.2. The rest of the service classes include classes `DelGenerator`, `Application`, `DataBaseReader` and `DesignParameter`.

The remainder of this chapter will briefly discuss the service package classes and their functionality. A more detailed description of the service classes is provided in the Java documentation of the application.

8.3.1 Calculator Hierarchy

The calculator hierarchy for the calculators is shown in Figure 8-13. As can be seen from the figure, all the design types were included in the hierarchy. In order to accommodate changes in the design procedures or to add additional design procedures, the contents of these classes need to be changed.

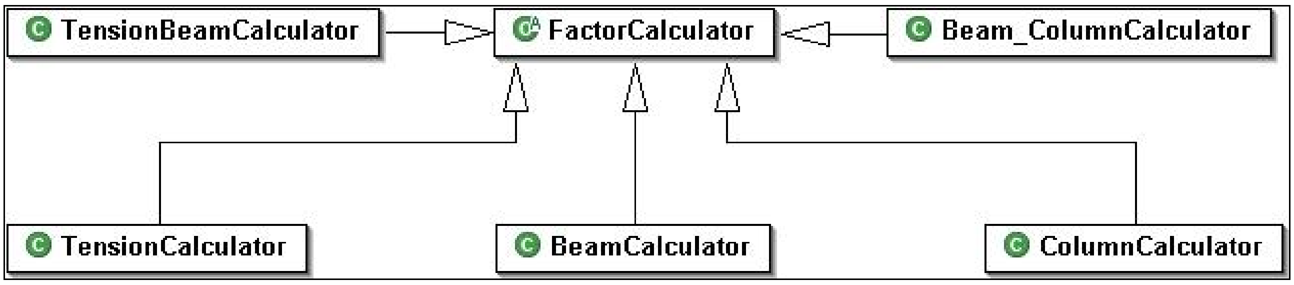


Figure 8-13 Calculator hierarchy

8.3.2 Calculator descriptions

Axis systems: For the purpose of maintaining clear transition of the descriptions of the cross sectional and member factors used for design, the axis system as described by *SANS 10162 Part1: The Structural use of Steel* was implemented by the design calculators of the framework. This is illustrated in Figure 8-14. All the method names and descriptions implemented in the calculator classes refer to the local axis system described by the design code and not by the axis system developed in chapters 3 and 6 of the Design Elements.

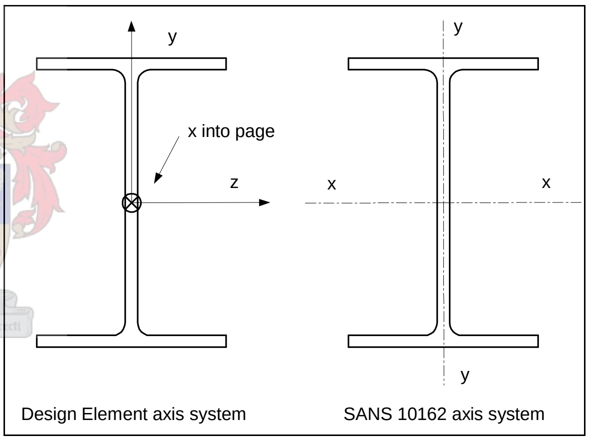


Figure 8-14 Local axis systems of model and design code

8.3.2.1 `FactorCalculator`

Class `FactorCalculator` provides the general attributes and methods that are needed to analyse a steel member. These methods calculate the dimensionless factors that are common for the design procedures that are implemented in the application, entire cross sectional resistances of the steel members as well as other common member resistances used by the various design procedures. Important methods are as follows:

- `public static void setEffLengthFactor(ISSteelDesignElement el)`

  This method calculates and sets the effective length factors of the steel member.

- `public static double getKappa_X(ISSteelDesignElement el)`

  This method calculates and returns the κ value as calculated from x – axis end moments of the steel member.
- `public static double getOmega2_X(ISSteelDesignElement el)`

  This method calculates and returns the ω_2 value as calculated from the κ_x value of the steel member.
- `public static double getVr(HRSteelProfile section)`

  This method calculates and returns the shear resistance of a steel profile.
- `public static double getCrossSecMrx(HRSteelProfile section)`

  This method calculates the cross sectional moment of resistance about the local x – axis of the steel section.
- `public static double getCrossSecMry(HRSteelProfile section)`

  This method calculates the cross sectional moment of resistance about the local y – axis of the steel section.
- `public static double getMcr(HRSteelProfile section, ...)`

  This method calculates the torsional flexural critical moment of the steel member.
- `public static double getCrossSecCr(HRSteelProfile p)`

  This method calculates the cross sectional compressive resistance of the steel member.
- `public static double getGCrossSecTr(ISSteelDesignElement el, ...)`

  This method calculates the gross cross section tensile resistance of the steel member.

8.3.2.2 BeamCalculator

Class `BeamCalculator` contains the methods and attributes that are needed to design any steel structural member that is subjected to flexural bending only. Uniaxial strong axis bending, uniaxial weak axis bending as well as biaxial bending are covered by this calculator.

Important methods implemented are:

- `public static double getUnsupMrx(HRSteelProfile section, ...)`

  This method returns the unsupported strong axis bending strength of the steel member. Weak axis bending resistance is covered by the parent class `FactorCalculator`.
- `private static void filterProfileSet(Set profileSet, ...)`

  This method removes all the steel profiles that are not suitable or allowed for use with the design procedure.

8.3.2.3 `ColumnCalculator`

Class `ColumnCalculator` contains the methods and attributes that are needed to design any structural steel member that is subjected to axial compression only.

Important methods implemented are:

- .`public static SSInternalElement getLSSegment(SSDesignElement del)`

  This method returns the longest portion of the steel member that is unbraced against buckling about its strong axis.

- `public static SSInternalElement getLWSegment(SSDesignElement del)`

  This method returns the longest portion of the steel member that is unbraced against buckling about its weak axis.

- `public static double getFBResistance(SSDesignElement del, ...)`

  This method returns the flexural buckling resistance of the steel member.

- `public static double getTFBResistance(SSDesignElement del, ...)`

  This method returns the torsional buckling or torsional flexural buckling resistance of the steel member.

- `public static double getCr(SSDesignElement del, HRSteelProfile p)`

  This method returns the overall compressive resistance of the steel member.

- `public static void filterProfileSet(Set profileSet, ...)`

  This method removes all the steel profiles that are not suitable or allowed for use with the design procedure.

8.3.2.4 `BeamColumnCalculator`

Class `BeamColumnCalculator` contains the methods and attributes that are needed to design any structural steel member that is subjected to combined flexural bending and axial compression. Uniaxial strong axis bending, uniaxial weak axis bending as well as biaxial bending are covered by this calculator. Important methods implemented are:

- `public static double getCrossSecU1x(ISSteelDesignElement is, ...)`

  This method calculates the factor that takes into account the second order effects of a deformed member under axial compression, with reference to the cross sectional x axis of the member.

- `public static double getCrossSecU1y(ISSteelDesignElement is, ...)`

  This method calculates the factor that takes into account the second order effects of a deformed member under axial compression, with reference to the cross sectional y axis of the member.

- `private static double getCrossSecStrengthRatio(ISSteelDesignElement is, .)`
  This method returns the cross sectional strength ratio of a steel member, according to the appropriate interaction formula.
- `private static double getStrongAxisMemberRatio(ISSteelDesignElement iel, )`
  This method returns the overall member strength ratio of a steel member subjected to uniaxial strong axis bending, according to the appropriate interaction formula.
- `private static double getBiAxialMemberRatio(ISSteelDesignElement is, ...)`
  This method calculates the overall member strength ratio of a steel member subjected to bi axial bending, according to the appropriate interaction formula.
- `public static double getLatTorBucklingRatio(ISSteelDesignElement is, ...)`
  This method calculates the lateral torsional buckling strength ratio of a steel member. The bending resistance is calculated according to lateral torsional buckling strength and the compressive resistance is calculated according to weak axis buckling. The ratio is determined according to the appropriate interaction formula.

8.3.2.5 `TensionBeamCalculator`

Class `TensionBeamCalculator` contains the methods and attributes that are needed to design any steel structural member that is subjected to combined flexural bending as well as axial tension. The axial resistance of the ends of the member are not considered. There are no additional methods for this calculator.

8.3.2.6 `TensionCalculator`

Class `TensionCalculator` contains the methods and attributes that are needed to design any steel structural member that is subjected to a tensile axial force only. This axial resistance due to connection detail is not considered. There are no additional methods for this calculator.

8.3.3 Design management classes

Class descriptions and usage: Classes `DelManager`, `DesignParameter` and `Application` are concerned with the management and organization of the design model and its components.
Class `DelManager` is responsible for the creation of Design Elements and their components and the addition of such Design Elements to the design model. This class has control over the creation of all the internal elements concerned with a particular design element as a result of the presence of internal restraints. All information that is provided by the finite element model (e.g. finite elements, nodes, load cases etc) is correctly transitioned to the concerning design element by this class. When a design element is created by the framework all the necessary active parameters used by the

implemented design code are assigned to the design element. This is executed by class `DelManager`. Active design parameters are parameters that are concerned with the creation of a Design Element. Some of this information is not provided by the finite element model and must be stipulated by the user. The framework provides the ability to adjust the design parameters for a specific design element as required. These parameters remain active in the framework and all design elements created are assigned these parameters. These parameters include design type (e.g. BEAM, COLUMN), load cases from the finite element model and created design sets. Class `DesignParameter` is concerned with the storage of the currently selected design parameters. Class `Application` manages the current design model and the underlying finite element model. This class provides the ability to interact with components of the underlying finite element model and the surrounding design model.

8.3.3.1 `DelManager`

An object of class `DelManager` controls the creation of a design element and is responsible for adding the design element and all of its components to the design model. All internal elements that are concerned with a design element are created at the point of design by this class.
Important attributes of an object of class `DelManager`:

- `private SSDesignElement del;`

  The reference to the current design element that is created.

Important methods implemented in class `DelManager`:

- `public void createNew()`

  This method creates a new design element object for the design.

- `private boolean setInfo()`

  This method assigns all the necessary and applicable components of a finite element model to a design element. These components include the selected finite elements.

- `public static void prepare(SSDesignElement del)`

  This method prepares the design element for the design process by creating all the internal elements that are applicable for a specific design.

- `private void setFactors(SSDesignElement del)`

  This method sets all the current design factors of the design element that is to be used in the design process. These include design type, load case and design set.

- `public boolean toModel()`

  This method adds the current design element as well as all its components to the current member design model.

8.3.3.2 `DesignParameter`

Class `DesignParameter` has references to the considered design parameters that are applicable for a particular design element as required by the user. This class allows for numerous design elements to have similar design parameters. These parameters can be edited on an existing design element, if preferred by the user.

Important attributes of class `DesignParameter`:

- `public static DesignSet activeDesignSet;`

  The selected design set that to which all design elements shall be added.
- `public static byte CURRENT_DESIGN_TYPE;`

  The design type of all currently created design elements.
- `private static String[] loadCases;`

  The reference to the array of all available load cases for the currently loaded finite element model.
- `private static String currentLoadCase;`

  The reference to the currently considered load case.

Important methods implemented in class `DesignParameter`:

- `public static void setDesignSetActive(DesignSet a_DesignSet)`

  This method sets a design set active. This causes all design elements that are created to be added to that design set.
- `public static void setDesignType(byte designType)`

  This method sets the current design type.
- `public static void setAvailableLoadCases(String[] loadCNames)`

  This method sets the available load cases for the design model. These load cases are obtained from the finite element model under consideration.
- `public static String[] getAvailableLoadCases()`

  This method returns the names of all the available load cases from the underlying finite element model.
- `public static void setLoadCaseActive(Model currentFEMModel, ...)`

  This method sets a specific load case active from a given finite element model.
- `public static String getActiveLoadCase()`

  This method returns the name of the current load case.

8.3.3.3 `Application`

Class `Application` has references to the loaded finite element model as well as to the current member design model. This class contains all the necessary methods for working with the components in both the loaded finite element model as well as the current member design model.
Important attributes of class `Application` are:

- `private static SSMemModel active_ssModel;`

  The current structural member design model.
- `public static Model active_femModel;`

  The current finite element model.
- `public static SSMemGui gui;`

  The main graphical user interface.
- `private static DrawPanel drawPanel;`

  The drawing surface of the application on which all components are rendered.

8.4 Member Model

Class `SSMemModel` implements interface `IModel` and represents a structural steel member design model. Class model allows one to reach all components needed for member design from searching through the component set of the model. Class `SSMemModel` has reference to a single set that contains all the components to do with member design in the application. This set is filterable in a sense that while it can contain a variety of components, a specific type of component can be found within the set by applying the correct filtering procedure.
Class `SSMemModel` has the following attributes:

- `public FilterableSet components;`

  This is a reference to the filterable set to which all components in the member design model are added. This set is traversed when looking for a specific component within the model.

Important methods of an object of class `SSMemModel`:

- `public void addComponent(AppObject component)`

  This method adds a component to the model, and in turn to the filterable set.
- `public void removeComponent(AppObject component)`

  This method removes a component from the model, and in turn from the filterable set.
- `public Iterator iterator(FilterableSet.Filter filter)`

This method returns an iterator with specific filter attributes. This iterator allows for the searching for specific components depending on the filter's attributes.

- `public int size(FilterableSet.Filter filter)`

  This method returns the number of components in the model that are specific to the type that is defined by the filter.

- `public int size()`

  This method returns the total number of components that are in the model.

- `public void clear()`

  This method removes all the components from the model.

- `public boolean isEmpty()`

  This method returns the status of the whether the model is empty or not. `True` if the model is empty and `false` if not.

- `public Set getDesignSetElements(DesignSet aDesignSet)`

  This method returns the set of design element included in a given design set.

- `public void setModelDesignState(boolean designed)`

  This method sets the design state of the model. If the model has been designed, the status is set to `true`, and `false` if not.

- `public void analyseCurrent()`

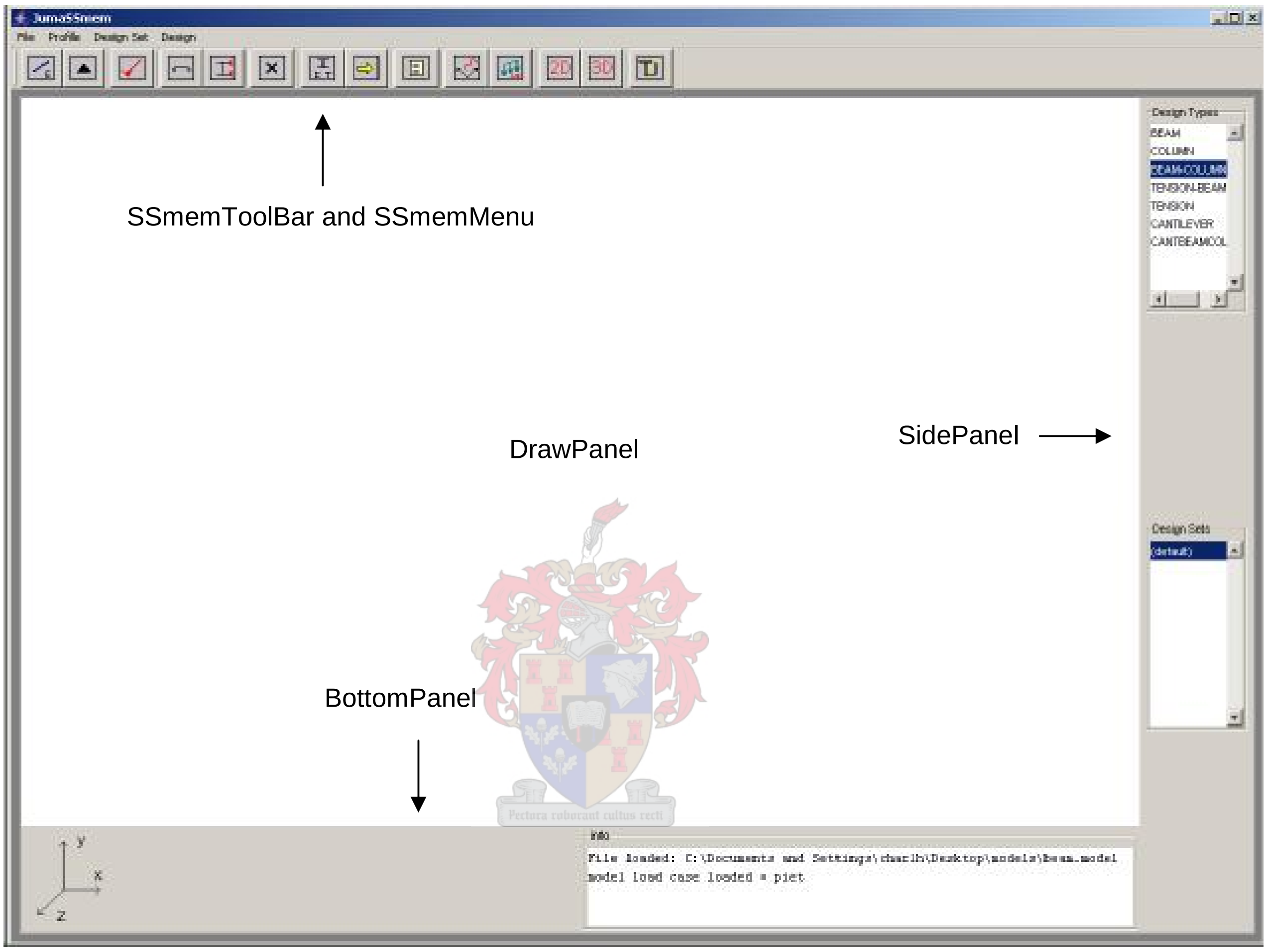
  This method designs all the components that are included in a specific design set in the model, namely the design set that is active in the application at the time of the design process.

- `public void analyseAll()`

  This method designs the model.

8.5 Graphical user interface

This part of the thesis describes the graphical user interface (GUI) for interaction between the designer and the application. The GUI facilitates the design task and complements the object oriented member design framework. Figure 8-15 shows the layout of the GUI.



```
File loaded: C:\Documents and Settings\charlh\Desktop\models\beam.model
model load case loaded = piet
```

Figure 8-15 The graphical user interface layout

8.5.1 GUI Structure

The basic GUI structure of the application is shown in Figure 8-16. This structure shows all the fixed graphical components. All classes pertaining to the graphical user interface provide functionality to visually display certain aspects about the options involved in the design model. These aspects include the display of the design elements themselves, available design sets, design types, internal restraints, external restraints, available steel profiles and other design options. The remainder of this chapter will discuss all the classes of the GUI structure as shown in Figure 8-16. A more detailed description is provided in the Java documentation of the application.

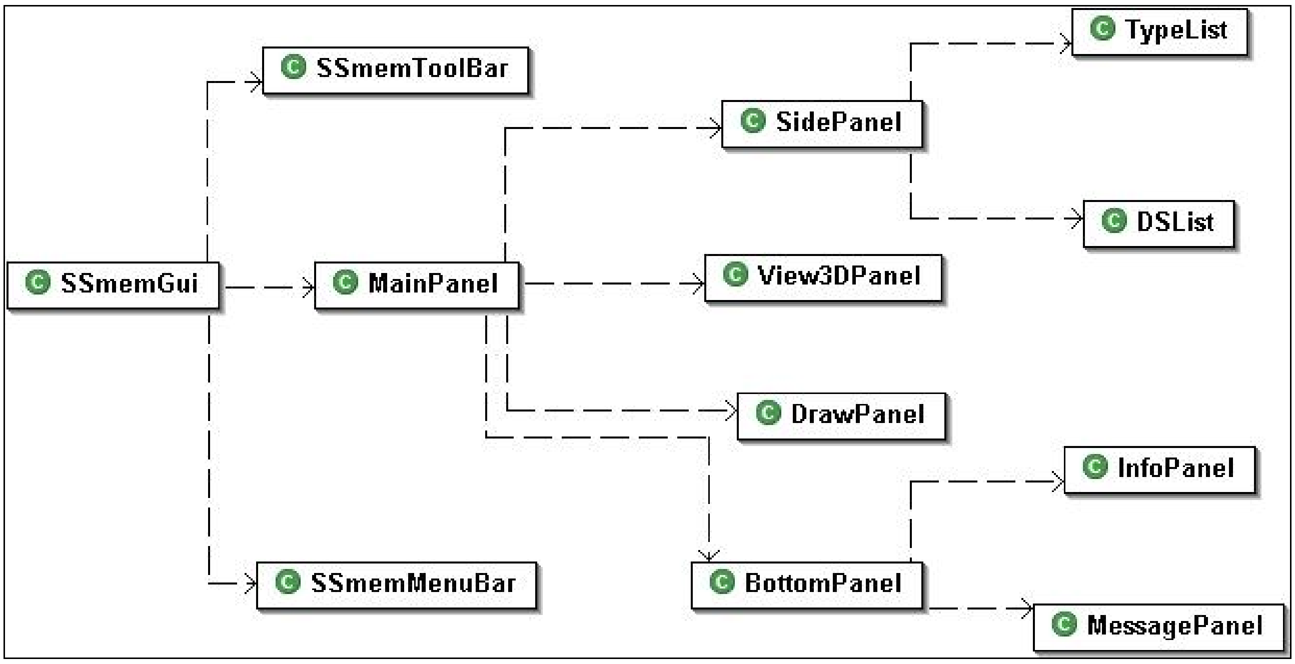


Figure 8-16 The GUI structure

8.5.2 GUI descriptions

8.5.2.1 `SSmemGui`

An object of class `SSmemGui` represents the main frame of the application. This frame contains references to the three main components of the frame, namely the menubar (`SSmemMenuBar`), the toolbar (`SSmemToolBar`) as well as the panel in the middle (`MainPanel`) of the frame. This can be seen in Figure 8-15.

Important attributes of an object of class `SSmemGui`:

- `private SSmemMenuBar menuBar;`

  The reference to the menu bar of the main frame. Class `SSmemMenuBar` is discussed in section 8.5.2.2.

- `private SSmemToolBar toolBar;`

  The reference to the tool bar of the main frame. Class `SSmemToolBar` is discussed in section 8.5.2.3

- `public MainPanel mainPanel;`

  The reference to the main panel of the main frame. Class `MainPanel` is discussed in section 8.5.2.4.

Important methods implemented in class `SSmemGui`:

- `public SSmemMenuBar getSSMenuBar()`

This method returns the menubar of the main frame.

- `public SSmemToolBar getSSToolBar()`

  This method returns the toolbar of the main frame.

8.5.2.2 SSmemMenuBar

An object of class `SSmemMenuBar` represents a menubar of the main frame for the application. The menubar consists of four menus, namely ‘*File*’, ‘*Profile*’, ‘*DesignSet*’ and ‘*Design*’. The ‘*File*’ menu is responsible for opening a finite element model file. In order to create a design model and make use of the application, a finite element model must be loaded from the ‘*File*’ menu as mentioned earlier. The application can be terminated from the ‘*File*’ menu as well. The various menus in class SSmemMenuBar hold references to various ‘pop up’ dialogs and frames that provide a further array of functionality to the GUI. These graphical components will be discussed in section 8.5.3. Class `SSmemMenuBar` extends `ActionListener` and is responsible for listening to all the events generated by its components.

Important attributes of an object of class `SSmemMenuBar`:

- `private JMenu fileMenu;`

  The reference to the ‘*File*’ menu of the menubar. This menu contains two menu items such as ‘*Open*’ – to get access to a finite element model file and ‘*Exit*’ – to exit the application.

- `private JMenu profileMenu;`

  The reference to the ‘*Profile*’ menu of the menubar. This menu contains a single menu item, namely ‘*profile database*’ – to get access to the database of steel profiles for use in the design of steel members.

- `private JMenu dsMenu;`

  The reference to the ‘*DesignSet*’ menu of the menubar. This menu contains a single menu item, namely ‘*add design set*’ – to create additional `DesignSet` objects for use in the application.

- `private JMenu designMenu;`

  The reference to the ‘*Design*’ menu of the menubar. This menu contains a two menu items, namely ‘*design all*’ – to design all the steel members in the application and ‘*design current*’ – to design all the steel members in the currently selected design set.

Important methods implemented in class `SSmemMenuBar`:

- `public void enableAll()`

This method enables all the components of the menubar object.

- `public void disableAll()`

  This method disables all the components of the menubar object.

8.5.2.3 `SSmemToolBar`

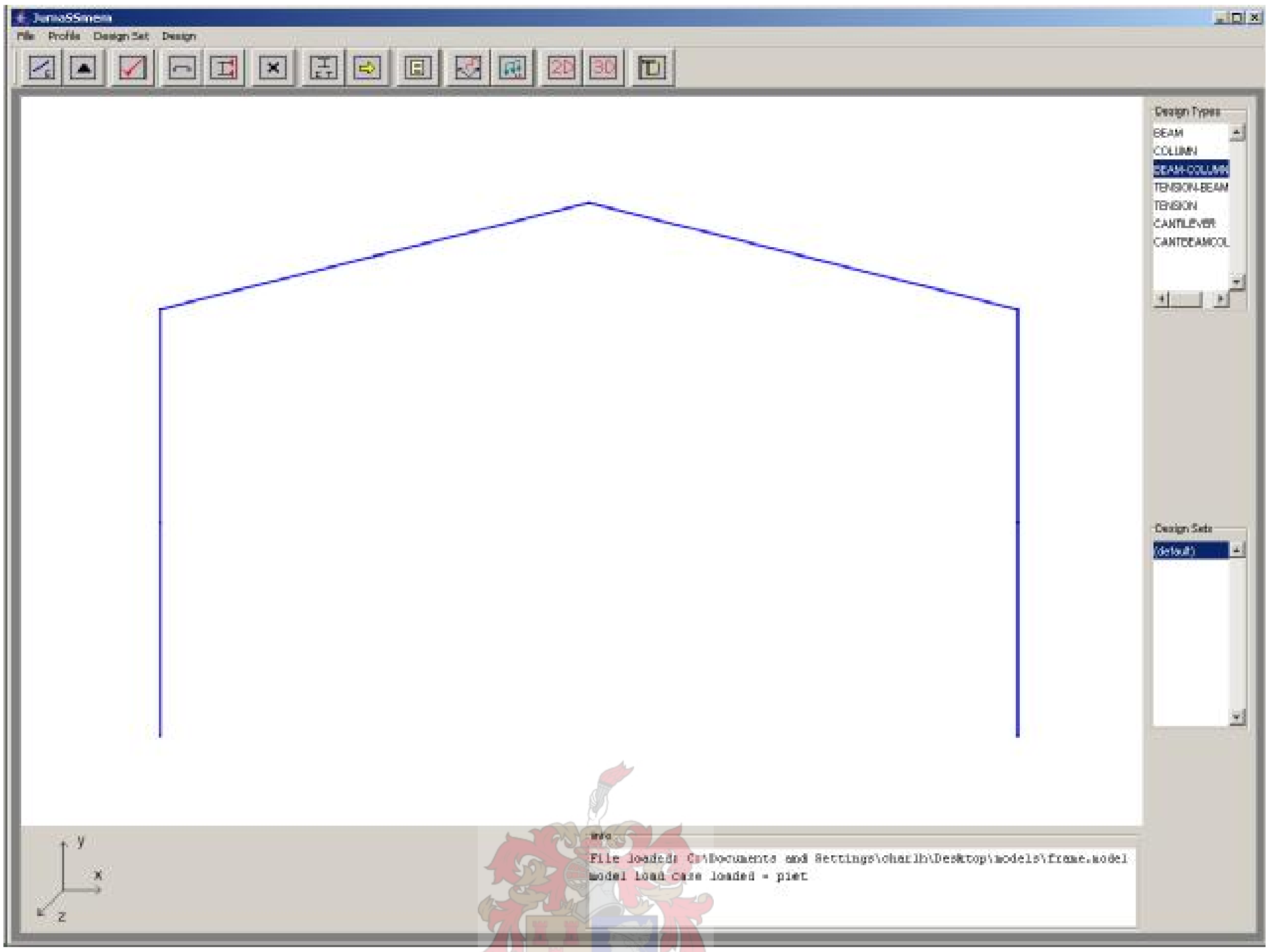
An object of class `SSmemToolBar` represents a toolbar which contains tools for editing and creating design elements and thus creating a design model. This is shown in Figure 8-15. The object consists of 14 buttons all with certain functionality or ability to invoke functionality. The functionality of these buttons are namely to: select a finite element, add a sector point on to the ends of a design element, edit a design element, add an internal restraint to a design element, get access to the steel profile database, design the current design set, show the results of a design in the form of a calc sheet, display and select different load cases of the finite element model, display the all the internal force diagrams of a design element, remove particular design elements from the member design model as well as to toggle between 2 D and 3 D perspectives of the member design elements. All the additional graphical components that these buttons represent will be discussed in section 8.5.3 as mentioned in 8.5.2.2.

8.5.2.4 `MainPanel`

An object of class `MainPanel` contains references to four components, namely the drawing surface (`DrawPanel`), the panel at the bottom (`BottomPanel`), the panel on the side (`SidePanel`) and the panel that renders the 3D view of the members (`View3DPanel`). This is illustrated in Figure 8-16.

8.5.2.5 `DrawPanel`

Objects of class `DrawPanel` provide the connection between two dimensional shapes and the physical components the shapes actually represent. This class allows for the two dimensional rendering of the structural steel member components as well as for the manipulation of these components through graphical interaction. This class forms the basis for interaction with the finite element model as well as the structural steel design model. Figure 8-17 shows a `DrawPanel` with a finite element model of a portal frame loaded.



```
File loaded: C:\Documents and Settings\charlb\Desktop\models\frame.model
model load case loaded = piet
```

Figure 8-17 Illustrating an object of class `DrawPanel`

Important methods implements in class `DrawPanel`:

- `public Object getRelatedObject(DrawableShape shape)`

  This method returns a component that is related to a specific provided graphical shape.
- `public DrawableShape getRelatedShape(Object val)`

  This method returns the shape that is related to a provided component.
- `public void enablePointSelecter(boolean enable)`

  This method enables shapes to be selected from a point and click procedure from mouse input level.
- `public void enableBoxSelecter(boolean enable)`

  This method enables shapes to be selected by creating a bounding box from a point and drag procedure from mouse input level.
- `public DrawableShape select(Point2D p)`

  This method selects a specific shape from a given point in 2 dimensional space provided the given point lies within the bounds of the shape.
- `public boolean put(DrawableShape ds, Object obj)`

This method adds a component or object and its representative shape to the registry of the panel, thus allowing for the graphical rendering of the component.

- `public void drop(DrawableShape ds)`

  This method removes a shape from the registry of the panel, thus removing it from the drawing surface.

8.5.2.6 `SidePanel`

An object of class `SidePanel` forms the starting point of selecting design types and design sets for the application. The object consists of two lists namely, '*Design Types*' and '*Design Sets*'.

The '*Design Types*' list contains an object of class `TypeList` which represents the list used to display all the available design types as well as provides the ability to select specific design types. These design types are applied to the design elements and govern the way in which they are handled in the design process. This is shown in Figure 8-18. As an implementation aspect, while a particular design type is highlighted, all design elements created in that instance will be assigned that particular design type. Class `TypeList` is described in section 8.5.2.10.

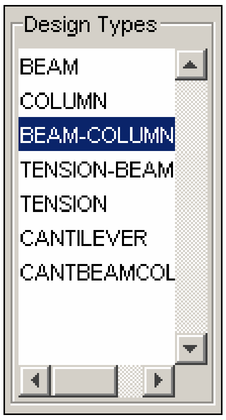


Figure 8-18 Design types

The '*Design Sets*' list contains an object of class `DSList` which represents the list used to display all the available design sets as well as the ability to select specific design sets. These design sets are used to group specific members together for specific design reasons. Available design sets for a particular design (e.g. default, beams, columns) are shown in Figure 8-19. As an implementation aspect, while a particular design set is highlighted, all design elements created in that instance will be added to that design set. Class `DSList` is described in section 8.5.2.11.

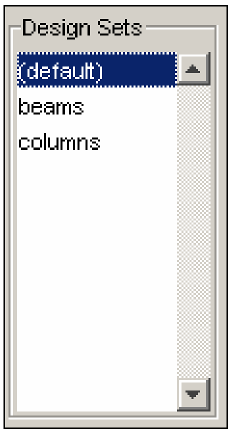


Figure 8-19 Design sets

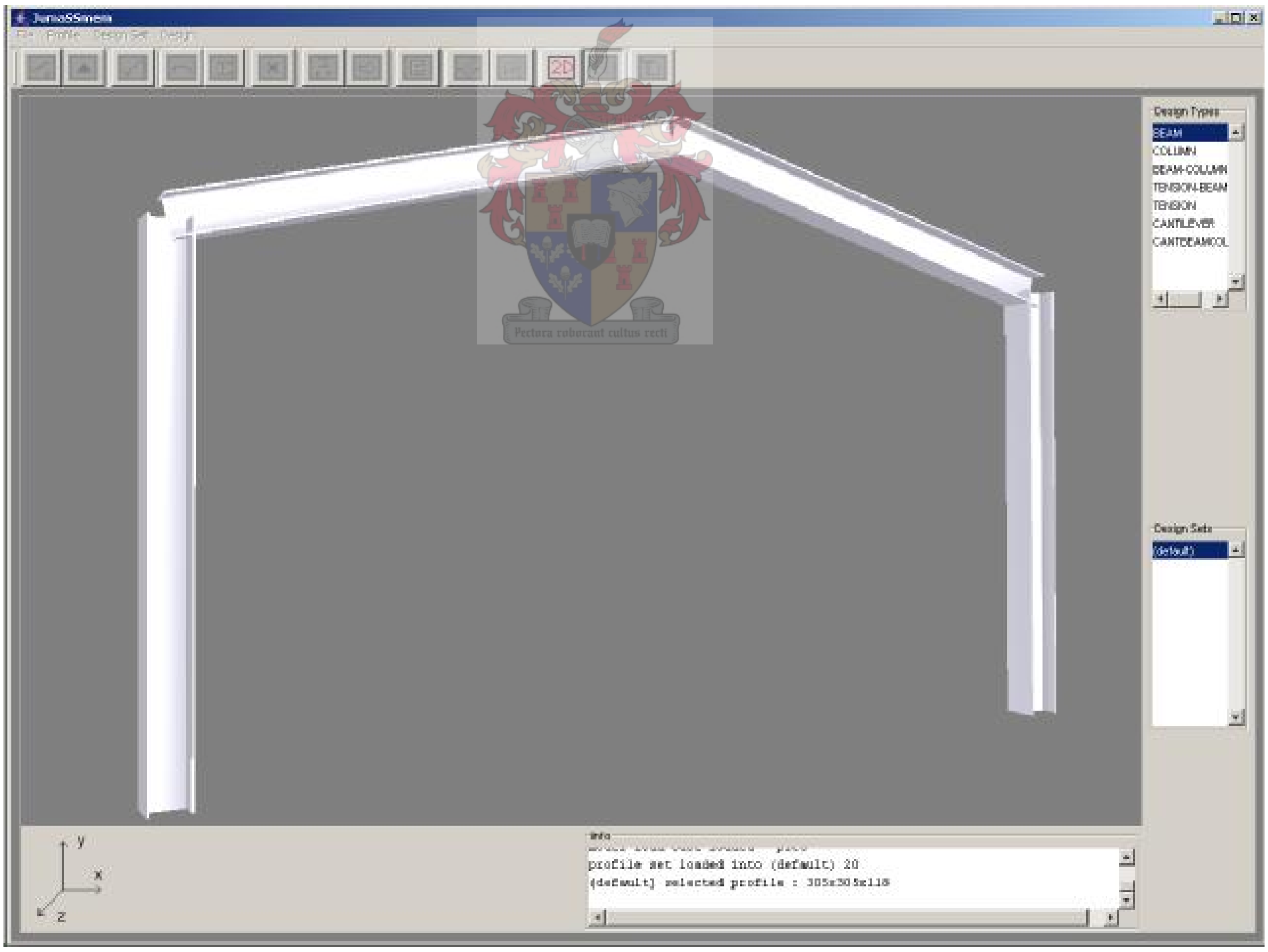
8.5.2.7 `BottomPanel`

An object of class `BottomPanel` is used primarily to display messages and information to the user. The object consists of two panels, namely a panel that provided information about the commands that the user executes as well as a panel that display information about the different design elements that have been created.

The panel that provides the information about the commands of the user is an object of class `MessagePanel`. This class is described in Section 8.5.2.12.
The panel that provides graphical and textual information about the specific design elements is an object of class `InfoPanel`.

8.5.2.8 `View3DPanel`

An object of class View3DPanel represents a three dimensional viewing panel for displaying the designed steel members in scale in three dimensions. The displayed steel members are fully rotatable, pannable and zoomable.
Each of the included steel profiles in the database is represented by a different three dimensional shape, creating scaled and distinguishable three dimensional profiles. Any structural geometry can be converted to three dimensions and displayed on this panel. Figure 8-20 illustrates the three dimensional viewing panel.



```
profile set loaded into (default) 20
[default] selected profile : 305x305x118
```

Figure 8-20 Illustrating an object of class `View3DPanel`

8.5.2.9 DataTextPane

An object of class `DataTextPane` represents a text pane that contains the relevant data for the specific member design. The data includes the profiles that are adequate for the steel members, the internal forces of the steel members, the design input and the design results.

Important methods implemented in class `DataTextPane`:

- `public void setHeaderInfo()`

  This method sets the header information for the specific member design. The header information included the company name, the designer's name, the project name and the data.

8.5.2.10 TypeList

An object of class `TypeList` represents the list in which all the design types available in the application are stored. These design types govern the manner in which the steel members are designed. This object allows for the graphical representation of the design types as well as the functionality to select the design type for a particular member or group of members. This graphical component is illustrated in Figure 8-18.

8.5.2.11 DSList

An object of class `DSList` represents the list in which all the design sets available in the application are stored. These design sets govern the manner in which the steel members are designed as whole. This object allows for the graphical representation of these design sets as well as the functionality to select the design set for a particular member or group of members. This graphical component is illustrated in Figure 8-19.

8.5.2.12 MessagePanel

Class `MessagePanel` is responsible for the textual display of messages to the user. This class consists of text editor for textually displaying information. The application makes use of this class to display simple messages to the user about the commands that have been entered.

8.5.2.13 InfoPanel

An object of class `InfoPanel` provides a graphical representation of information about design elements that are currently displayed. This class provides information about the length, name, design type, design set and selected profile type of a design element or member. When a design element is not selected, the class displays the global axis system.

8.5.3 GUI Editors and Further Components

Apart from the fixed GUI components mentioned in section 8.5.2, there are various other graphical components that are responsible for editing design parameters and displaying certain information about the various elements at a time as required by the user. These graphical components are temporary editors that are not always present or required by the user of the application. Furthermore, these components differ in appearance and functionality depending on the context in which they are required.

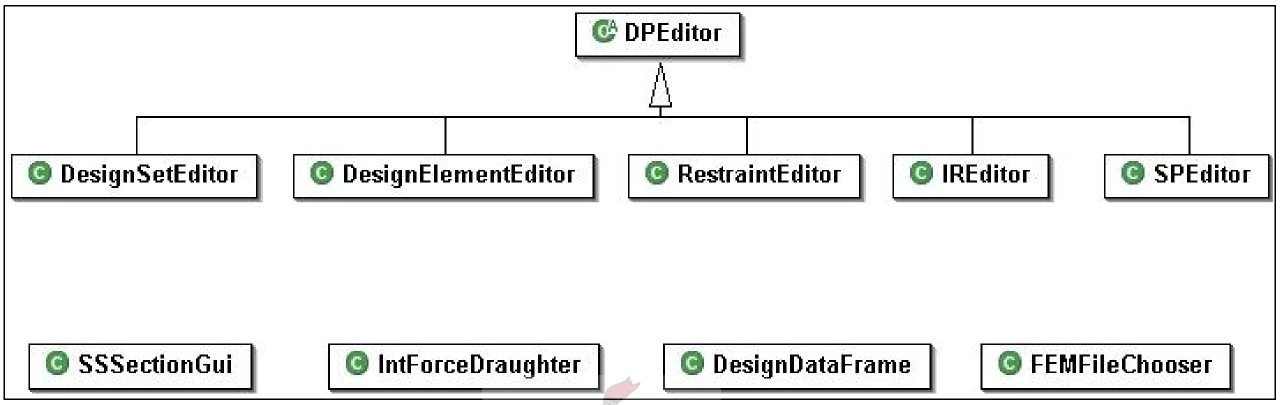


Figure 8-21 The Gui editor hierarchy

The remainder of this section will discuss the GUI editor hierarchy as illustrated in Figure 8-21. A more detailed description and explanation of each component is provided in the Java Documentation of the application.

8.5.3.1 `DPEditor`

An object of class `DPEditor` provides for the basic functionality of creating a 'pop up' graphical component that can be modified for any context in which it is required. This class forms the basis for all the other editor classes.

8.5.3.2 `SPEditor`

An object of class `SPEditor` extends `DPEditor` and provides the graphical link for creating sector point objects along the length of an element. This editor is illustrated in Figure 8-22. The editor contains an object of class `JTextField` in which the offset as a fraction of the specific finite element's length is entered. For the purpose of placing sector points, the start point of a finite element is determined by the proximity of where the user clicks the mouse on a selected finite element. The offset is the n measured relative to this temporary start point.

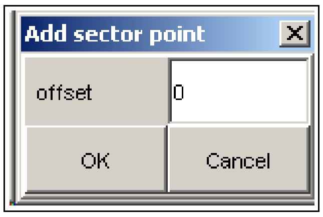
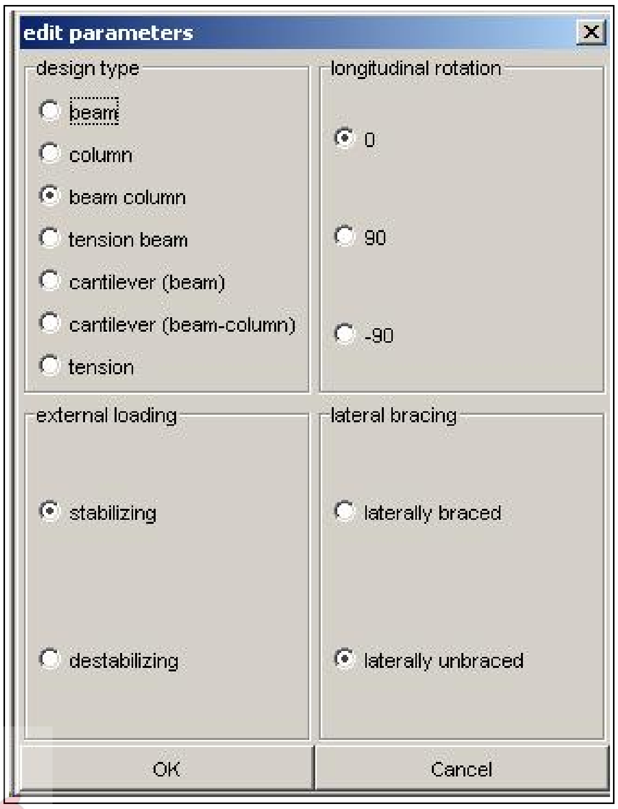


Figure 8-22 Sector Point graphical editor

8.5.3.3 `DesignElementEditor`

An object of class `DesignElementEditor` extends `DPEditor` and provides the graphical link for editing the design parameters for all created steel elements. This editor is illustrated in Figure 8-23.

This editor consists of four main panels. The first panel is the ‘*design type’* panel, which allows the user to change the design type of an individually selected design element. The second panel is the ‘*longitudinal rotation* panel, which allows the user to rotate the design element in 90 degree intervals about its longitudinal x - axis. Other multiples are not implemented. The third panel, namely the *‘external loading’* panel, is concerned with the external loading stability of the element. The final panel, namely the ‘*lateral bracing’* panel, allows the user to state whether the element is continuously laterally restrained along its length or not. There are no important attributes or methods for this class.



edit parameters

design type: beam, column, [✓ beam column], tension beam, cantilever (beam), cantilever (beam-column), tension

longitudinal rotation: [✓ 0], 90, -90

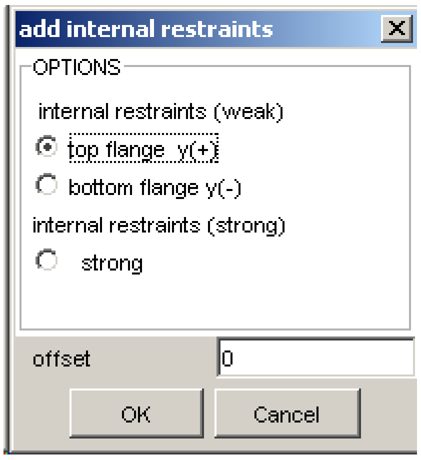
external loading: [✓ stabilizing], destabilizing

lateral bracing: laterally braced, [✓ laterally unbraced]

Figure 8-23 Design Element graphical editor

8.5.3.4 `IntRestraintEditor`

An object of class `IntRestraintEditor` extends `DPEditor` and provides the graphical link for creating and placing internal restraints along the length of a steel element. This editor is illustrated in Figure 8-24. This editor consists of an option to apply an internal restraint to either the top or bottom flanges or both flanges simultaneously of a steel member or SSDesign Element, as indicated by the ‘+y’ and ‘-y’ indicators. The offset field is to position the restraint at the given fraction of the length of the steel member or SSDesign Element. The user has the option to remove or edit existing internal restraints.



add internal restraints

OPTIONS

internal restraints (weak): [✓ top flange y(+)], bottom flange y(-)

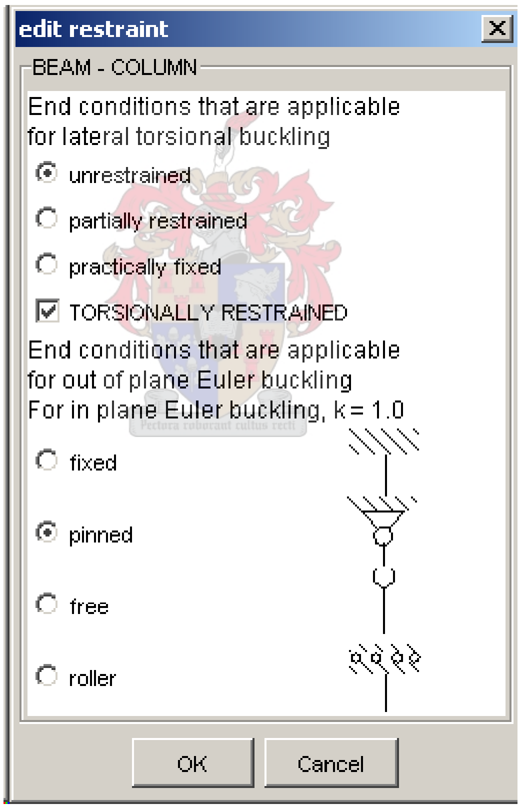
internal restraints (strong): strong

offset: 0

Figure 8-24 Internal Restraint graphical editor

8.5.3.5 `RestraintEditor`

An object of class `RestraintEditor` extends class `DPEditor` and provides the graphical functionality to change and edit the restraint conditions at the ends of the steel members. These objects are design type sensitive, meaning that their appearance and functionality are dependent on the design types of the individual steel members. There are six main types of restraint editors and they are dependent on the design types BEAM, CANTILEVER, BEAMCOLUMN, CANTILEVER BEAMCOLUMN and COLUMN. The restraint options for a BEAMCOLUMN are illustrated in Figure 8-25. For such a design type and restraint, the panel is divided into 2 parts. The top panel refers to end restraints that are applicable for lateral torsional buckling. The bottom panel refers to the restraint conditions against weak axis (out of plane) Euler buckling. These restraint options must be chosen correctly for the applicable member.


edit restraint

BEAM - COLUMN

End conditions that are applicable for lateral torsional buckling

[✓ unrestrained] partially restrained practically fixed

[✓ TORSIONALLY RESTRAINED]

End conditions that are applicable for out of plane Euler buckling

For in plane Euler buckling, k = 1.0

fixed [✓ pinned] free roller

Figure 8-25 Beam and column restraint options

8.5.3.6 `LCEditor`

An object of class `LCEditor` extends class `DPEditor` and provides the graphical functionality for the user to select and load different load cases from the current finite element model. This object allows the user to switch between the different load cases at any time during the construction of the design model. Figure 8-26 illustrates this editor with two available load cases for the model.

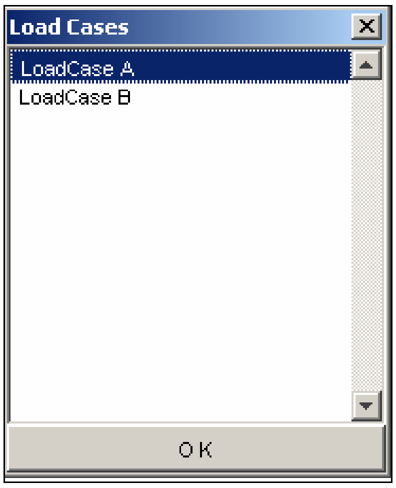
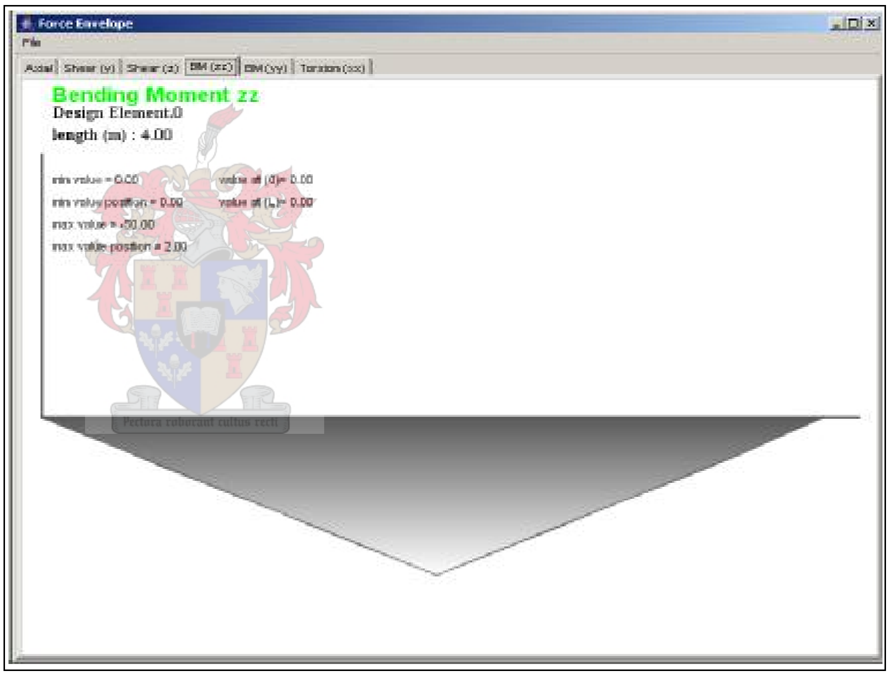


Figure 8-26 Load case editor

8.5.3.7 `IntForceDraughter`

An object of class `IntForceDraughter` is used to display all the necessary internal force diagrams for a particular steel member created in the design model.



Design Element:0
length (m) : 4.00
min value = 0.00
value at (0)= 0.00
min value position = 0.00
value at (L)= 0.00
max value = -50.00
max value position = 2.00

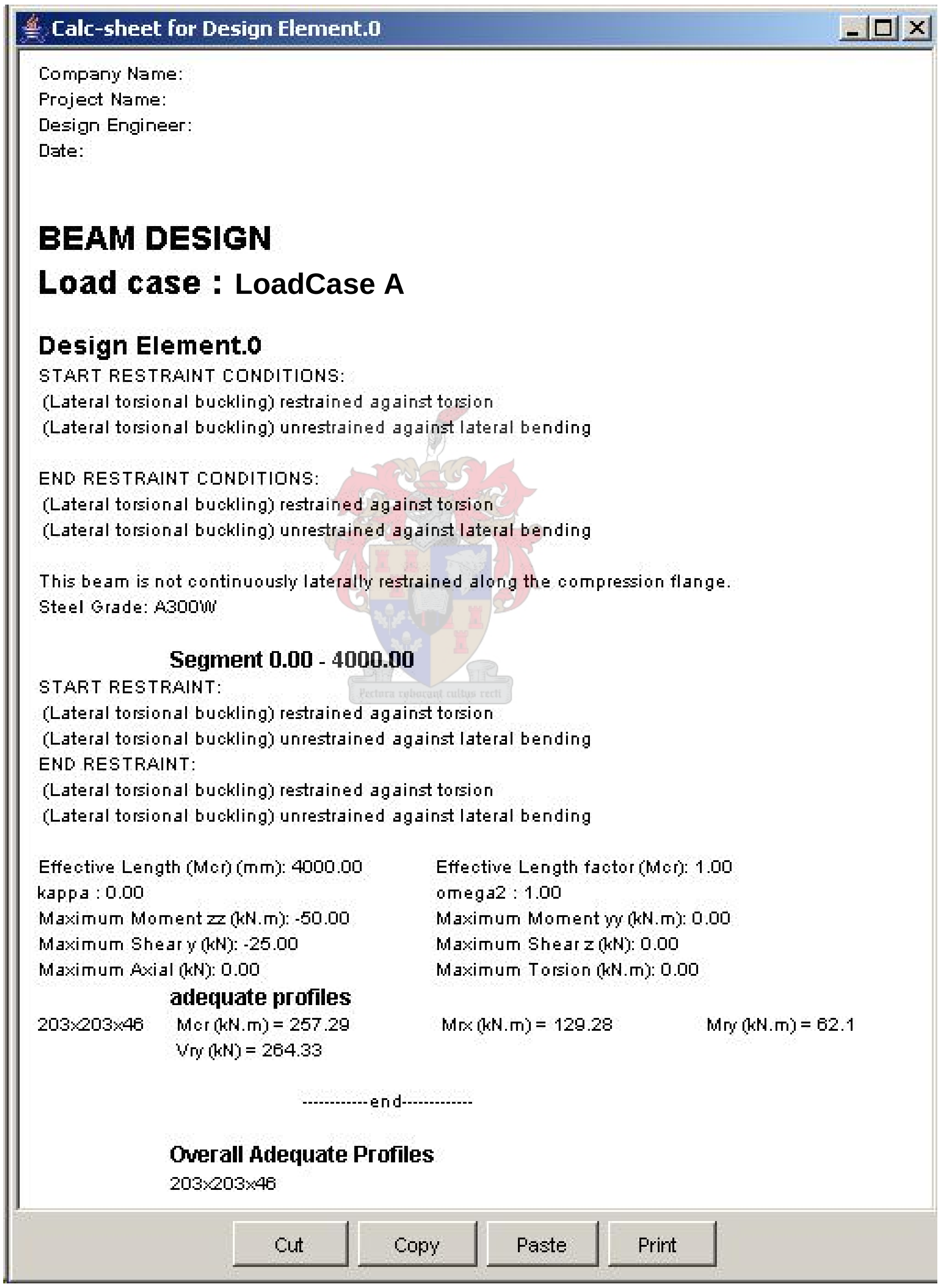
Figure 8-27 Internal force diagrams

This graphical component consists of six tabbed panes, namely '*Axial'*, '*Shear y*', '*Shear z*', '*BM zz*, '*BM yy'* and '*Torsion xx'*. These functions of these tabbed panes are discussed below.

The '*Axial'* tabbed pane is used to graphically display the axial force diagram of the selected steel member. The '*Shear y'* and '*Shear z'* tabbed panes are used to display the shear force diagram in the direction of the local y axis and local z axis of the selected steel member respectively.
The '*BM zz'* and '*BM yy'* tabbed panes are used to display the bending moment diagram about the local z axis and local y axis of the steel member respectively.
The '*Torsion xx'* tabbed pane is used to display the torsional moment diagram of the steel member. The tabs are illustrated in Figure 8-27.

8.5.3.8 `DesignDataFrame`

An object of class `DesignDataFrame` includes a graphical text area that is used to display the design results in a structured and ordered data sheet. This data sheet can be edited as well as printed out to record all the design procedures and data. The layout of the data sheets differ according to design type. Figure 8-28 illustrates a data sheet for a steel member that has been designed as a beam. All the necessary information about the member is shown.



Company Name:
Project Name:
Design Engineer:
Date:

BEAM DESIGN
Load case : LoadCase A

Design Element.0
START RESTRAINT CONDITIONS:
(Lateral torsional buckling) restrained against torsion
(Lateral torsional buckling) unrestrained against lateral bending

END RESTRAINT CONDITIONS:
(Lateral torsional buckling) restrained against torsion
(Lateral torsional buckling) unrestrained against lateral bending

This beam is not continuously laterally restrained along the compression flange.
Steel Grade: A300W

Segment 0.00 - 4000.00
START RESTRAINT:
(Lateral torsional buckling) restrained against torsion
(Lateral torsional buckling) unrestrained against lateral bending
END RESTRAINT:
(Lateral torsional buckling) restrained against torsion
(Lateral torsional buckling) unrestrained against lateral bending

```
Effective Length (Mcr) (mm): 4000.00      Effective Length factor (Mcr): 1.00
kappa : 0.00                              omega2 : 1.00
Maximum Moment zz (kN.m): -50.00          Maximum Moment yy (kN.m): 0.00
Maximum Shear y (kN): -25.00              Maximum Shear z (kN): 0.00
Maximum Axial (kN): 0.00                  Maximum Torsion (kN.m): 0.00
```

adequate profiles

```
203x203x46   Mcr (kN.m) = 257.29     Mrx (kN.m) = 129.28     Mry (kN.m) = 62.1
             Vry (kN) = 264.33
```

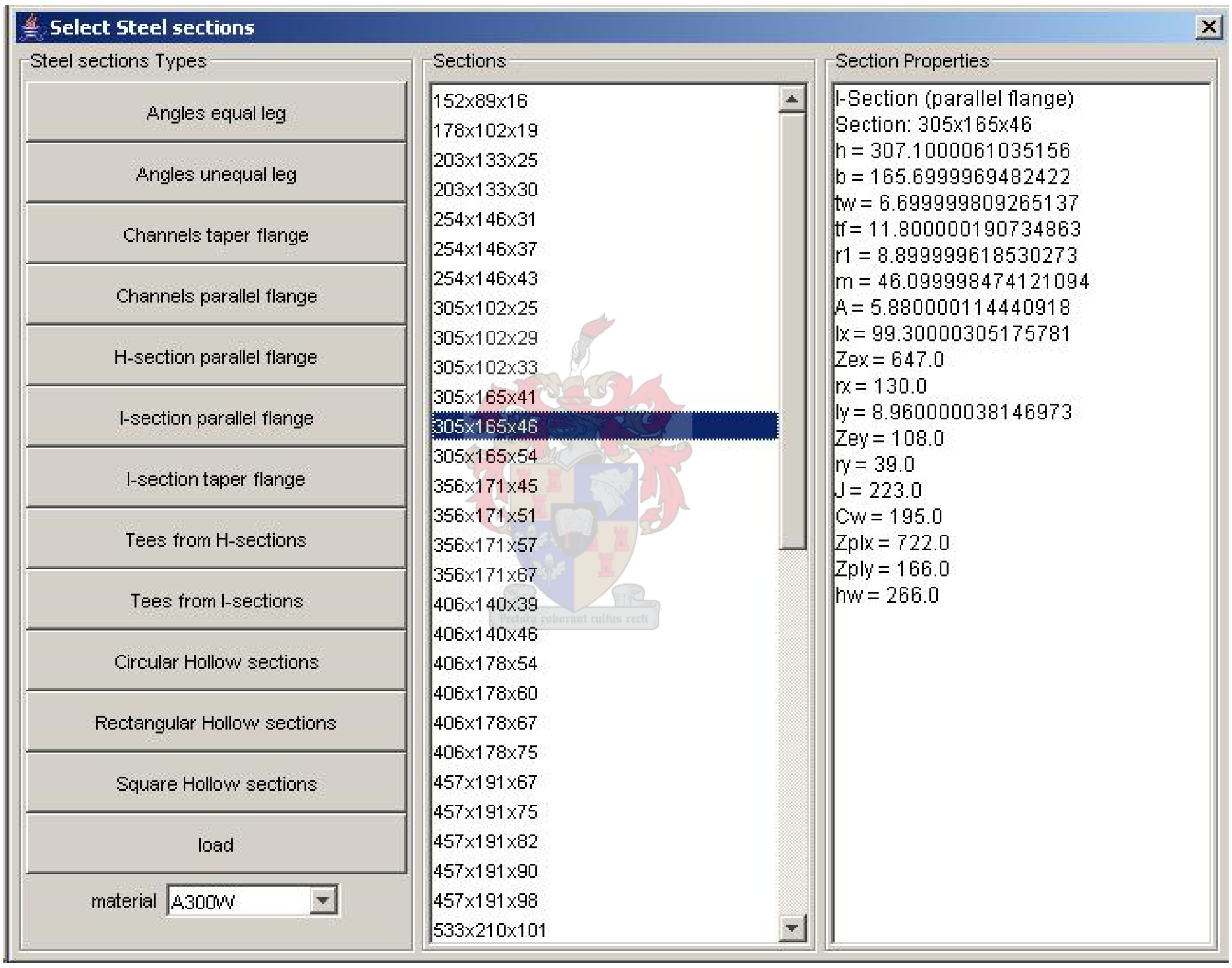
------------end------------

Overall Adequate Profiles
203x203x46

Figure 8-28 Calc/Data sheet for a Beam design

8.5.3.9 `SProfileLoader`

An object of class `SProfileLoader` provides graphically the ability for the user to load steel profiles into the application for the design process. The user has the option to load an entire selection of a specific type of profile or individually select the profiles that are to be used.
The object consists of three parts. They are the buttons with references to the names of all the steel profile types that are available, a list that displays all the profile names that belong to the specific type as well a text field that displays all the dimensions and sectional properties of the selected steel profile. This is illustrated in Figure 8-29



```
152x89x16
178x102x19
203x133x25
203x133x30
254x146x31
254x146x37
254x146x43
305x102x25
305x102x29
305x102x33
305x165x41
305x165x46
305x165x54
356x171x45
356x171x51
356x171x57
356x171x67
406x140x39
406x140x46
406x178x54
406x178x60
406x178x67
406x178x75
457x191x67
457x191x75
457x191x82
457x191x90
457x191x98
533x210x101
```

```
I-Section (parallel flange)
Section: 305x165x46
h = 307.1000061035156
b = 165.6999969482422
tw = 6.699999809265137
tf = 11.800000190734863
r1 = 8.899999618530273
m = 46.099998474121094
A = 5.880000114440918
Ix = 99.30000305175781
Zex = 647.0
rx = 130.0
Iy = 8.960000038146973
Zey = 108.0
ry = 39.0
J = 223.0
Cw = 195.0
Zplx = 722.0
Zply = 166.0
hw = 266.0
```

Figure 8-29 Steel profile loader

8.6 3D Graphics

The 3D graphics package contains all the classes for constructing a 3D rendering of a steel member. These classes include 3D component classes, 3D Utility classes and 3D steel structure classes. The following subsections will briefly describe each of these classes. A more detailed description of each class is provided with the Java documentation of the application.

8.6.1 3D Component Classes

The 3D component classes provide the 3D rendering of the components of a basic steel member. This does not include end conditions but only the member itself. These components include all the included steel profiles in the application e.g. I profile, T profile and Channel profile.
The 3D rendering of a component is represented by surfaces that are rendered by their surface normals. The 3D objects are not solid models. This is how the 3D modelling is managed in Java. The displayed surfaces are defined by the perimeter points that describe the layout of the surface. The direction of the surface points defines the directions of the surface normals, which in turn describe the orientated visibility of the surfaces.

All the component classes extend class `Object3D` and provide their corresponding objects with inherited functionality. A further specialization of class `Object3D` is class `Profile3D` which provides their implementing classes with additional functionality.

The remainder of this section will briefly discuss class `Object3D`, `Profile3D` as well as all the remaining component classes.

8.6.1.1 `Object3D`

Class `Object3D` extends class `TransformGroup` and provides a basic 3D object that has position and orientation.
Important attributes of an object of class `Object3D`:

- `protected Point3f position;`

  The current position of the object in 3D space.
- `protected TransformGroup orientationTransformGroup;`

  The TransformGroup object concerned with orientating the object in 3D space.

Important methods implemented in class Object3D:

- `public void rotateX(float angle)`

  This method rotates the object in a counter clockwise direction with the given angle (in radians) about the x – axis.
- `public void rotateY(float angle)`

  This method rotates the object in a counter clockwise direction with the given angle (in radians) about the y – axis.
- `public void rotateZ(float angle)`

  This method rotates the object in a counter clockwise direction with the given angle (in radians) about the z – axis.
- `public void setPosition(Point3f _position)`

  This method sets the current position of the object equal to a given position.

8.6.1.2 Profile3D

Class `Profile3D` extends class `Object3D` and provides the functionality for rendering a basic steel profile in three dimensions with different sizes and positions.
Important attributes of an object of class `Profile3D`:

- `private float scale;`

  The scale factor of the physical steel profile to the rendered 3D universe
- `private Shape3D shape;`

  The three dimensional shape object that represents the steel profile and contains all the surface points of the steel profile.
- `private Vector3f depthVector;`

  The depth vector that describes the length or depth of the steel profile.

There are only two methods in class Profile3D that need to be implemented by all sub classes, and they are:

- `public abstract void setScale(float scale);`

  This method sets the scale factor of the steel profile to the value provided.
- `public abstract float getScale();`

  This method returns the current scale factor of the steel profile.
- `protected abstract void createShape();`

  This method creates the shape object that represents the steel profile. This method takes all the points that represent the surfaces of the profile and adds them to the shape object.

8.6.1.3 `IH3D`

An object of class `IH3D` represents a 3D I or H steel profile.

Important attributes of an object of class `IH3D` are:

- `private float h;`

  The height of the 3D IH steel profile.

- `private float b;`

  The breadth of the 3D IH steel profile.

- `private float tw;`

  The thickness of the web of the 3D IH steel profile.

- `private float tf;`

  The thickness of the flange of the 3D IH steel profile.

- `private float r1;`

  The radius of the fillet welds of the 3D IH steel profile.

8.6.1.4 `Channel3D`

An object of class `Channel3D`represents a 3D Channel steel profile.

Important attributes of an object of class `Channel3D`are:

- `private float h;`

  The height of the 3D Channel steel profile.

- `private float b;`

  The breadth of the 3D Channel steel profile.

- `private float tw;`

  The thickness of the web of the 3D Channel steel profile.

- `private float tf;`

  The thickness of the flanges of the 3D Channel steel profile.

- `private float r1;`

  The radius of the fillet welds of the 3D Channel steel profile.

8.6.1.5 `EAngle3D`

An object of class `EAngle3D` represents a 3D equal leg Angle steel profile.

Important attributes of an object of class `EAngle3D` are:

- `private float h;`

  The height of the 3D equal leg Angle steel profile.

- `private float b;`

  The breadth of the 3D equal leg Angle steel profile.

- `private float t;`

  The thickness of the legs of the 3D equal leg Angle steel profile.

8.6.1.6 UAngle3D

An object of class `UAngle3D` represents a 3D unequal leg Angle steel profile.

Important attributes of an object of class `UAngle3D` are:

- `private float h;`

  The height of the 3D unequal leg Angle steel profile.

- `private float b;`

  The breadth of the 3D unequal leg Angle steel profile.

- `private float t;`

  The thickness of the legs of the 3D unequal leg Angle steel profile.

8.6.1.7 T3D

An object of class `T3D` represents a 3D T steel profile.

Important attributes of an object of class `T3D` are:

- `private float h;`

  The height of the 3D T steel profile.

- `private float b;`

  The breadth of the 3D T steel profile.

- `private float tw;`

  The thickness of the web of the 3D T steel profile.

- `private float tf;`

  The thickness of the flange of the 3D T steel profile.

- `private float r1;`

  The radius of the fillet welds of the 3D T steel profile.

8.6.1.8 Rectangular3D

An object of class `Rectangular3D` represents a 3D structural hollow rectangular steel profile.

Important attributes of an object of class `Rectangular3D` are:

- `private float h;`

  The height of the 3D structural hollow rectangular steel profile.

- `private float b;`

  The breadth of the 3D structural hollow rectangular steel profile.

- `private float t;`

  The thickness of the walls of the 3D structural hollow rectangular steel profile.

8.6.1.9 `Circular3D`

An object of class `Circular3D` represents a 3D structural hollow circular steel profile.

Important attributes of an object of class `Circular3D` are:

- `private float h;`

  The outside diameter of the 3D structural hollow circular steel profile.
- `private float t;`

  The thickness of the walls of the 3D structural hollow circular steel profile.

8.6.2 3D Utility Classes

The utility classes provide methods for setting the appearances of the background of the 3D drawing as well as for the drawing in its entirety. These classes provide methods for facilitating the creation and grouping of the components of the steel members. The remainder of these sections will briefly describe each utility class.

8.6.2.1 `Utility3D`

Class `Utility3D` contains static methods for creating and altering 3D objects as well as their appearance.

The methods implemented in class `Utility3D` are:

- `public static GeometryArray getExtrudedGeometryArray(...)`

  This method extrudes an array of coordinates along an extrusion path vector and returns a `GeometryArray` object with normals. The array of coordinates represents the front face of the shape
- `public static GeometryArray getHollowExtrudedGeometryArray(...)`

  This method extrudes an array of coordinates along an extrusion path vector and returns a `GeometryArray` object with normals. The array of coordinates represents the front face of the shape. The difference lies in the fact that the front face has a hole in its surface and due to the fact that Java does not represent its 3D models as solid models, a different extrusion procedure is required.
- `public static Appearance getFilledAppearance(Color3f _color)`

  This method returns an `Appearance` object with specific material attributes and a solid filled appearance. The material attributes define the appearance of the object under illumination.

8.6.2.2 `Structure3D`

This class provides the functionality to orientate the steel members, that from part of a structure, accordingly to the rendered 3D environment. This class consists of only two static methods that allow for the rendering of all the steel elements in their correct positions and orientation. The methods of class Structure3D are as follows:

- `public static void orientateMember(SSDesignElement del, ...)`

  This method orientates the steel member in 3D space that is to be rendered.
- `public static float getInPlaneRotation(IOneDDElement del)`

  This method returns the in plane rotation of the element in radians.

9 Verification

Examples of the main implemented design types are provided to illustrate and verify the design output and results of the application.

Each example is divided into three sections. The first section is the problem statement. This section concerns the loaded finite element model and illustrates its composition. This provides the basis for the geometry of the members as well as the concerned internal forces and external loading. The finite element model's 2D view along with relevant force diagrams are shown in this section. The second section includes all the design parameters to complete a design on the steel members. This includes differences in steel member topology from the finite elements, the profiles that are to be used, internal restraints, external restraints, member bracing, rotation of the steel profiles as well as all the important internal and external force values. The corresponding 2D and 3D views of the application in this state are shown. The third and final section is concerned with the hand calculations according to the specifications described in chapter 4. The results are compared to values yielded by the application, which are shown in between square brackets.

9.1 Beams

9.1.1 Example 1 – Simply supported beam

Finite element model: The finite element model for example 1, illustrated in Figure 9-1, is as follows:

The finite elements model consists of two Euler beam elements joined together to from one continuous beam element. The left and right end conditions of the finite elements have the translational degree of freedom fixed with the rotational degree of freedom unrestricted. The loading on the beam is a 50 kN load at it mid point.

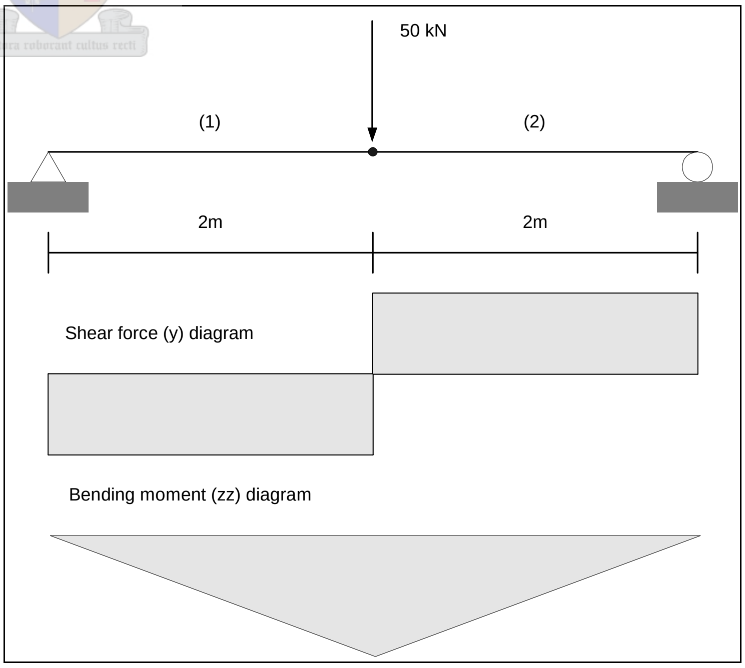


Figure 9-1 Finite element model for example 1

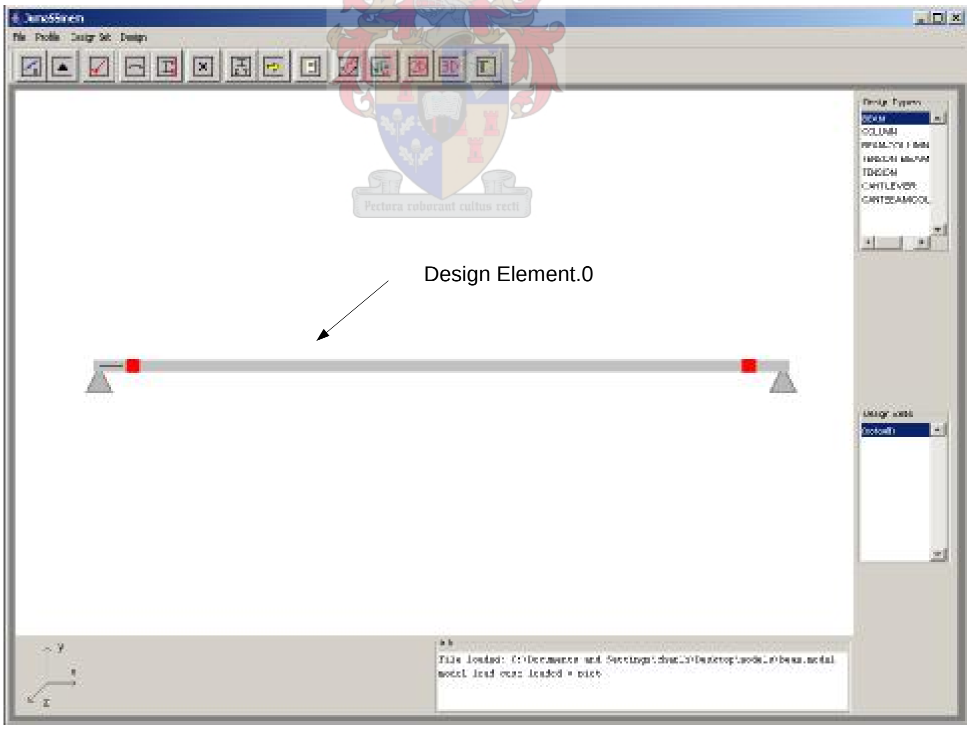
Application design layout: The layout used to portray the parameters and objects used by the application is described. This layout remains the same for all the examples included in this chapter. The layout used is as follows:

The number of Design Elements is listed along with their respective names and lengths. The number of Design Sets is then listed. Each Design Element belonging to a specific Design Set is indicated next to the Design Set names. The profiles used in the design for each Design Set are shown with their respective descriptions. Each Design Element's end forces and maximum internal loading that is relevant to the particular design, as calculated by the application, is listed. Finally, the restraint conditions applied to the ends of the individual Design Elements are listed.

Application design input: In this section the design input and results of internal forces and end forces are shown:

- Design Elements = 1
  - "DesignElement.0"
  - length = = 4m

The Design Element(s) are illustrated in Figure 9-2



```
File loaded: C:\Documents and Settings\[illegible]\Desktop\models\beam.model
model load case loaded = [illegible]
```

Figure 9-2 Illustration of the Design Elements of example 1

- Design Sets = 1
  - "default" → "DesignElement.0"

- Loaded profiles: (Grade 300W) for Design Set “default”.

  “152x89x16”

  “457x191x67”

- “DesignElement.0 ” End Forces (start)

Axial Force	= 0.0 kN
Shear Force (y)	= -25 kN
Moments (zz)	= 0.0 kN.m

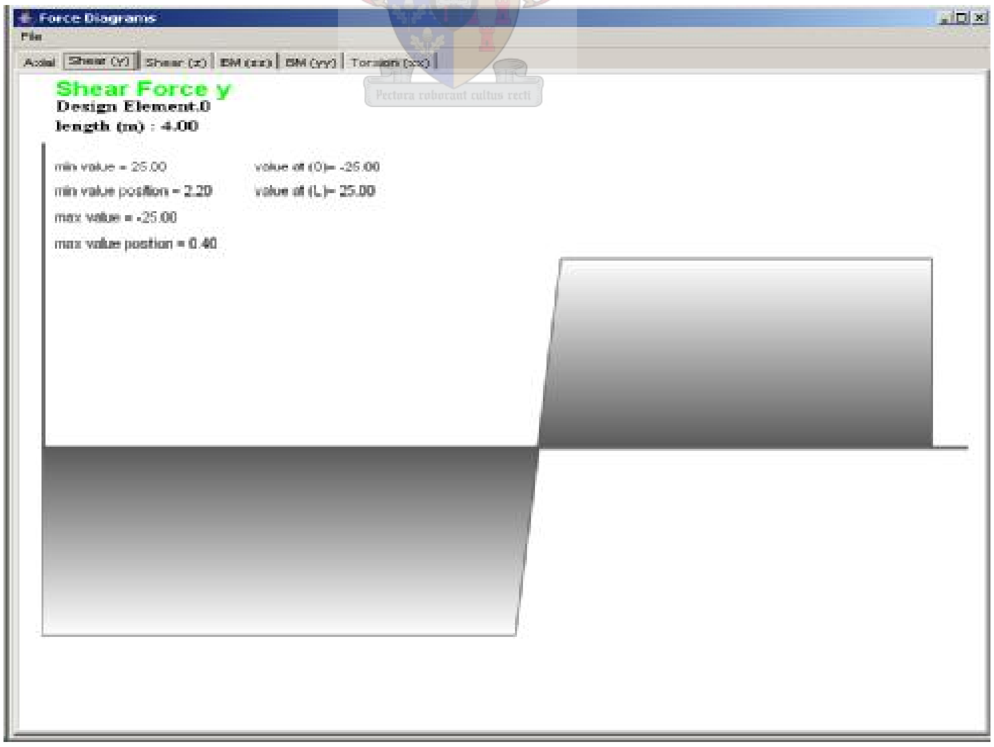
- “DesignElement.0” End Forces (end)

Axial Force	= 0.0 kN
Shear Force (y)	= 25 kN
Moments (zz)	= 0.0 kN.m

- “DesignElement.0” Ultimate Forces

Axial Force	= 0.0 kN
Shear Force (y)	= 25 kN
Moments (zz)	= 50 kN.m

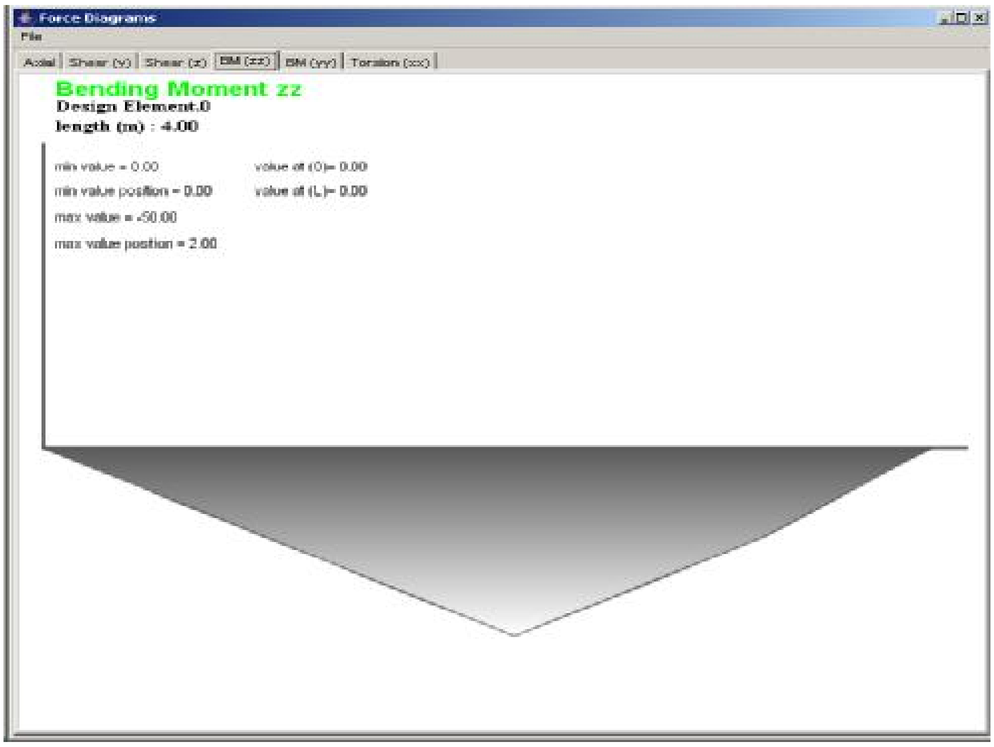
The important internal force diagrams of DesignElement.0 in example 1 are illustrated in Figure 9-3 and Figure 9-4. These diagrams include the Shear Force diagram as well as the Bending Moment diagram as these are the only none zero diagrams.



Shear Force y
Design Element.0
length (m) : 4.00

```
min value = 25.00              value at (0)= -25.00
min value position = 2.20      value at (L)= 25.00
max value = -25.00
max value postion = 0.40
```

Figure 9-3 Shear Force diagram for example 1



Bending Moment zz
Design Element.0
length (m) : 4.00

```
min value = 0.00                value at (0)= 0.00
min value position = 0.00       value at (L)= 0.00
max value = -50.00
max value postion = 2.00
```

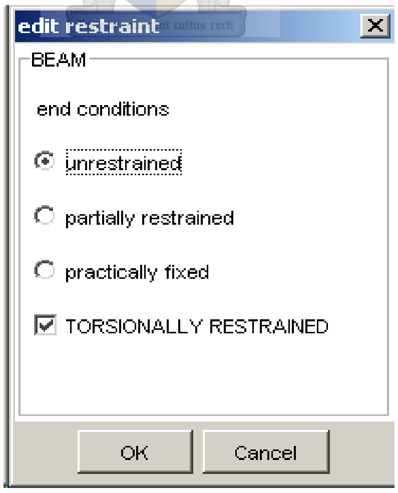
Figure 9-4 Bending Moment diagram for example 1

- “DesignElement.0” Restraint Conditions (start)

  Unrestrained (free to rotate about the vertical plane through the support)

  Restrained against torsion

The application of such restraint conditions is illustrated in Figure 9-5.



BEAM

end conditions

[✓ unrestrained]

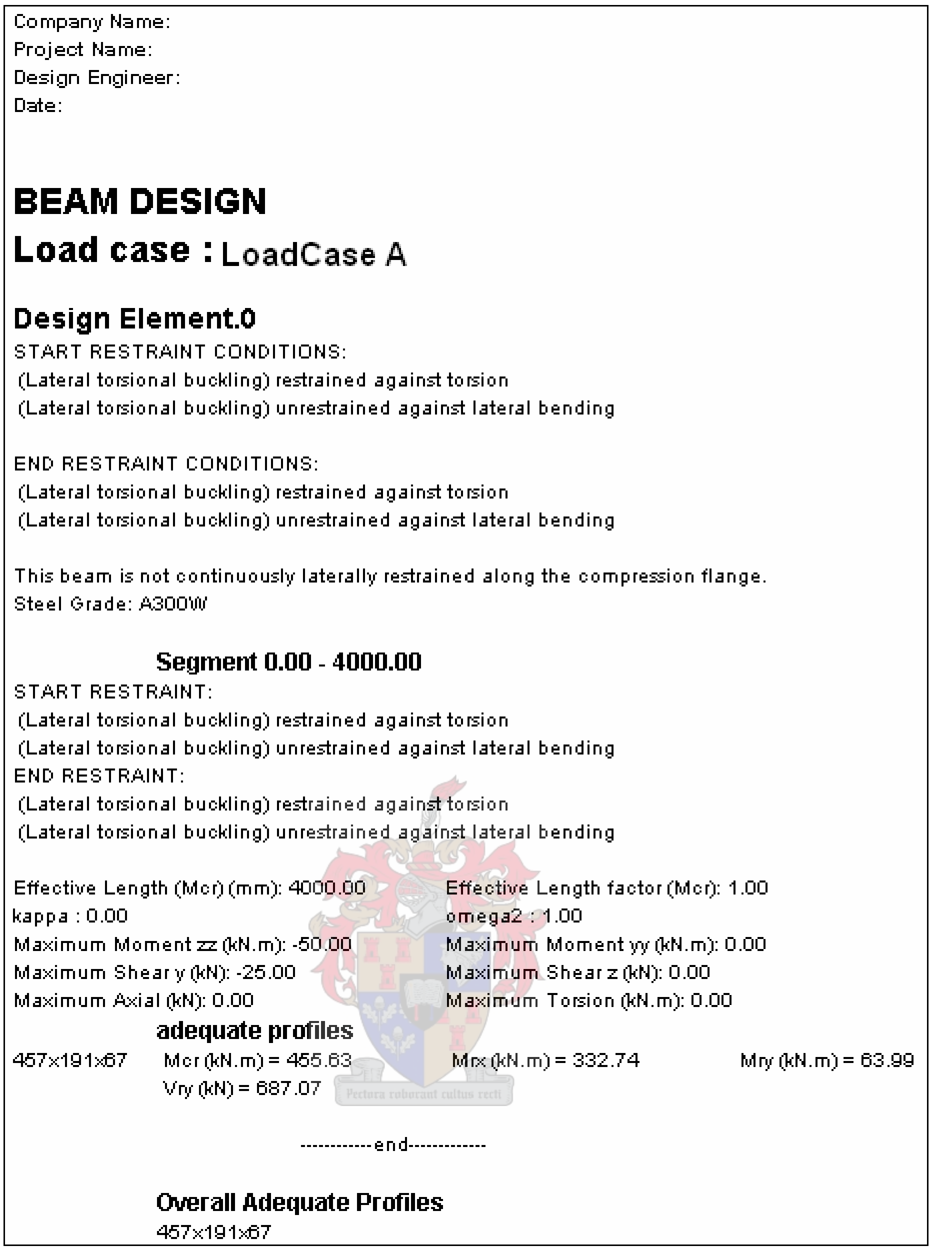
partially restrained

practically fixed

[✓ TORSIONALLY RESTRAINED]

Figure 9-5 Illustrating the restraint conditions for example 1

The loading on the member is assumed to be stabilized.



```
Company Name:
Project Name:
Design Engineer:
Date:
```

BEAM DESIGN

Load case : LoadCase A

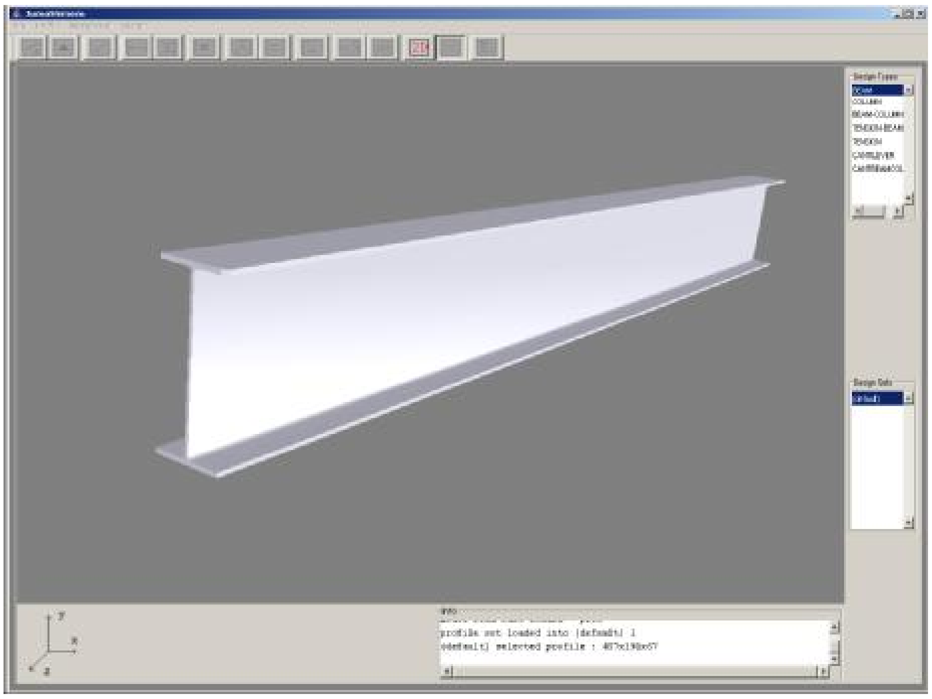
Design Element.0

```
START RESTRAINT CONDITIONS:
 (Lateral torsional buckling) restrained against torsion
 (Lateral torsional buckling) unrestrained against lateral bending

END RESTRAINT CONDITIONS:
 (Lateral torsional buckling) restrained against torsion
 (Lateral torsional buckling) unrestrained against lateral bending

This beam is not continuously laterally restrained along the compression flange.
Steel Grade: A300W

                Segment 0.00 - 4000.00
START RESTRAINT:
 (Lateral torsional buckling) restrained against torsion
 (Lateral torsional buckling) unrestrained against lateral bending
END RESTRAINT:
 (Lateral torsional buckling) restrained against torsion
 (Lateral torsional buckling) unrestrained against lateral bending

Effective Length (Mcr) (mm): 4000.00           Effective Length factor (Mcr): 1.00
kappa : 0.00                                   omega2 : 1.00
Maximum Moment zz (kN.m): -50.00               Maximum Moment yy (kN.m): 0.00
Maximum Shear y (kN): -25.00                   Maximum Shear z (kN): 0.00
Maximum Axial (kN): 0.00                       Maximum Torsion (kN.m): 0.00
                adequate profiles
457x191x67       Mcr (kN.m) = 455.63           Mrx (kN.m) = 332.74          Mry (kN.m) = 63.99
                 Vry (kN) = 687.07

                        ------------end------------

                Overall Adequate Profiles
                457x191x67
```

Figure 9-6 Text results of the design for example 1

Figure 9-6 illustrates the text results of the design for example 1 as illustrated by the "Design Data Sheet" of the application.

The adequate steel profiles to be used as steel members are calculated by the application to be "457x191x67". Figure 9-7 illustrates the 3D view of the "457x191x67" steel profile.



```
profile set loaded into [default] 1
[default] selected profile : 457x191x67
```

Figure 9-7 Illustration of the 3D view for example 1

Hand calculated design: This section shows the hand calculated design results and comparison with the results from the application.

<u>Classification of profiles</u>

“152x89x16”

$h = 152.4$ mm

$b = 88.9$ mm

$t_f = 7.7$ mm

$t_w = 4.6$ mm

$$\frac{b}{2 \cdot t_f} = \frac{88.9}{2 \cdot 7.7} = 5.77 \leq \frac{145}{\sqrt{300}} \rightarrow \text{Class 1 flange}$$

$$\frac{h - 2t_f}{t_w} = \frac{152.4 - 2 \cdot 7.7}{4.6} = 29.78 \leq \frac{1100}{\sqrt{300}} \rightarrow \text{Class 1 web}$$

The profile is therefore an overall class 1 profile

“457x191x67”

$h = 453.6$ mm

$b = 189.9$ mm

$t_f = 12.7$mm

$t_w = 8.5$mm

$$\frac{b}{2 \cdot t_f} = \frac{189.9}{2 \cdot 12.7} = 7.47 \leq \frac{145}{\sqrt{300}} \longrightarrow \text{class 1 flange}$$

$$\frac{h - 2t_f}{t_w} = \frac{152.4 - 2 \cdot 7.7}{4.6} = 29.78 \leq \frac{1100}{\sqrt{300}} \longrightarrow \text{class 1 web}$$

The profile is therefore an overall class 1 profile

<u>DesignElement.0</u>

The element is not continuously braced against lateral buckling and thus the lateral torsional buckling strength as well as the cross sectional strength of the member needs to be tested. The cross sectional strong axis bending resistance of the two steel profiles are as follows:

"152x89x16" $M_r = \Phi Z_{pl} f_y = 0.9(124000)(300) = 33.48\,\text{kN.m} < 50\ \text{kN.m}$ FAIL.

"457x191x67" $M_r = \Phi Z_{pl} f_y = 0.9(1470000)(300) = 396.7\,\text{kN.m} > 50\ \text{kN.m}$ O.K.

The end moments of the steel element are both zero, with a single point load acting on the beam of 50 kN. The calculation of the ω_2 and κ values is as follows.

$\kappa = 0.0$ [0.0]

ω_2 = 1.0 (internal moment larger than the end moments) [1.0]

The critical moment for the steel member is now calculated. Due to the fact that the "152x89x16" profile has failed already due to strength, its lateral torsional buckling capacity will not be calculated here. This steel profile is also not shown in the applications design test results in Figure 9-6 as the profile is inadequate in terms of strength. Only the "457x191x67" is considered further.

The effective length factor of the steel element is 1.0, due to the restraint conditions applied. The critical moment is calculated as follows:

$$M_{cr} = \frac{\omega_2 \cdot \pi}{K \cdot L}\sqrt{EI_y GJ + \left(\frac{\pi E}{K \cdot L}\right)^2 I_y C_w}$$

$$\therefore \frac{1.0(\pi)}{1.0(4000)}\sqrt{200\times10^3(14.5\times10^6)(77\times10^3)(376\times10^3) + \left(\frac{(\pi)(200\times10^3)}{(1.0)4000}\right)^2 (14.5\times10^6)(706\times10^9)}$$

$\therefore M_{cr} = 455.63$ kN.m [455.63 kN.m]

This value is larger than the cross sectional bending strength of the profile's cross section that was calculated as 396.7 kN.m. The bending resistance of the steel member is thus calculated as follows:

$$M_r = 1.15 \cdot \Phi \cdot M_p \cdot \left(1 - \frac{0.28 \cdot M_p}{M_{cr}}\right) \leq \Phi \cdot M_p$$

$$\therefore = 1.15(0.9)(440.78)\left(1 - \frac{0.28(440.78)}{455.63}\right) = 332.63 \text{ kN.m} < 369.7 \text{ kN.m}$$

$$\therefore M_r = 332.63 \text{ kN.m} \qquad [332.74 \text{ kN.m}]$$

The shear resistance of the beam is calculated as follows:

$$V_r = \phi \cdot A_v \cdot f_s$$

$$A_v = t_w \cdot h = (8.5)453.6 = 3855.6\text{mm}^2$$

$$f_s = 0.66 \cdot f_y = (0.66)300 = 198\text{MPa}$$

$$V_r = (0.9)(3855.6)(198) = 687.07\text{kN} > 25\text{kN} \qquad [687.07 \text{ kN}]$$

The steel profile "457x191x67" as used under the conditions prescribed by example 1 is adequate in terms of ultimate strength.

9.1.2 Example 2 – Simply supported beam (continued)

The finite element model is exactly the same as in Example 1. This is illustrated in Figure 9-1

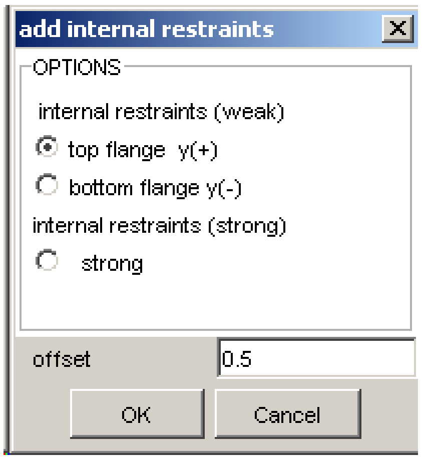
The design makeup and inputs are in majority the same as in Example 1 except for two changes.

(1) The loaded steel profile is a "305x102x25" grade 300W profile.

(2) An internal lateral restraint is applied at midspan to the top flange of the beam.

The application of the internal restraint is illustrated in Figure 9-8.

The 2D view of the steel member is illustrated in Figure 9-9.


add internal restraints

OPTIONS

internal restraints (weak)

[✓ top flange y(+)]

bottom flange y(-)

internal restraints (strong)

strong

offset 0.5

Figure 9-8 Illustrating the addition of an internal restraint

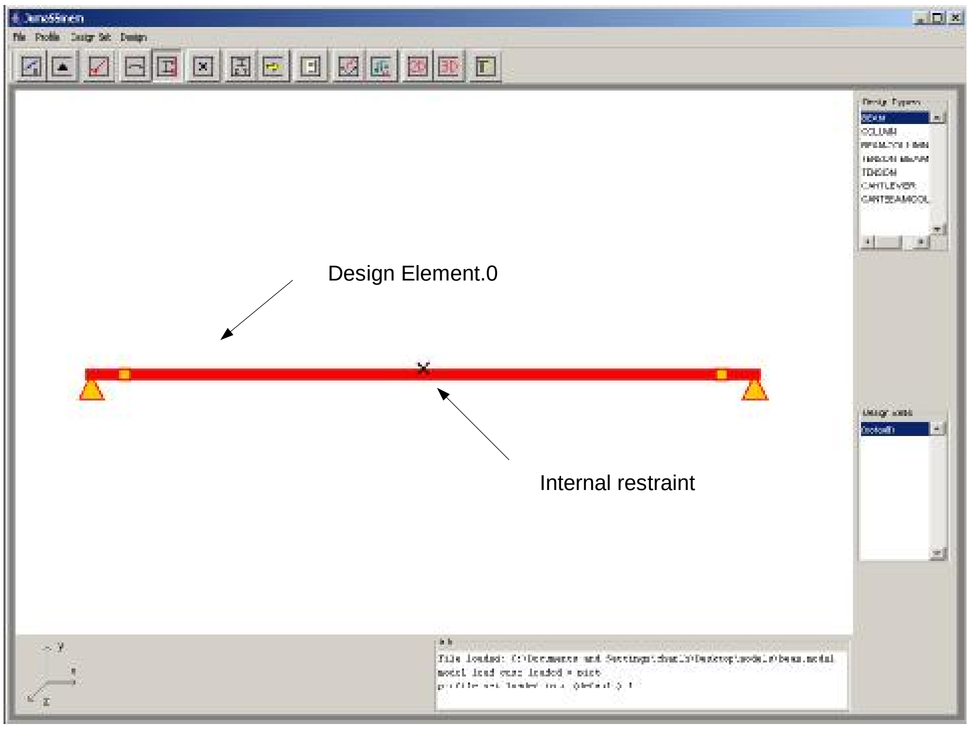
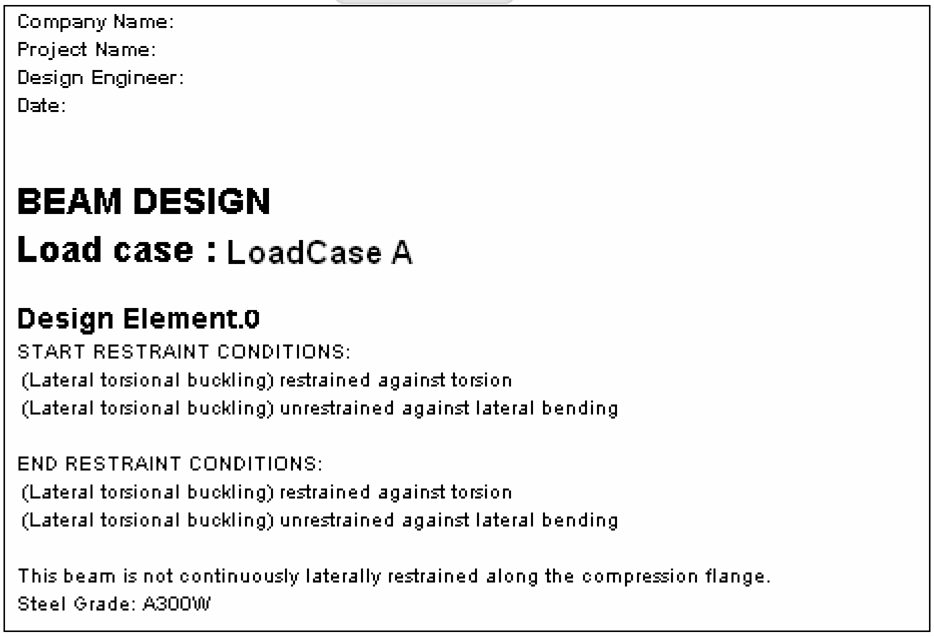


Figure 9-9 Illustration of the Design Elements in example 2

The important internal force diagrams of example 2 are identical to those of example 1. They are illustrated in Figure 9-3 and Figure 9-4.

The restraint conditions at the ends of the steel beam are the same as in example 1.

The text results for example 2 are illustrated in Figure 9-10.



Company Name:
Project Name:
Design Engineer:
Date:

BEAM DESIGN

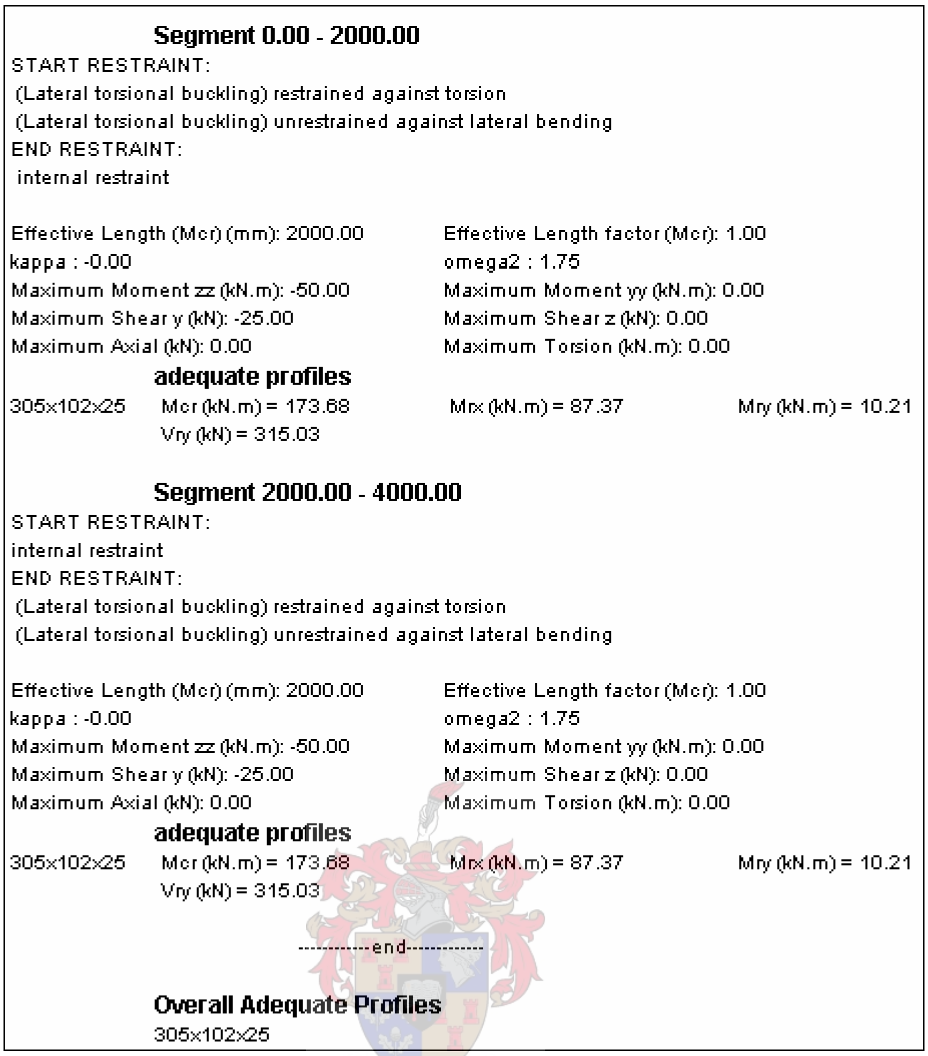
Load case : LoadCase A

Design Element.0

START RESTRAINT CONDITIONS:
(Lateral torsional buckling) restrained against torsion
(Lateral torsional buckling) unrestrained against lateral bending

END RESTRAINT CONDITIONS:
(Lateral torsional buckling) restrained against torsion
(Lateral torsional buckling) unrestrained against lateral bending

This beam is not continuously laterally restrained along the compression flange.
Steel Grade: A300W



```
                Segment 0.00 - 2000.00
START RESTRAINT:
 (Lateral torsional buckling) restrained against torsion
 (Lateral torsional buckling) unrestrained against lateral bending
END RESTRAINT:
 internal restraint

Effective Length (Mcr) (mm): 2000.00          Effective Length factor (Mcr): 1.00
kappa : -0.00                                 omega2 : 1.75
Maximum Moment zz (kN.m): -50.00              Maximum Moment yy (kN.m): 0.00
Maximum Shear y (kN): -25.00                  Maximum Shear z (kN): 0.00
Maximum Axial (kN): 0.00                      Maximum Torsion (kN.m): 0.00
                adequate profiles
305x102x25      Mcr (kN.m) = 173.68            Mrx (kN.m) = 87.37            Mry (kN.m) = 10.21
                Vry (kN) = 315.03

                Segment 2000.00 - 4000.00
START RESTRAINT:
internal restraint
END RESTRAINT:
 (Lateral torsional buckling) restrained against torsion
 (Lateral torsional buckling) unrestrained against lateral bending

Effective Length (Mcr) (mm): 2000.00          Effective Length factor (Mcr): 1.00
kappa : -0.00                                 omega2 : 1.75
Maximum Moment zz (kN.m): -50.00              Maximum Moment yy (kN.m): 0.00
Maximum Shear y (kN): -25.00                  Maximum Shear z (kN): 0.00
Maximum Axial (kN): 0.00                      Maximum Torsion (kN.m): 0.00
                adequate profiles
305x102x25      Mcr (kN.m) = 173.68            Mrx (kN.m) = 87.37            Mry (kN.m) = 10.21
                Vry (kN) = 315.03

                            ------------end------------

                Overall Adequate Profiles
                305x102x25
```

Figure 9-10 Text design results for example 2

As can be seen from the text results created by the “Design Data Sheet” the steel beam is divided into two segments. These segments represent the two unbraced lengths of the member due to the presence of a lateral internal restraint at a point along its length. The design procedure as described in chapter 4 requires this approach. The segments used in this approach are an example of the “Internal Elements” discussed in chapter 6.

Hand calculated design: This section shows the hand calculated design results and comparison with the results from the application.

<u>Classification of profiles</u>

“305x102x25”

h = 304.8

b = 101.6

t_f = 6.8

$t_w = 5.8$

$$\frac{b}{2 \cdot t_f} = \frac{101.6}{2 \cdot 6.8} = 7.47 \leq \frac{145}{\sqrt{300}} \rightarrow \text{class 1 flange}$$

$$\frac{h - 2t_f}{t_w} = \frac{304.8 - 2 \cdot 6.8}{5.8} = 50.2 \leq \frac{1100}{\sqrt{300}} \rightarrow \text{class 1 web}$$

The profile is therefore an overall class 1 profile.

<u>DesignElement.0</u>

This member is laterally braced at its midspan and is thus broken up into two distinctive parts, namely Segment 1 and Segment 2. These segments are thus designed as individual members each having a length of 2m with the entire member spanning a full 4m.

This is illustrated in Figure 9-11.

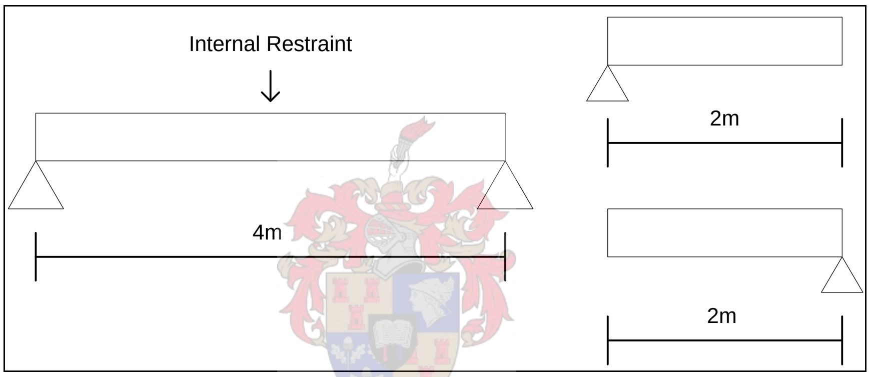


Figure 9-11 Illustration of the segments of a beam in example 2

The cross sectional bending resistance of the chosen profile as well as the lateral torsional buckling strength of the steel member with the chosen profile is tested.

The cross sectional resistance of the steel profile “305x102x25” is as follows:

$M_v = \Phi \cdot Z_{pl} \cdot f_y = 0.9(336000)(300) = 90.72\,\text{kN.m} > 50\ \text{kN.m}$ O.K.

The shear resistance of the beam is identical to that of example 1.

The following calculations are all dependent on the “Segments” of the steel member

<u>DesignElement.0 (0m – 2m)</u>

The end moments of this segment are 0 kN.m and -50 kN.m in turn. The calculation of the κ factor is as follows:

$$\kappa = \frac{0}{-50} = 0 \qquad [0]$$

The ω_2 factor used in the calculation of the critical moment is then calculated.

$$\omega_2 = 1.75 + 1.05 \cdot \kappa + 0.3 \cdot \kappa^2 = 1.75 + 1.05(0) + 0.3(0) = 1.75 \qquad [1.75]$$

The critical moment for this length of steel member is now calculated. The effective length factor, K, is the same as in example 1 due to the end restraint conditions being identical.

$$M_{cr} = \frac{\omega_2 \cdot \pi}{K \cdot L}\sqrt{EI_y GJ + \left(\frac{\pi E}{K \cdot L}\right)^2 I_y C_w}$$

$$\therefore \frac{1.75(\pi)}{1.0(2000)}\sqrt{200\times10^3(1.19\times10^6)(77\times10^3)(48\times10^3) + \left(\frac{(\pi)(200\times10^3)}{(1.0)2000}\right)^2 (1.19\times10^6)(26.5\times10^9)}$$

$\therefore = 173.68$ kN.m [173.68 kN.m]

This value is larger than the value of $0.67M_p$, namely 0.67(90.72) = 67.54 kN.m. The bending resistance of the segment is then calculated as follows:

$$M_r = 1.15 \cdot \Phi \cdot M_p \cdot \left(1 - \frac{0.28 \cdot M_p}{M_{cr}}\right) \le \Phi \cdot M_p$$

$$\therefore = 1.15(0.9)(100.8)\left(1 - \frac{0.28(100.8)}{173.68}\right) = 87.37 \text{ kN.m} < 90.72 \text{ kN.m}$$ [87.37 kN.m]

<u>DesignElement.0 (2m – 4m)</u>

The design of this segment is identical to that of the segment that ranges from 0m to 4m as shown above. This is due to the fact that the entire steel member is symmetrical in its internal loading and the fact that the internal restraint is in the middle of the beam.

9.2 Columns

9.2.1 Example 3 – Simple Column

The finite element for example 3 is briefly described below.

The finite element model consists of two Euler beam elements connected end to end to form a vertical continuous column. The column is loaded by a 50 kN vertical load at its end point in a downwards direction, causing constant compression along the length of the column. The column is supported against horizontal and vertical translation at its start and end. The finite element model is illustrated in Figure 9-12.

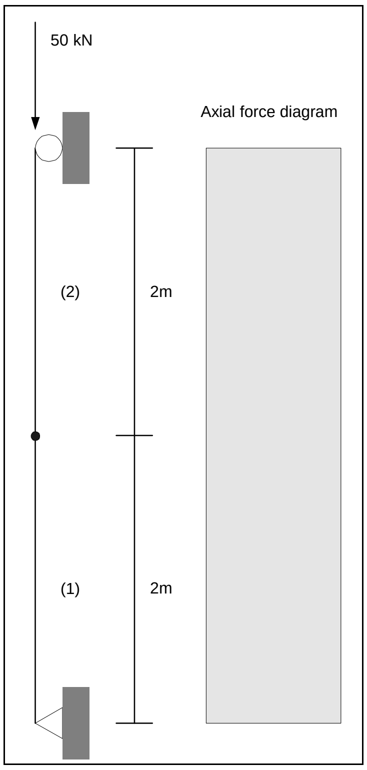
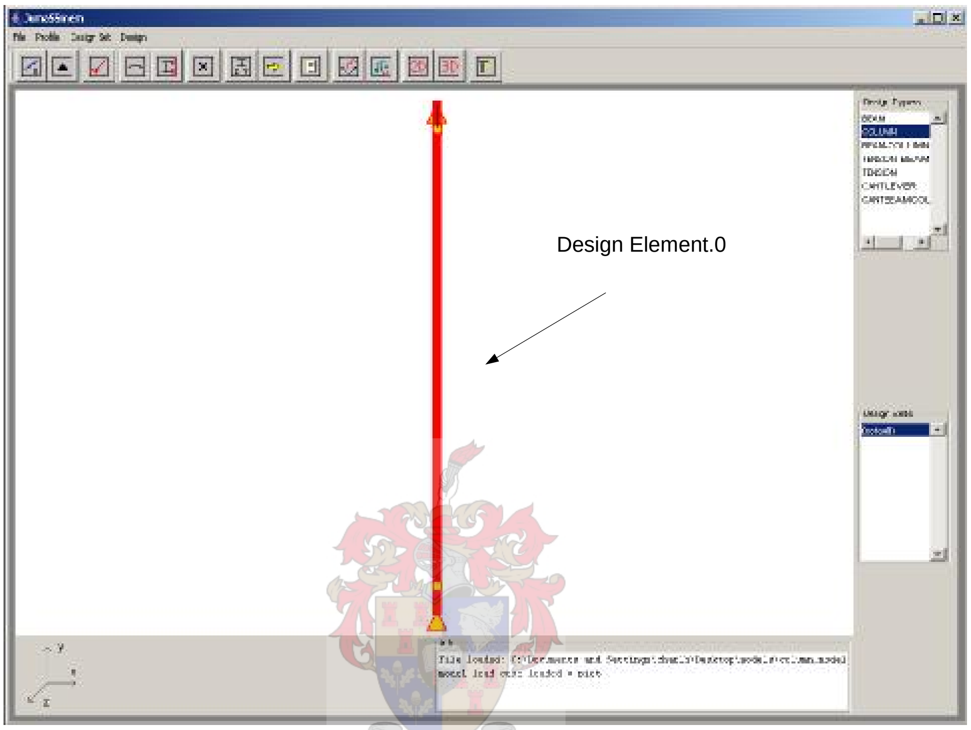


Figure 9-12 Finite element model for example 3

Application design input: In this section the design input and results of internal forces and end forces are shown:

o	Design Elements	= 1
	"DesignElement.0"	
	length	= 4m

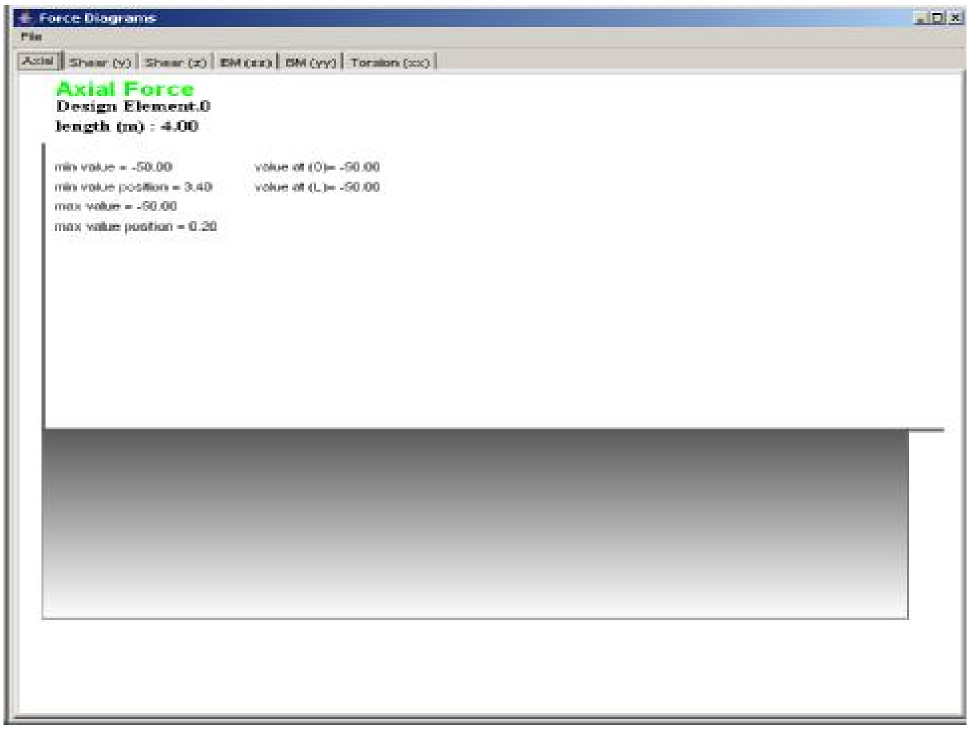
The Design Element(s) are illustrated in Figure 9-13.



```
File loaded: C:\Documents and Settings\[illegible]\Desktop\models\column.model
model load case loaded = [illegible]
```

Figure 9-13 Illustration of the Design Elements of example 3

o	Design Sets	= 1
	"default"	→ "DesignElement.0"
o	Loaded profiles: Grade 300W for Design Set "default"	
	"PFC 160x65" "Channel (parallel flange)"	
o	"DesignElement.0" Ultimate Forces	
	Axial Force	= -50kN (compressive)
	Shear Force (y)	= 0.0 kN
	Moments(zz)	= 0.0 kN



Axial Force
Design Element.0
length (m) : 4.00

min value = -50.00
min value position = 3.40
max value = -50.00
max value position = 0.20
value at (0)= -50.00
value at (L)= -50.00

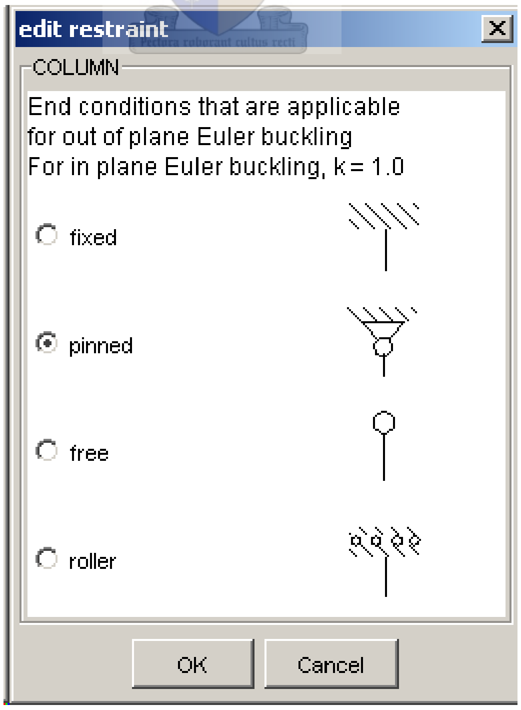
Figure 9-14 Axial force diagram for example 3

Figure 9-14 illustrates the only non-zero internal force diagram, namely the axial force diagram.

- “DesignElement.0” Restraint conditions

  Pinned (translation fixed with respect to out of plane buckling)

The application of such restraint conditions is illustrated in Figure 9-15.

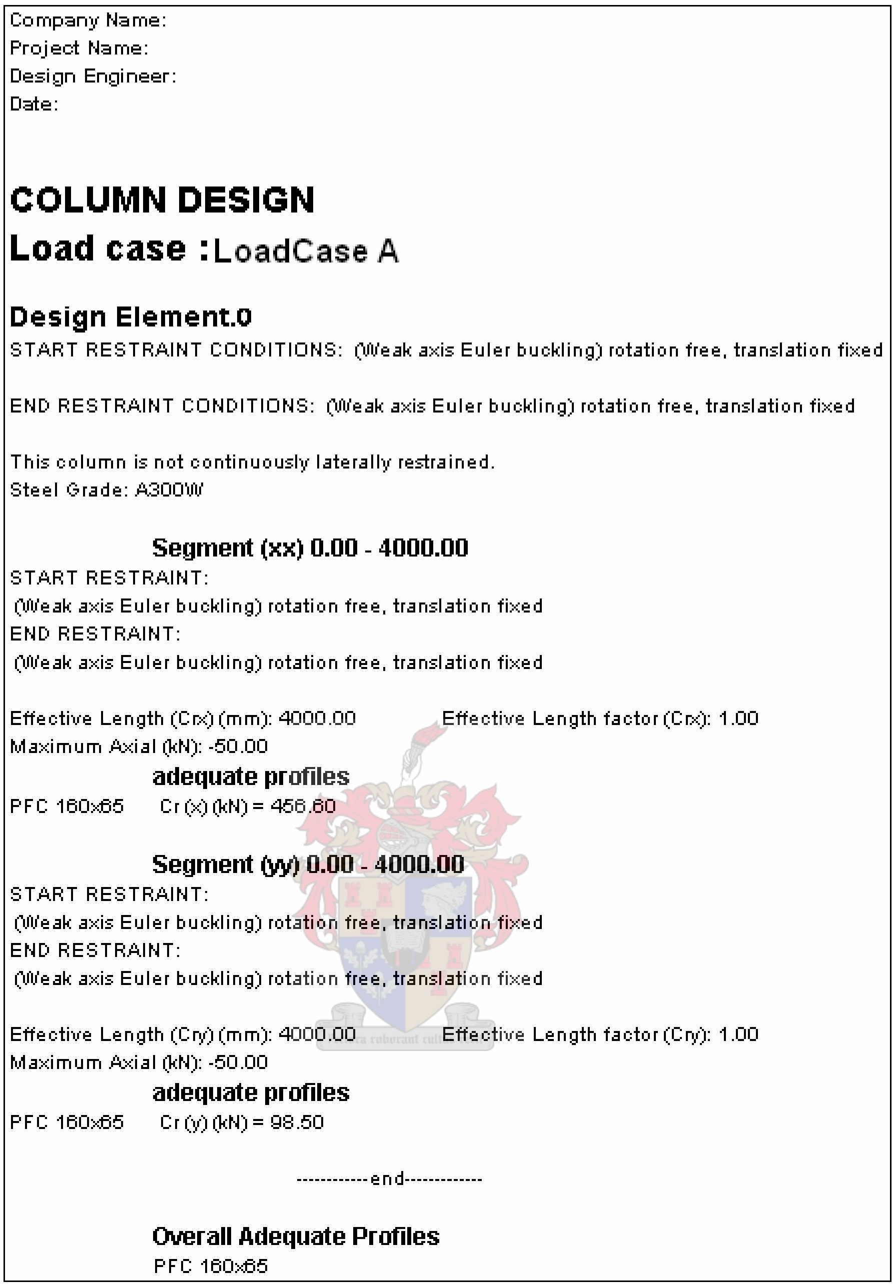


End conditions that are applicable
for out of plane Euler buckling
For in plane Euler buckling, k = 1.0

fixed [✓ pinned] free roller

Figure 9-15 Illustrating the restraint conditions for example 3

The text design results as generated by the "Design Data Sheet" are illustrated in Figure 9-16.



```
Company Name:
Project Name:
Design Engineer:
Date:

COLUMN DESIGN
Load case :LoadCase A

Design Element.0
START RESTRAINT CONDITIONS:  (Weak axis Euler buckling) rotation free, translation fixed

END RESTRAINT CONDITIONS:  (Weak axis Euler buckling) rotation free, translation fixed

This column is not continuously laterally restrained.
Steel Grade: A300W

                    Segment (xx) 0.00 - 4000.00
START RESTRAINT:
 (Weak axis Euler buckling) rotation free, translation fixed
END RESTRAINT:
 (Weak axis Euler buckling) rotation free, translation fixed

Effective Length (Crx) (mm): 4000.00           Effective Length factor (Crx): 1.00
Maximum Axial (kN): -50.00
                    adequate profiles
PFC 160x65     Cr (x) (kN) = 456.60

                    Segment (yy) 0.00 - 4000.00
START RESTRAINT:
 (Weak axis Euler buckling) rotation free, translation fixed
END RESTRAINT:
 (Weak axis Euler buckling) rotation free, translation fixed

Effective Length (Cry) (mm): 4000.00           Effective Length factor (Cry): 1.00
Maximum Axial (kN): -50.00
                    adequate profiles
PFC 160x65     Cr (y) (kN) = 98.50

                              ------------end------------

                    Overall Adequate Profiles
                    PFC 160x65
```

Figure 9-16 Text results of the design for example 3

As indicated by the text results, the chosen steel profile is adequate in terms of strength.

Hand calculated design: This section shows the hand calculated design results and comparison with the results from the application.

<u>Classification of profiles</u>

"PFC 160x65"

h = 160 mm

b = 65 mm

$t_w = 6.5$

$t_f = 10.4$

$$\frac{b}{t_f} = \frac{65}{10.4} = 6.25 < \frac{200}{\sqrt{300}}$$

The profile is thus at most a class 3 profile.

DesignElement.0

The element is free to buckle about either its strong or weak axis due to the fact that there is no lateral bracing present along its length. The profile that was chosen to represent the member has a single axis of symmetry resulting in the compressive resistance of the member to be based on either torsional or torsional flexural buckling compressive resistance. For the purpose of this example the buckling resistances about both axes are calculated, although it is apparent that buckling resistance about the weaker axis would be governing.

The effective length factor calculated from the restraint conditions described earlier in this example is 1.0. [1.0]

The calculation of the compressive resistance is as follows:

Buckling about the strong or x – axis

$$\frac{K \cdot L_x}{r_x} = \frac{1.0(4000)}{63.4} = 63.09 < 200$$ O.K.

$$\therefore \lambda = \frac{K \cdot L_x}{r_x}\sqrt{\frac{f_y}{\pi^2 \cdot E}} = 63.09\sqrt{\frac{300}{\pi^2(200 \times 10^3)}} = 0.77778$$

$$C_{rx} = \Phi \cdot A \cdot f_y (1 + \lambda^{2n})^{\frac{-1}{n}}$$

The value of n is taken as 1.34 for hot rolled, fabricated steel sections.

$$\therefore C_{rx} = 0.9(2.3 \times 10^3)(300)(1 + 0.77778^{2(1.34)})^{\frac{-1}{1.34}} = 456.6\text{kN}$$ [456.6 kN]

Buckling about the weak or y – axis.

$$\frac{K \cdot L_y}{r_y} = \frac{1.0(4000)}{20.3} = 197.04 < 200$$ O.K.

The compressive resistance is calculated by using the lesser of the two values f_{ey} and f_{exz}.

$$f_{ey} = \frac{\pi^2 \cdot E}{\left(\dfrac{K_y \cdot L_y}{r_y}\right)^2} = \frac{\pi^2 (200\times10^3)}{197.04^2} = 50.84$$

$$f_{ex} = \frac{\pi^2 \cdot E}{\left(\dfrac{K_x \cdot L_x}{r_x}\right)^2} = \frac{\pi^2 (200\times10^3)}{\left(\dfrac{1.0(4000)}{63.4}\right)^2} = 495.89$$

$$f_{ez} = \left(\frac{\pi^2 \cdot E \cdot C_w}{K_z{}^2 \cdot L_z{}^2} + G \cdot J\right) \cdot \frac{1}{A \cdot \bar{r}_0{}^2}$$

$$\bar{r}_0{}^2 = x_0{}^2 + y_0{}^2 + r_x{}^2 + r_y{}^2 = 6.33^2 + 0^2 + 63.4^2 + 20.3^2 = 4471.7$$

$$\therefore f_{ez} = \left(\frac{\pi^2 (200\times10^3)(3.82\times10^9)}{1.0^2 (4000)^2} + 77\times10^3 (64.4\times10^3)\right) \cdot \left(\frac{1}{2.3\times10^3 (4471.4)}\right)$$

$$\therefore f_{ez} = 528.0$$

$$f_{exz} = \frac{f_{ex} + f_{ez}}{2 \cdot \Omega}\left[1 - \sqrt{1 - \frac{4 \cdot f_{ex} \cdot f_{ez} \cdot \Omega}{(f_{ex} + f_{ez})^2}}\right]$$

$$\Omega = 1 - \left(\frac{x_0{}^2 + y_0{}^2}{\bar{r}_0{}^2}\right) = 1 - \left(\frac{6.33^2 + 0^2}{4471.7^2}\right) = 0.9999$$

$$\therefore f_{exz} = \frac{(495.89 + 528.0)}{2(0.9999)}\left[1 - \sqrt{1 - \frac{4(495.89)(528.0)(0.9999)}{(495.89 + 528.0)^2}}\right] = 2117.47$$

The summarized results are:

$$f_{ex} = 495.89 \because f_{ey} = 50.84 \because f_{exz} = 2117.47$$

As can be seen from the results above, the governing value is f_{ey}. The compressive resistance for weak axis buckling is then calculated according to this value.

$$\therefore \lambda = \sqrt{\frac{f_y}{f_{ey}}} = \sqrt{\frac{300}{50.84}} = 2.429$$

$$\therefore C_{ry} = 0.9(2.3\times10^3)(300)(1 + 2.429^{2(1.34)})^{\frac{-1}{1.34}} = 98.51\text{kN}$$ [98.50 kN]

The steel profile "PFC 160x65" as used under the conditions prescribed by example 3 is adequate in terms of strength.

9.2.2 Example 4 – Columns (continued)

The finite element model is exactly the same as in example 3. This is illustrated in Figure 9-12.

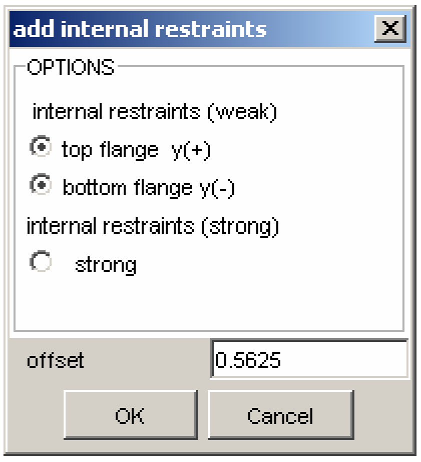
The design member construction is similar to that of example 3, except for two changes.

(1) The loaded steel profiles are an "IPE$_{AA}$ 100" and an "IPE 100".

(2) An internal lateral restraint is applied to the column at 2.25m from its start point along its weak axis.

The application of the internal restraint is illustrated in Figure 9-17.

The 2D view of the steel member is illustrated in Figure 9-18.



add internal restraints

OPTIONS

internal restraints (weak)

[✓ top flange y(+)]

[✓ bottom flange y(-)]

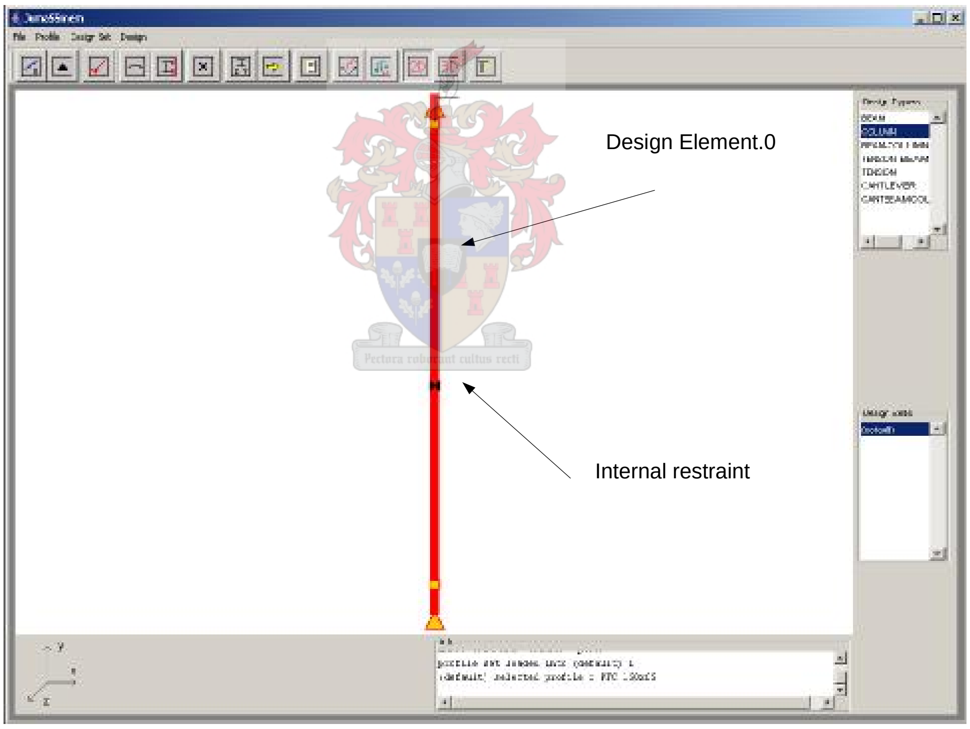
internal restraints (strong)

strong

offset: 0.5625

OK | Cancel

Figure 9-17 Application of the internal restraint for example 4



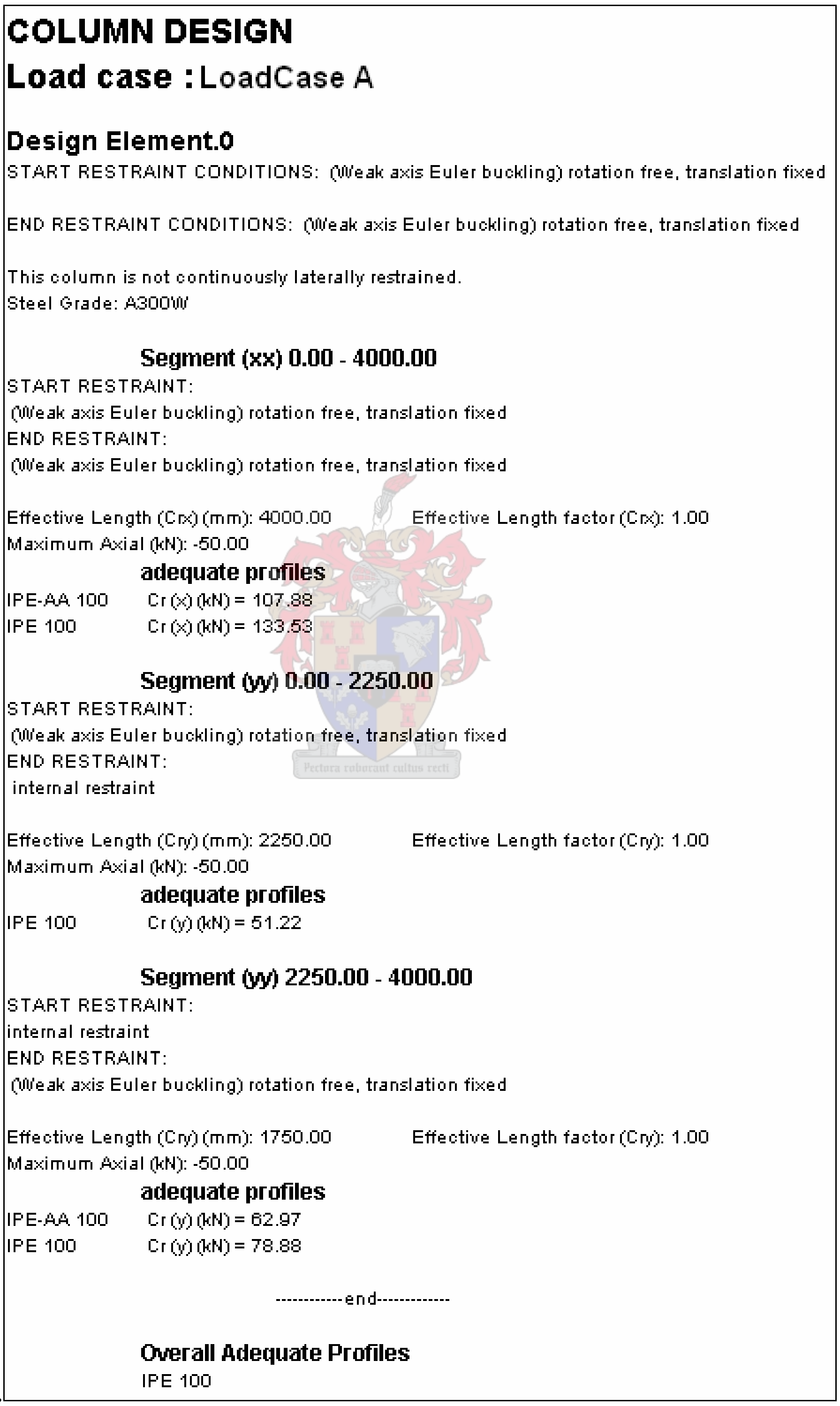
```
profile set [illegible]
(default) selected profile : [illegible]
```

Figure 9-18 Illustration of the steel members in example 4

The internal axial force diagram of example 4 is identical to that of example 3. It is illustrated in Figure 9-14.

The restraint conditions at the ends of the steel member illustrated are the same as in example 3. This is illustrated in Figure 9-15.

As is shown in Figure 9-19, the weak axis of the steel column is divided into two parts, each with their own set of adequate profiles. These two “segments” represent the two portions of unbraced length of the steel column. Each of the segments is designed on its own as if it were an entire steel member. The representation of the adequate profile(s) for the entire column is taken from the intersection of all the adequate profile sets for each segment, similar to example 2.



```
COLUMN DESIGN
Load case : LoadCase A

Design Element.0
START RESTRAINT CONDITIONS:  (Weak axis Euler buckling) rotation free, translation fixed

END RESTRAINT CONDITIONS:  (Weak axis Euler buckling) rotation free, translation fixed

This column is not continuously laterally restrained.
Steel Grade: A300W

                  Segment (xx) 0.00 - 4000.00
START RESTRAINT:
 (Weak axis Euler buckling) rotation free, translation fixed
END RESTRAINT:
 (Weak axis Euler buckling) rotation free, translation fixed

Effective Length (Crx) (mm): 4000.00        Effective Length factor (Crx): 1.00
Maximum Axial (kN): -50.00
                  adequate profiles
IPE-AA 100      Cr (x) (kN) = 107.88
IPE 100         Cr (x) (kN) = 133.53

                  Segment (yy) 0.00 - 2250.00
START RESTRAINT:
 (Weak axis Euler buckling) rotation free, translation fixed
END RESTRAINT:
 internal restraint

Effective Length (Cry) (mm): 2250.00        Effective Length factor (Cry): 1.00
Maximum Axial (kN): -50.00
                  adequate profiles
IPE 100         Cr (y) (kN) = 51.22

                  Segment (yy) 2250.00 - 4000.00
START RESTRAINT:
internal restraint
END RESTRAINT:
 (Weak axis Euler buckling) rotation free, translation fixed

Effective Length (Cry) (mm): 1750.00        Effective Length factor (Cry): 1.00
Maximum Axial (kN): -50.00
                  adequate profiles
IPE-AA 100      Cr (y) (kN) = 62.97
IPE 100         Cr (y) (kN) = 78.88

                        ------------end------------

                  Overall Adequate Profiles
                  IPE 100
```

Figure 9-19 Text results of the design for example 4

Hand calculated design: This section shows the hand calculated design results and comparison with the results from the application.

Classification of profiles

“IPE$_{AA}$ 100”

$$\frac{b}{2\cdot t_f}=\frac{55}{2(4.5)}=6.111<\frac{200}{\sqrt{300}}$$

$$\frac{b}{2\cdot t_w}=\frac{55}{2(3.6)}=7.639\leq\frac{670}{\sqrt{300}}$$

The profile is thus a class 3 profile.

“IPE 100”

$$\frac{b}{2\cdot t_f}=\frac{55}{2(5.7)}=4.825<\frac{200}{\sqrt{300}}$$

$$\frac{b}{2\cdot t_w}=\frac{55}{2(4.1)}=6.707\leq\frac{670}{\sqrt{300}}$$

The profile is thus a class 3 profile.

DesignElement.0

The member is free to buckle about either it’s strong of weak axis. The buckling length of the weak axis is reduced due to the presence of an internal lateral restraint being applied to the member. The profile(s) that were chosen to represent the steel member under these design conditions are bi - symmetrical I – sections resulting in the compressive resistance of the member to be based on flexural buckling. This is discussed in chapter 4. The design is performed by testing the compressive resistances of the member using both chosen profiles about both axes of the member.

The effective length factor calculated from the restraint conditions described earlier in this example is 1.0. [1.0]

The calculation of the compressive resistance is as follows.

Buckling about the strong or x – axis.

“IPE$_{AA}$ 100”

$$\frac{K\cdot L_x}{r_x}=\frac{1.0(4000)}{39.8}=100.5<200 \qquad \text{O.K.}$$

$$\therefore \lambda = \frac{K \cdot L_x}{r_x}\sqrt{\frac{f_y}{\pi^2 \cdot E}} = 100.5\sqrt{\frac{300}{\pi^2(200\times10^3)}} = 1.2389$$

$$C_{rx} = \Phi \cdot A \cdot f_y (1+\lambda^{2n})^{\frac{-1}{n}}$$

The value of n is taken as 1.34 for hot rolled, fabricated steel sections.

$$\therefore C_{rx} = 0.9(0.856\times10^3)(300)(1+1.2389^{2(1.34)})^{\frac{-1}{1.34}} = 107.88\text{kN}$$ [107.88 kN]

"IPE 100"

$$\frac{K \cdot L_x}{r_x} = \frac{1.0(4000)}{40.7} = 98.28 < 200$$ O.K.

$$\therefore \lambda = \frac{K \cdot L_x}{r_x}\sqrt{\frac{f_y}{\pi^2 \cdot E}} = 98.28\sqrt{\frac{300}{\pi^2(200\times10^3)}} = 1.2115$$

$$C_{rx} = \Phi \cdot A \cdot f_y (1+\lambda^{2n})^{\frac{-1}{n}}$$

The value of n is taken as 1.34 for hot rolled, fabricated steel sections.

$$\therefore C_{rx} = 0.9(1.03\times10^3)(300)(1+1.2115^{2(1.34)})^{\frac{-1}{1.34}} = 133.54\text{kN}$$ [133.53 kN]

Both of these profiles are adequate in terms of compressive resistance about the strong axis.

Buckling about the weak or y – axis.

<u>DesignElement.0 (0m – 2.25m)</u>

"IPE$_{AA}$ 100"

$$\frac{K \cdot L_y}{r_y} = \frac{1.0(2250)}{12.1} = 185.950 < 200$$ O.K.

$$\therefore \lambda = \frac{K \cdot L_y}{r_y}\sqrt{\frac{f_y}{\pi^2 \cdot E}} = 185.950\sqrt{\frac{300}{\pi^2(200\times10^3)}} = 2.292$$

$$C_{ry} = \Phi \cdot A \cdot f_y (1+\lambda^{2n})^{\frac{-1}{n}}$$

The value of n is taken as 1.34 for hot rolled, fabricated steel sections.

$$\therefore C_{ry} = 0.9(0.856\times10^3)(300)(1+2.292^{2(1.34)})^{\frac{-1}{1.34}} = 40.736\text{kN}$$ [FAIL]

"IPE 100"

$$\frac{K \cdot L_y}{r_y} = \frac{1.0(2250)}{12.4} = 181.451 < 200 \qquad \text{O.K.}$$

$$\therefore \lambda = \frac{K \cdot L_y}{r_y}\sqrt{\frac{f_y}{\pi^2 \cdot E}} = 181.451\sqrt{\frac{300}{\pi^2(200\times10^3)}} = 2.236$$

$$C_{ry} = \Phi \cdot A \cdot f_y (1+\lambda^{2n})^{\frac{-1}{n}}$$

The value of n is taken as 1.34 for hot rolled, fabricated steel sections.

$$\therefore C_{ry} = 0.9(1.03\times10^3)(300)(1+2.236^{2(1.34)})^{\frac{-1}{1.34}} = 51.227\text{kN} \qquad [51.22 \text{ kN}]$$

As can be seen from the results, the IPE$_{AA}$ fails in terms of compressive resistance under the stipulated member criteria. From the text results shown in Figure 9-19 it is shown that this profile is not indicated under the "adequate profiles" section for this segment.

<u>DesignElement.0 (2.25m – 4m)</u>

"IPE$_{AA}$ 100"

$$\frac{K \cdot L_y}{r_y} = \frac{1.0(1750)}{12.1} = 144.628 < 200 \qquad \text{O.K.}$$

$$\therefore \lambda = \frac{K \cdot L_y}{r_y}\sqrt{\frac{f_y}{\pi^2 \cdot E}} = 144.628\sqrt{\frac{300}{\pi^2(200\times10^3)}} = 1.783$$

$$C_{ry} = \Phi \cdot A \cdot f_y (1+\lambda^{2n})^{\frac{-1}{n}}$$

The value of n is taken as 1.34 for hot rolled, fabricated steel sections.

$$\therefore C_{ry} = 0.9(0.856\times10^3)(300)(1+1.783^{2(1.34)})^{\frac{-1}{1.34}} = 62.99\text{kN} \qquad [62.97 \text{ kN}]$$

"IPE 100"

$$\frac{K \cdot L_y}{r_y} = \frac{1.0(1750)}{12.4} = 141.129 < 200 \qquad \text{O.K.}$$

$$\therefore \lambda = \frac{K \cdot L_y}{r_y}\sqrt{\frac{f_y}{\pi^2 \cdot E}} = 141.129\sqrt{\frac{300}{\pi^2(200\times10^3)}} = 1.739$$

$$C_{ry} = \Phi \cdot A \cdot f_y (1+\lambda^{2n})^{\frac{-1}{n}}$$

The value of n is taken as 1.34 for hot rolled, fabricated steel sections.

$\therefore C_{ry} = 0.9(1.03\times10^{3})(300)(1+1.739^{2(1.34)})^{\frac{-1}{1.34}} = 78.932\text{kN}$ [78.88 kN]

As can be seen from the above calculations, both profiles are adequate for this segment.

The resultant adequate profile shown under the "overall adequate profiles" heading of the text results is "IPE 100". This profile was adequate in terms of compressive resistance for all segments.

9.3 Beam Columns

9.3.1 Example 5 – Beam Column

The finite element model loaded for example 5 is briefly described as follows: The finite element model loaded consists of two Euler beam elements connected end to end to form a vertical continuous beam-column. The beam – column is loaded as illustrated with a 50kN downwards force and a 10 kN horizontal force, causing strong axis bending. The column is built in against rotation and translation at its end point, while the tip is free. The finite element model is briefly illustrated in Figure 9-20 along with the axial and bending moment diagrams.

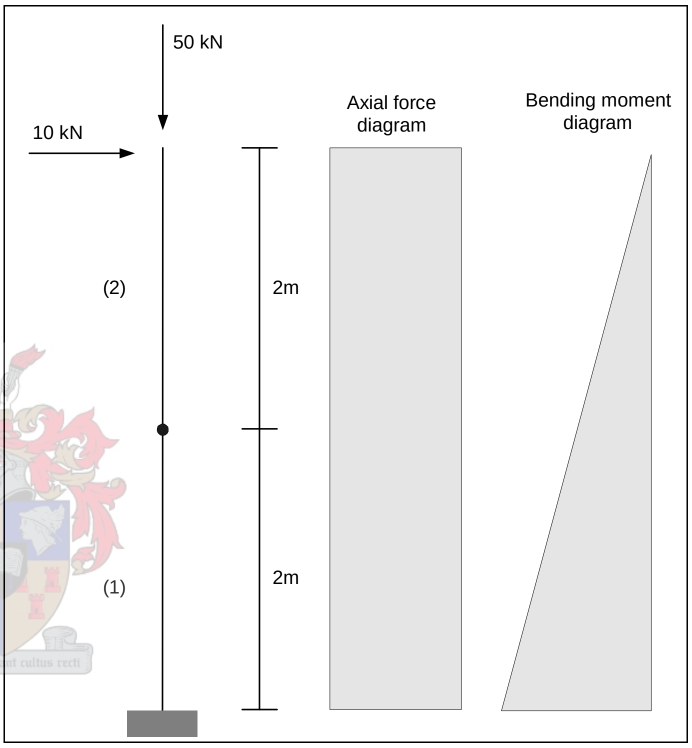


Figure 9-20 Finite element model for example 5

Application design input: In this section the design input and results of internal forces and end forces are shown:

- Design Elements = 1
  "DesignElement.0"
  length = 4m

The Design Element(s) are illustrated in Figure 9-21.

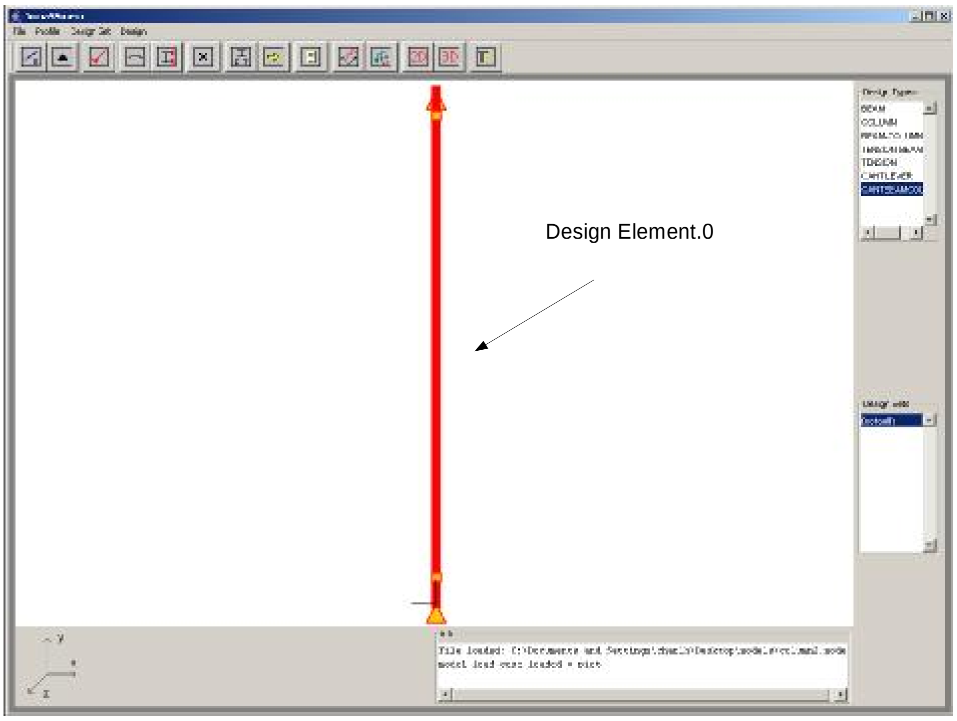


Figure 9-21 Illustration of the steel members of example 5

- Loaded profiles: Grade 300W for Design Set “default”

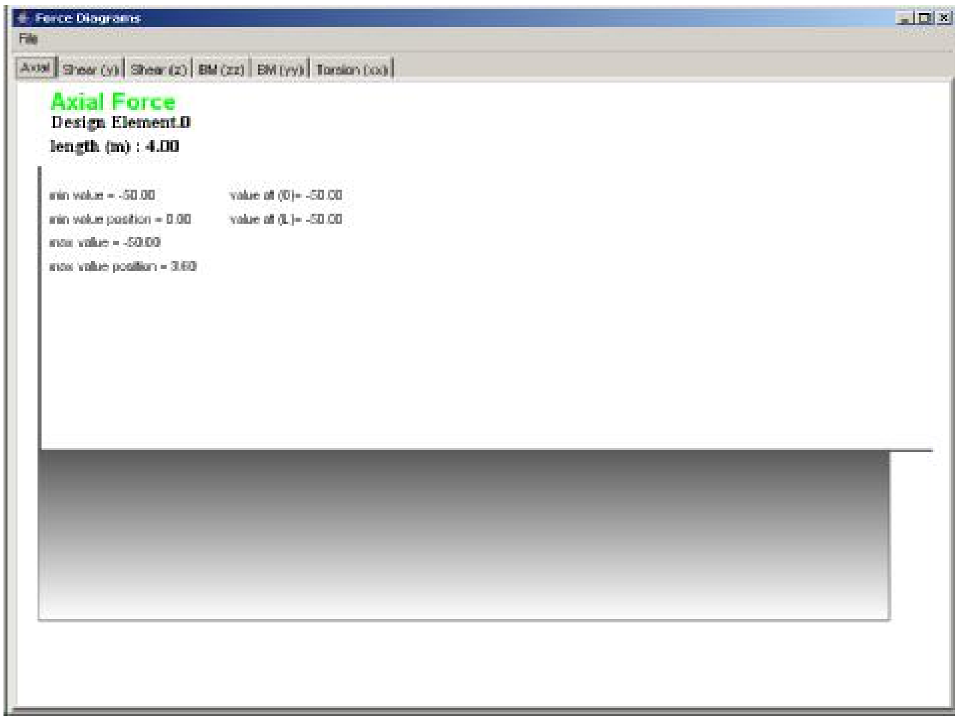
  “203x203x46” “H Profile”

- “DesignElement.0” Ultimate Forces

Axial Force	= -50kN (compressive)
Shear Force (y)	= -10.0 kN
Moments(zz)	= -40.0 kN

- Design Sets = 1

  “default” → “DesignElement.0”

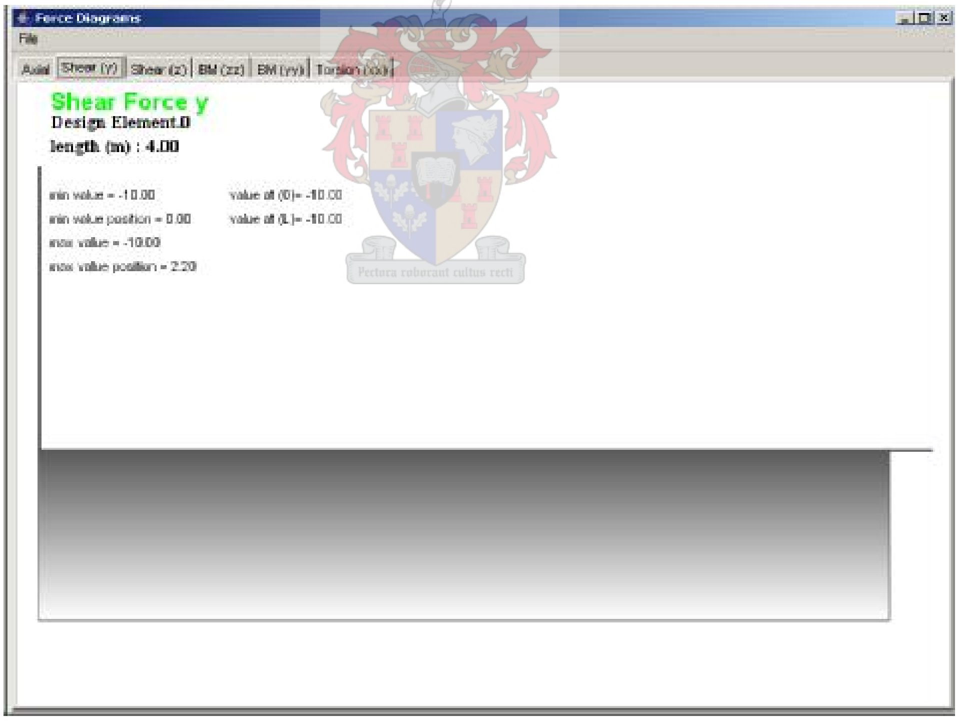


Axial Force
Design Element.0
length (m) : 4.00

min value = -50.00
min value position = 0.00
max value = -50.00
max value position = 3.60

value at (0)= -50.00
value at (L)= -50.00

Figure 9-22 Axial force diagram for example 5

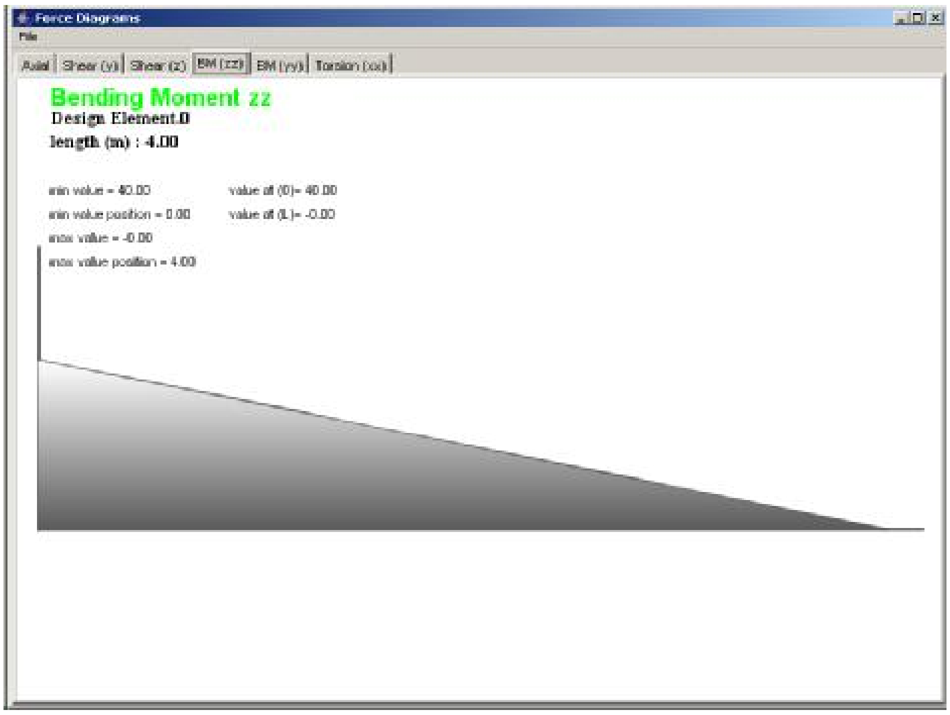


Shear Force y
Design Element.0
length (m) : 4.00

min value = -10.00
min value position = 0.00
max value = -10.00
max value position = 2.20

value at (0)= -10.00
value at (L)= -10.00

Figure 9-23 Shear force diagram of example 5



Bending Moment zz
Design Element.0
length (m) : 4.00

min value = 40.00
min value position = 0.00
max value = -0.00
max value position = 4.00

value at (0)= 40.00
value at (L)= -0.00

Figure 9-24 Bending moment diagram of example 5

Figure 9-22, Figure 9-23 and Figure 9-24 represent the non – zero internal force diagrams for example 5.

The applied restraint conditions against lateral torsional buckling are as follows:

- “DesignElement.0” Restraint conditions

  (start) built in laterally and torsionally

  (end) free

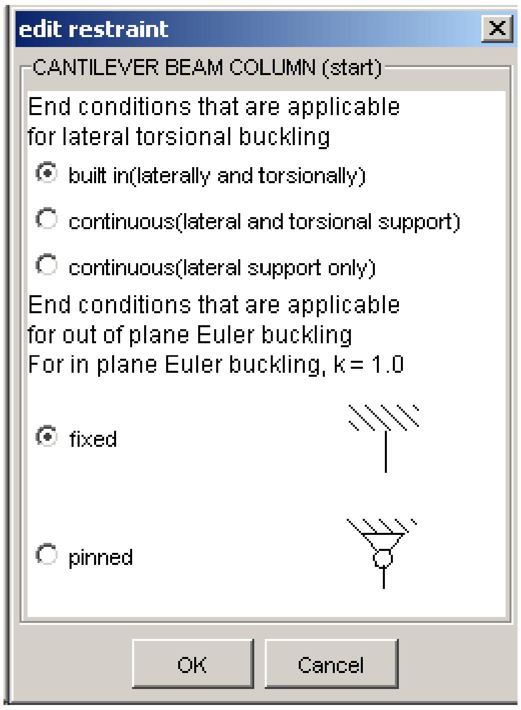
The applied restraint conditions against weak axis Euler buckling are as follows:

- “DesignElement.0” Restraint conditions

  (start) fixed

  (end) free

The application of such restraint conditions are illustrated in Figure 9-25.



edit restraint

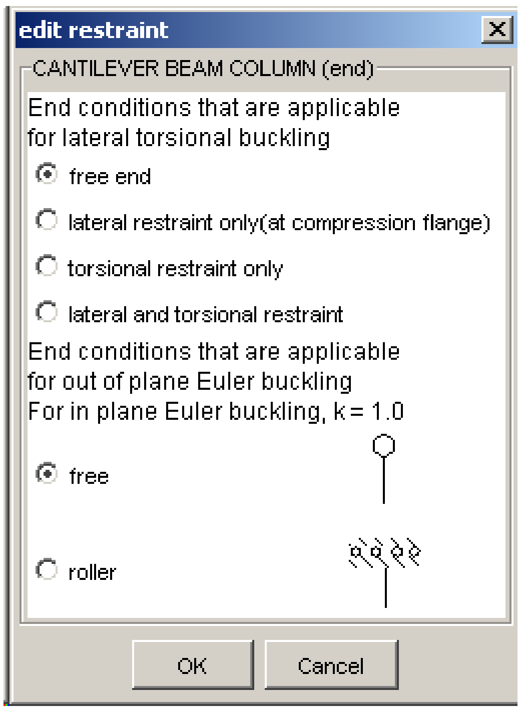
CANTILEVER BEAM COLUMN (start)

End conditions that are applicable for lateral torsional buckling

[✓ built in(laterally and torsionally)]
continuous(lateral and torsional support)
continuous(lateral support only)

End conditions that are applicable for out of plane Euler buckling
For in plane Euler buckling, k = 1.0

[✓ fixed]
pinned



edit restraint

CANTILEVER BEAM COLUMN (end)

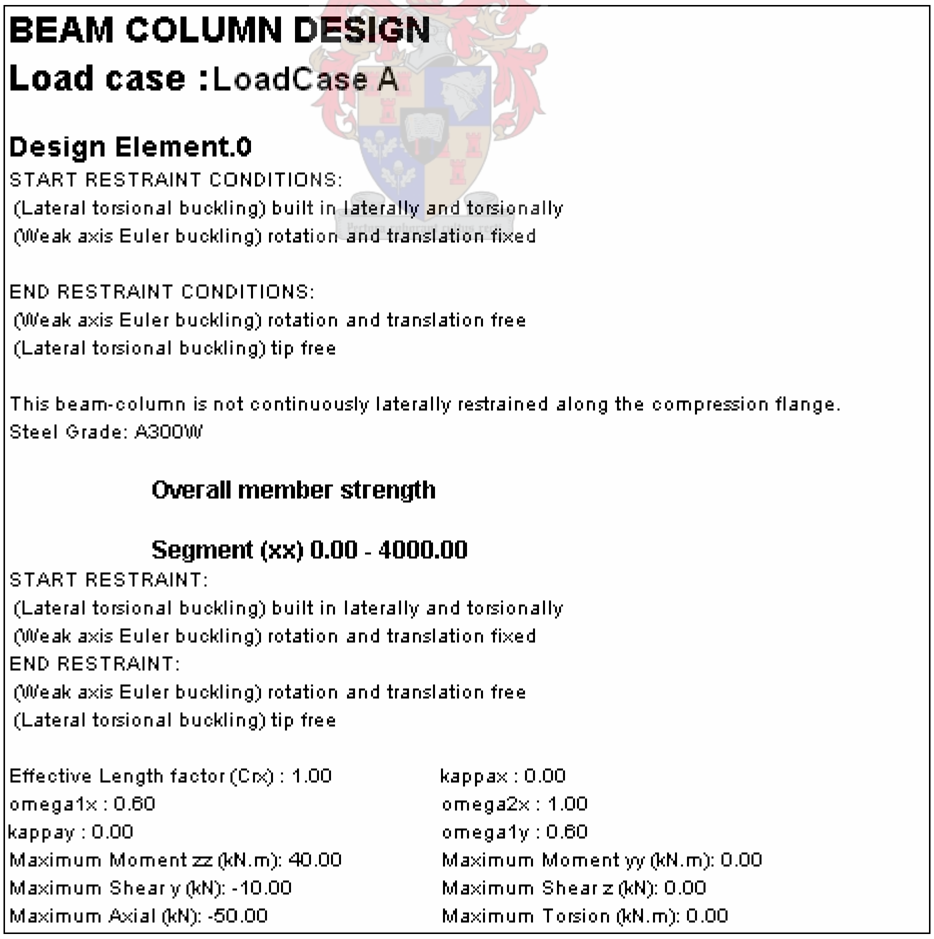
End conditions that are applicable for lateral torsional buckling

[✓ free end]
lateral restraint only(at compression flange)
torsional restraint only
lateral and torsional restraint

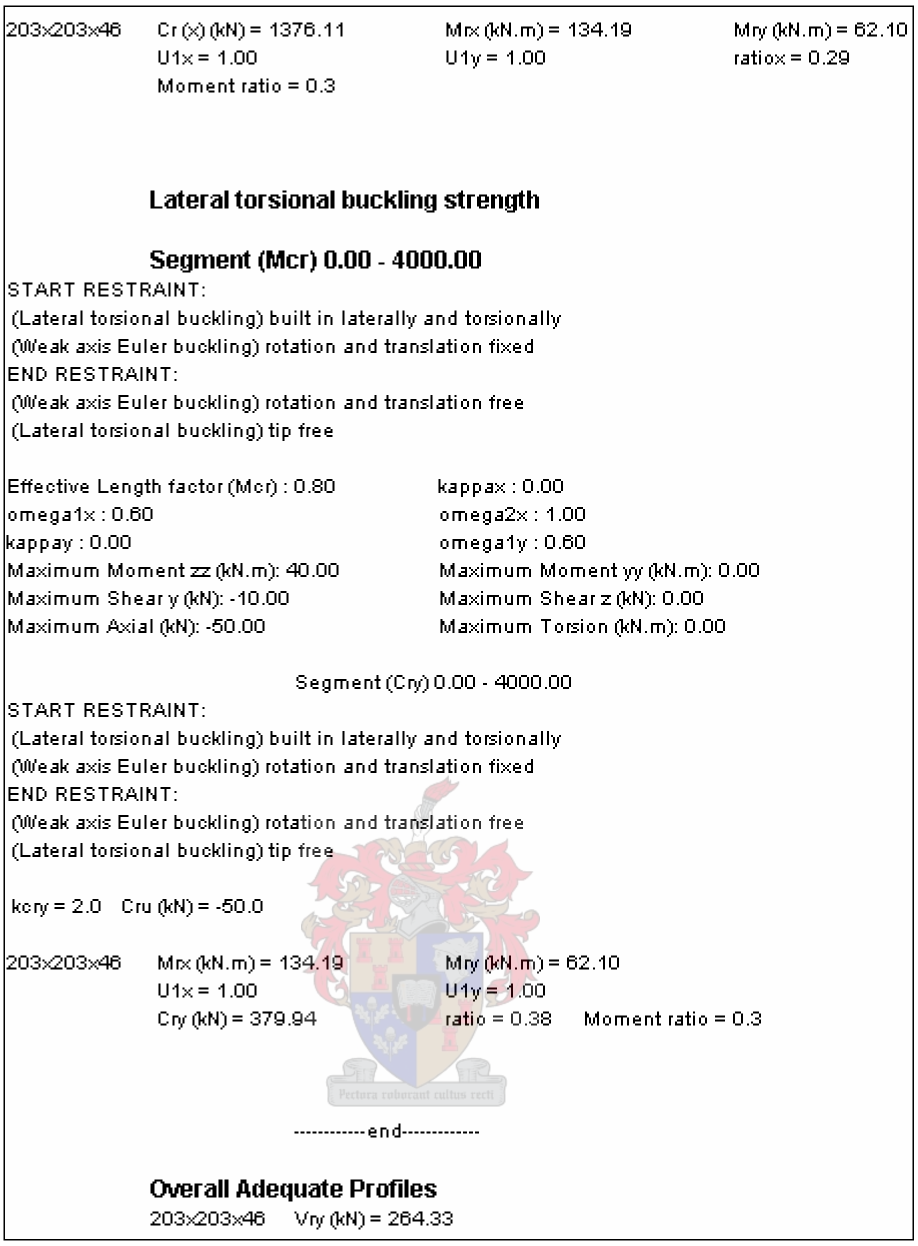
End conditions that are applicable for out of plane Euler buckling
For in plane Euler buckling, k = 1.0

[✓ free]
roller

Figure 9-25 Illustration of the restraint conditions for example 5



```
BEAM COLUMN DESIGN
Load case :LoadCase A

Design Element.0
START RESTRAINT CONDITIONS:
 (Lateral torsional buckling) built in laterally and torsionally
 (Weak axis Euler buckling) rotation and translation fixed

END RESTRAINT CONDITIONS:
 (Weak axis Euler buckling) rotation and translation free
 (Lateral torsional buckling) tip free

This beam-column is not continuously laterally restrained along the compression flange.
Steel Grade: A300W

            Overall member strength

            Segment (xx) 0.00 - 4000.00
START RESTRAINT:
 (Lateral torsional buckling) built in laterally and torsionally
 (Weak axis Euler buckling) rotation and translation fixed
END RESTRAINT:
 (Weak axis Euler buckling) rotation and translation free
 (Lateral torsional buckling) tip free

Effective Length factor (Crx) : 1.00          kappax : 0.00
omega1x : 0.60                                omega2x : 1.00
kappay : 0.00                                 omega1y : 0.60
Maximum Moment zz (kN.m): 40.00               Maximum Moment yy (kN.m): 0.00
Maximum Shear y (kN): -10.00                  Maximum Shear z (kN): 0.00
Maximum Axial (kN): -50.00                    Maximum Torsion (kN.m): 0.00
```



```
203x203x46     Cr (x) (kN) = 1376.11          Mrx (kN.m) = 134.19          Mry (kN.m) = 62.10
               U1x = 1.00                     U1y = 1.00                   ratiox = 0.29
               Moment ratio = 0.3


               Lateral torsional buckling strength

               Segment (Mcr) 0.00 - 4000.00
START RESTRAINT:
 (Lateral torsional buckling) built in laterally and torsionally
 (Weak axis Euler buckling) rotation and translation fixed
END RESTRAINT:
 (Weak axis Euler buckling) rotation and translation free
 (Lateral torsional buckling) tip free

Effective Length factor (Mcr) : 0.80          kappax : 0.00
omega1x : 0.60                                omega2x : 1.00
kappay : 0.00                                 omega1y : 0.60
Maximum Moment zz (kN.m): 40.00               Maximum Moment yy (kN.m): 0.00
Maximum Shear y (kN): -10.00                  Maximum Shear z (kN): 0.00
Maximum Axial (kN): -50.00                    Maximum Torsion (kN.m): 0.00

                              Segment (Cry) 0.00 - 4000.00
START RESTRAINT:
 (Lateral torsional buckling) built in laterally and torsionally
 (Weak axis Euler buckling) rotation and translation fixed
END RESTRAINT:
 (Weak axis Euler buckling) rotation and translation free
 (Lateral torsional buckling) tip free

 kcry = 2.0    Cru (kN) = -50.0

203x203x46     Mrx (kN.m) = 134.19            Mry (kN.m) = 62.10
               U1x = 1.00                     U1y = 1.00
               Cry (kN) = 379.94              ratio = 0.38     Moment ratio = 0.3


                              ------------end------------

               Overall Adequate Profiles
               203x203x46     Vry (kN) = 264.33
```

Figure 9-26 Text results of the design for example 5

The text design results are as illustrated in Figure 9-26. These results indicate the chosen steel profile, namely “203x203x46”, is adequate in terms of the design requirements as discussed in section 4.

As can be seen from the text results, the steel member is tested for 2 cases of resistance, namely the overall member strength and (when applicable) the torsional buckling strength.

Hand calculated design: This section shows the hand calculated design results and comparison with the results from the application.

Classification of profiles

“203x203x46”

$h = 203.2$

$b = 203.2$

$t_f = 11$

$t_w = 7.3$

$$\frac{b}{2 \cdot t_f} = \frac{203.2}{2(11)} = 9.236 < \frac{170}{\sqrt{300}} \rightarrow \text{class 2 flange}$$

$C_u = 50$ kN

$$C_y = A \cdot f_y = 5.88 \times 10^3 (300) = 1764\text{kN}$$

$$\frac{h - 2t_f}{t_w} = \frac{203.2 - 2(11)}{7.3} = 24.821 \le \frac{1100}{\sqrt{300}} \left(1 - 0.39 \frac{50}{\Phi \cdot 1764}\right) \rightarrow \text{class 1 web}$$

The profile is therefore a class 2 profile.

<u>DesignElement.0</u>

This element is not continuously braced along its weak axis, resulting in the lateral torsional buckling strength to be included in the design process. Although the member is unbraced, the cross sectional strength of the member is still tested.

The effective length of the member, for weak axis Euler buckling, is calculated as 2.0 due to the restraint conditions described in the design makeup. [2.0]

The effective length of the member, for lateral torsional buckling, is calculated as 0.8 due to the restraint conditions described in the design makeup. [0.8]

For strong axis Euler buckling, the effective length factor is taken as 1.0 due to a second order analysis assumed from the finite element analysis. [1.0]

The capacity of the member is examined for the following strength cases and is based on the following interaction formula, $\frac{C_u}{C_r} + \frac{0.85 \cdot U_{1x} \cdot M_{ux}}{M_{rx}} \le 1.0$.

- *Cross – sectional strength*

Due to the fact that the member is unbraced, the cross sectional resistance test of the member is not necessary. The application still performs the test but the results are not shown in the text results.

$C_r = \Phi \cdot A \cdot f_y = 0.9(5.88 \times 10^3)(300) = 1587.6\text{kN}$ [1587.60 kN]

$M_{rx} = \Phi \cdot Z_{plx} \cdot f_y = 0.9(497 \times 10^3)(300) = 134.19\text{kN.m}$ [134.19 kN.m]

$$U_{1x} = \frac{\omega_1}{1 - \frac{C_u}{C_{ex}}}$$

$\omega_1 = 0.6 - 0.4 \cdot \kappa \geq 0.4$

As can be seen from the bending moment diagram of this example, shown in Figure 9-24, the largest end moment appears at the end of the member with the other end equal to zero. This results in a value for κ to be equal to zero. [0.0]

$\therefore \omega_1 = 0.6 - 0.4(0.0) = 0.6$ [0.6]

$$C_{ex} = \frac{\pi^2 \cdot E \cdot I_x}{L_x} = \frac{\pi^2(200\times10^3)(45.6\times10^6)}{(4000)^2} = 5625.61\text{kN}$$

$$\therefore U_{1x} = \frac{0.6}{1 - \frac{50}{5625.61}} = 0.605$$

This value may not be less than 1.0 for the cross sectional analysis. [1.0]

The result of the interaction is therefore,

$$\frac{50}{1587.60} + \frac{0.85\cdot(1.0)(40)}{134.19} = 0.285 \leq 1.0$$ [0.28]

The profile is thus adequate in terms of cross sectional strength as illustrated by the interaction formula.

- *Overall member strength*

The compressive resistance of the member is calculated for buckling about the strong axis, due to uniaxial strong axis bending.

$$C_{rx} = \Phi \cdot A \cdot f_y \left(1 + \lambda^{2n}\right)^{\frac{-1}{n}}$$

$$\lambda = \frac{K \cdot L_x}{r_x} \cdot \sqrt{\left(\frac{f_y}{\pi^2 \cdot E}\right)} = \frac{1.0(4000)}{88.1}\sqrt{\left(\frac{300}{\pi^2(200\times10^3)}\right)} = 0.56$$

$\therefore C_{rx} = 0.9(5.88\times10^3)(300)(1 + 0.56^{2\cdot1.34})^{\frac{-1}{1.34}} = 1375.88\text{kN}$ [1376.11 kN]

$M_{rx} = \Phi \cdot Z_{plx} \cdot f_y = 0.9(497\times10^3)(300) = 134.19\text{kN.m}$ [134.19 kN.m]

$$U_{1x} = \frac{\omega_1}{1 - \frac{C_u}{C_{ex}}}$$

$\omega_1 = 0.6 - 0.4 \cdot \kappa \geq 0.4$

As can be seen from the bending moment diagram of this example, shown in Figure 9-24, the largest end moment appears at the end of the member with the other end equal to zero. This results in a value for κ to be equal to zero. [0.0]

$\therefore \omega_1 = 0.6 - 0.4(0.0) = 0.6$ [0.6]

$\therefore U_{1x} = 1.0$ for the overall member strength test of unbraced members [1.0]

The result of the interaction is therefore,

$$\frac{50}{1375.88} + \frac{0.85 \cdot (1.0)(40)}{134.19} = 0.29 \leq 1.0$$ [0.29]

The profile is thus adequate in terms of overall member strength as illustrated by the interaction formula.

- *Lateral torsional buckling strength*

The compressive resistance of the member, as used in the interaction formula, is dependent on weak axis or torsional flexural buckling.

$$C_{ry} = \Phi \cdot A \cdot f_y \left(1 + \lambda^{2n}\right)^{\frac{-1}{n}}$$

$$\lambda = \frac{K \cdot L_y}{r_y} \cdot \sqrt{\left(\frac{f_y}{\pi^2 \cdot E}\right)} = \frac{2.0(4000)}{51.2} \sqrt{\left(\frac{300}{\pi^2 (200 \times 10^3)}\right)} = 1.926$$

$$\therefore C_{ry} = 0.9(5.88 \times 10^3)(300)(1 + 1.926^{2 \cdot 1.34})^{\frac{-1}{1.34}} = 380\text{kN}$$ [379.94 kN]

The bending resistance of the member is based on lateral torsional buckling and not on cross sectional resistance.

$$M_{cr} = \frac{\omega_2 \cdot \pi}{K \cdot L} \sqrt{EI_y GJ + \left(\frac{\pi \cdot E}{K \cdot L}\right)^2 I_y C_w}$$

The value of κ remains 0 as calculated from the previous tests.

$$\omega_2 = 1.75 + 1.05 \cdot \kappa + 0.3 \cdot \kappa^2 = 1.75 + 1.05(0) + 0.3(0) = 1.75$$

The design code as implemented in section 4, states that if there is no effective lateral support at any of the ends of the member, the value of ω_2 shall be taken as 1.0. The restraint conditions described indicate that one of the ends of the member is completely free resulting in no effective lateral support at that end.

$\therefore \omega_2 = 1.0$ [1.0]

$$\therefore \frac{1.0(\pi)}{0.8(4000)}\sqrt{200\times10^{3}(15.4\times10^{6})(77\times10^{3})(225\times10^{3})+\left(\frac{(\pi)(200\times10^{3})}{(0.8)4000}\right)^{2}(15.4\times10^{6})(142\times10^{9})}$$

$= 364.26$ kN.m $> 0.67\ M_p$

This value is larger than the value of $0.67M_p = 0.67(141.9) = 99.9$ kN.m. The bending resistance of the segment is then calculated as follows:

$$M_r = 1.15\cdot\Phi\cdot M_p\cdot\left(1-\frac{0.28\cdot M_p}{M_{cr}}\right)\leq\Phi\cdot M_p$$

$$\therefore = 1.15(0.9)(149.1)\left(1-\frac{0.28(149.1)}{364.26}\right) = 136.63 \text{ kN.m} > 134.19 \text{ kN.m}$$

Due to the fact that the value calculated is larger than $\Phi\ M_p$, the value of bending resistance is calculated as $\Phi\ M_p$ as discussed in section 4.

$$\therefore M_{rx} = \Phi\cdot M_p = 0.9(149.1) = 134.19\text{kN.m}$$ [134.19 kN.m]

$$U_{1x} = \frac{\omega_1}{1-\frac{C_u}{C_{ex}}}$$

$$\omega_1 = 0.6-0.4\cdot\kappa\geq0.4$$

As can be seen from the bending moment diagram of this example, shown in Figure 9-24, the largest end moment appears at the end of the member with the other end equal to zero. This results in a value for κ to be equal to zero. [0.0]

$$\therefore \omega_1 = 0.6-0.4(0.0) = 0.6$$ [0.6]

$$C_{ex} = \frac{\pi^2\cdot E\cdot I_x}{L_x} = \frac{\pi^2(200\times10^{3})(45.6\times10^{6})}{(4000)^2} = 5625.61\text{kN}$$

$$\therefore U_{1x} = \frac{0.6}{1-\frac{50}{5625.61}} = 0.605$$

This value may not be less than 1.0 for this particular test. [1.0]

The result of the interaction is therefore,

$$\frac{50}{380}+\frac{0.85\cdot(1.0)(40)}{134.19} = 0.385\leq1.0$$ [0.38]

$$\frac{M_u}{M_{rx}} = \frac{40}{134.19} = 0.298$$ [0.3]

The chosen profile, “203x203x46”, is adequate in terms of strength as indicated under the “Overall Adequate Profiles” section in the text results.

9.3.2 Example 6 – Portal Frame

The finite element model consists of 8 Euler beam elements combined to form a frame structure. This is illustrated in Figure 9-27. The frame is loaded as shown, causing bending about the strong axis of the members. The supports of the frame prevent any horizontal and vertical translation, while rotation is allowed. For the purpose of design, the finite element results are assumed to be from a second order analysis, which include the sway effects of the structure in the analysis

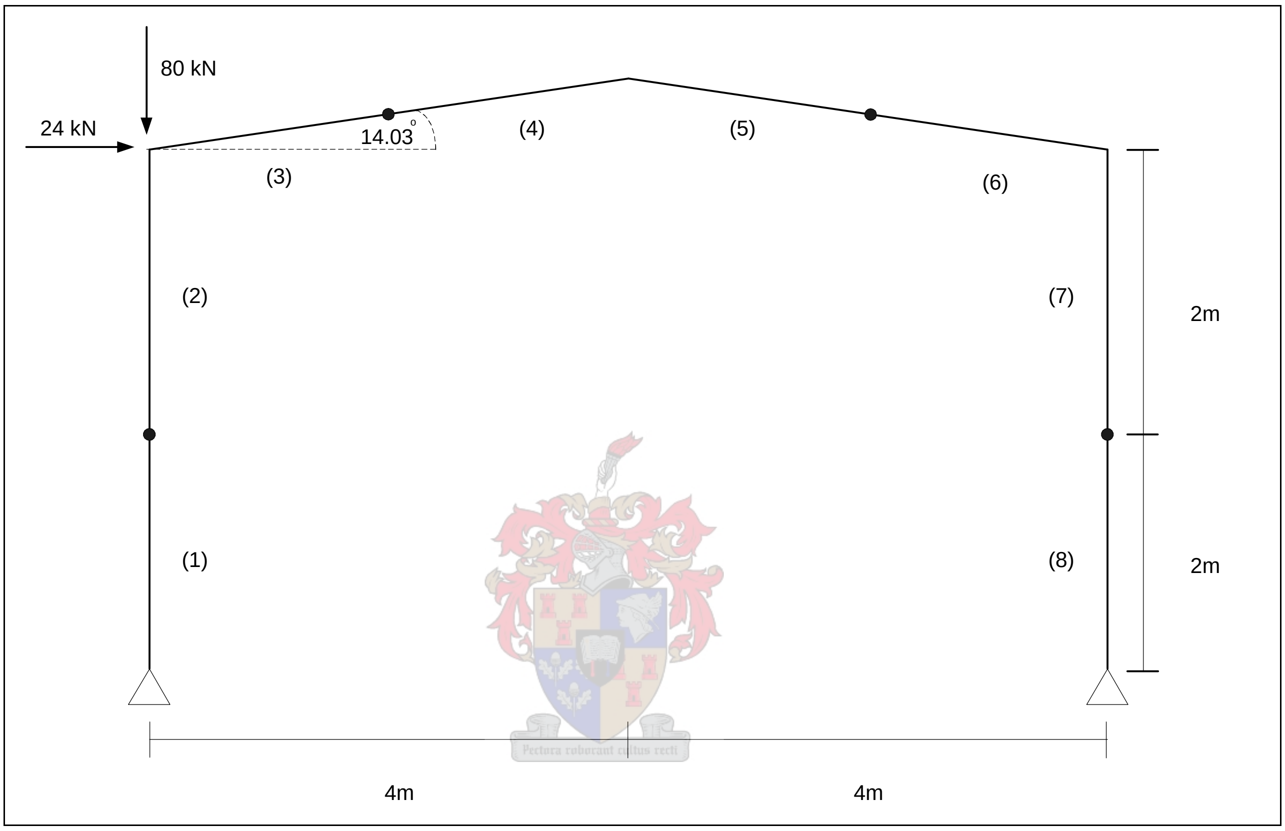


Figure 9-27 Illustration of the loaded finite element model

Application design input: In this section the design input and results of internal forces and end forces are shown:

- Design Elements = 4

"DesignElement.0"	
length	= 4m
"DesignElement.1"	
length	= 4.12m
"DesignElement.2"	
length	= 4.12m
"DesignElement.3"	
length	= 4m

For the purpose of this example, the internal force diagrams are shown for all Design Elements. The text design results as generated by the application are shown for DesignElement.3. Thus the hand calculations for Design Element.3 are used for comparison with the framework results.

- Design Sets = 2

"rafters"	→ "DesignElement.1" "DesignElement.2"
"columns"	→ "DesignElement.0" "DesignElement.3"

- Loaded profiles: Grade 300W for Design Set "rafters"

  "203x133x30" "I – section"
- Loaded profiles: Grade 300W for Design Set "columns"

  "254x146x31" "I – section"
- "DesignElement.0" Ultimate Forces

Axial Force	= -68kN (compressive)
Shear Force (y)	= -13.10 kN
Moment(zz)	= -52.39 kN.m

- "DesignElement.1" Ultimate Forces

Axial Force	= -7.67kN (compressive)
Shear Force (y)	= 14.29 kN
Moment(zz)	= -52.39 kN.m

- "DesignElement.2" Ultimate Forces

Axial Force	= -13.49kN (compressive)
Shear Force (y)	= 9.0 kN
Moment(zz)	= 43.61 kN.m

- "DesignElement.3" Ultimate Forces

Axial Force	= -12.0kN (compressive)
Shear Force (y)	= -10.90 kN
Moment(zz)	= -43.61 kN.m

As stated in the beginning of this example, a second order analysis is said to be included in the finite element analysis of the model. This results in the effective length factor for determining the strong axis Euler buckling resistance of the members to be taken as 1.0.

Two internal restraints are placed along the length of "DesignElement.3". The first internal restraint is placed at a position of 2m from the start point of the member. This internal restraint is applied to the top flange of the member. The second internal restraint is placed at a distance of 3m from the start point of the member. This internal restraint is applied to the bottom flange of the member.

Figure 9-28 illustrates all the steel members and their internal restraints

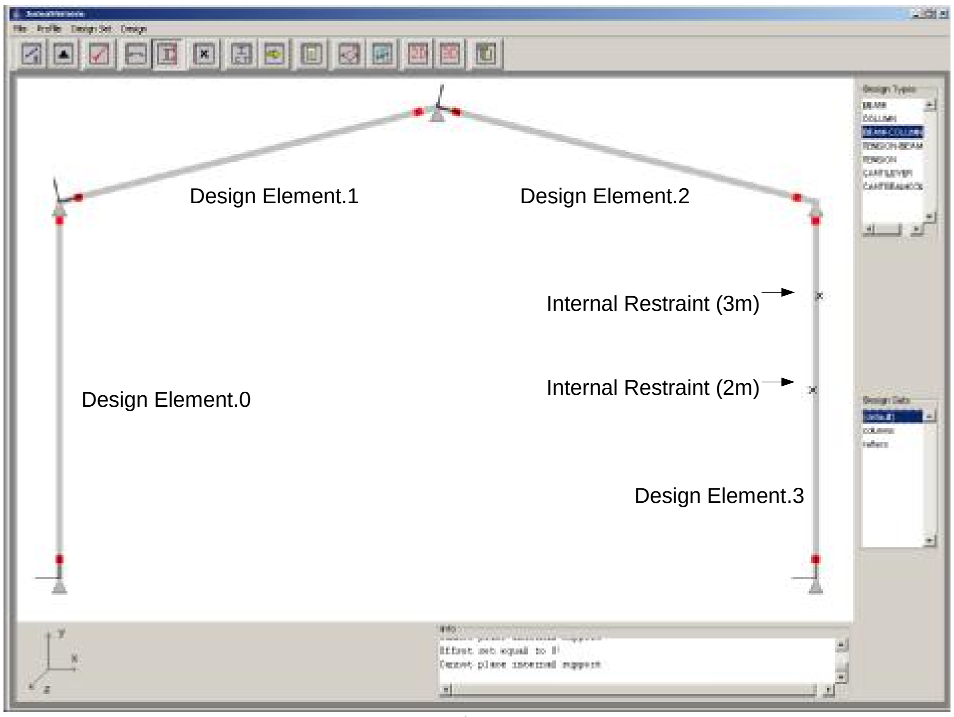


Figure 9-28 Illustrating the design members for example 6

The axial force and bending moment diagrams for each Design Element are illustrated in Figure 9-30 and Figure 9-31. The text design results of DesignElement.3 are shown in Figure 9-32.

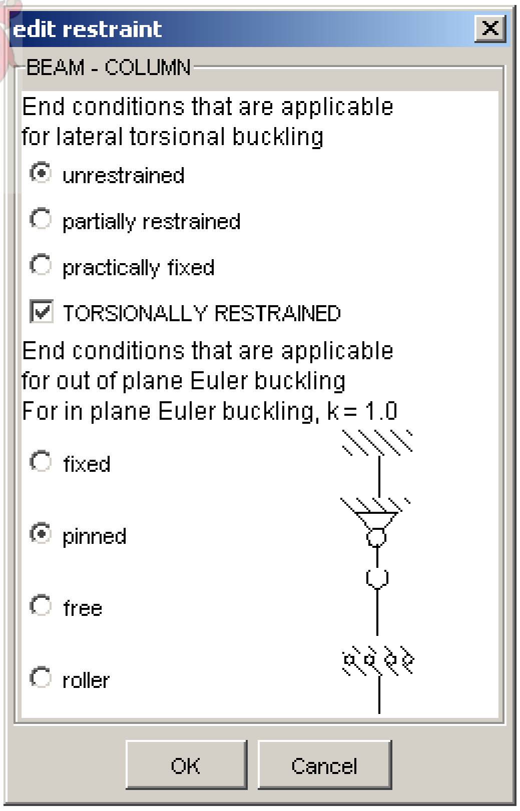
The restraint condition of DesignElement.3 against lateral torsional buckling is as follows:

- unrestrained (free to rotate about the vertical axis
- pinned (translation fixed, rotation free) for weak axis Euler buckling

The restraint condition of DesignElement.3 against weak axis Euler buckling is as follows:

- pinned (translation free, rotation fixed)

This is shown in Figure 9-29.


edit restraint

BEAM - COLUMN

End conditions that are applicable for lateral torsional buckling

[✓ unrestrained] partially restrained practically fixed

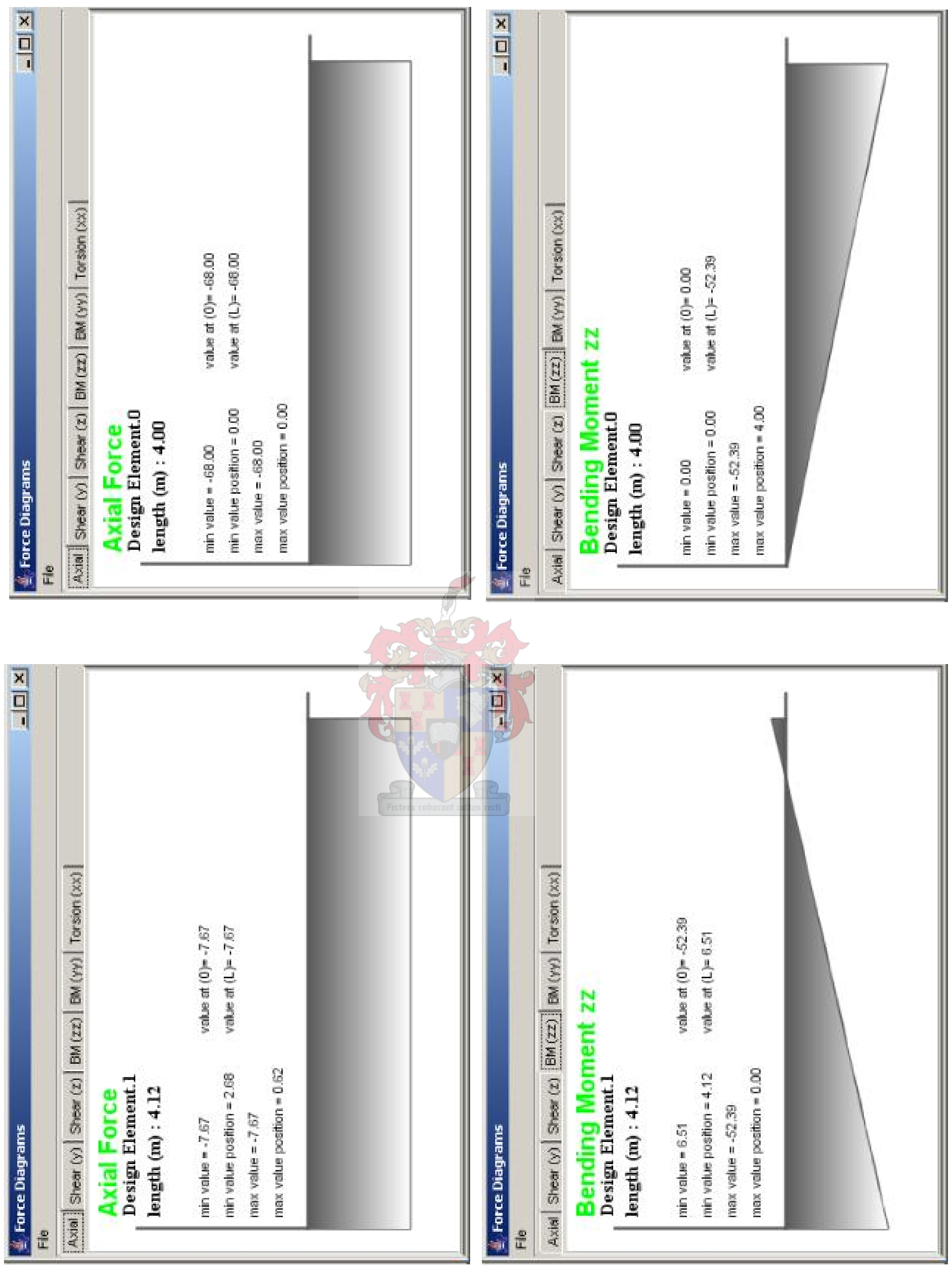
[✓ TORSIONALLY RESTRAINED]

End conditions that are applicable for out of plane Euler buckling

For in plane Euler buckling, k = 1.0

fixed [✓ pinned] free roller

Figure 9-29 Restraint conditions for "DesignElement.3"



Axial Force
Design Element.0
length (m) : 4.00

min value = -68.00
min value position = 0.00
max value = -68.00
max value position = 0.00
value at (0)= -68.00
value at (L)= -68.00

Bending Moment zz
Design Element.0
length (m) : 4.00

min value = 0.00
min value position = 0.00
max value = -52.39
max value position = 4.00
value at (0)= 0.00
value at (L)= -52.39

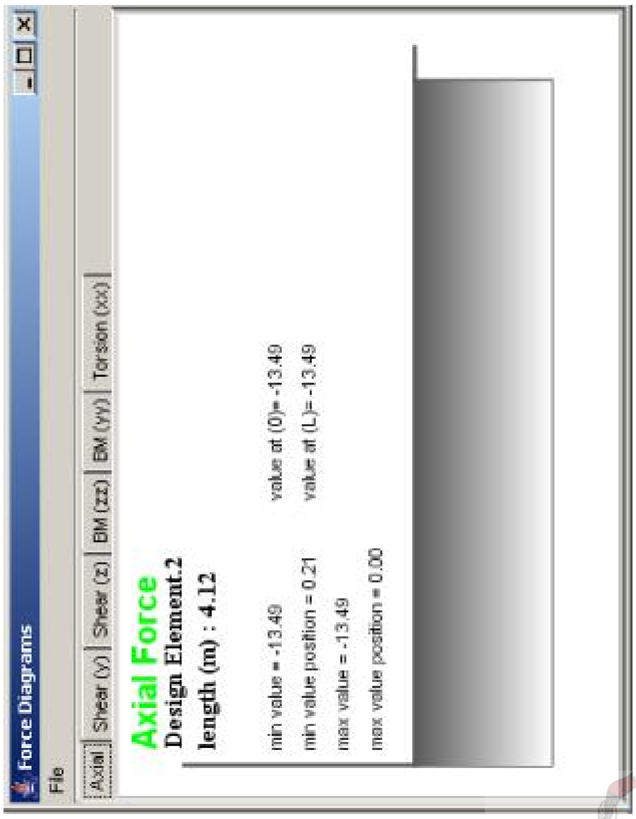
Axial Force
Design Element.1
length (m) : 4.12

min value = -7.67
min value position = 2.68
max value = -7.67
max value position = 0.62
value at (0)= -7.67
value at (L)= -7.67

Bending Moment zz
Design Element.1
length (m) : 4.12

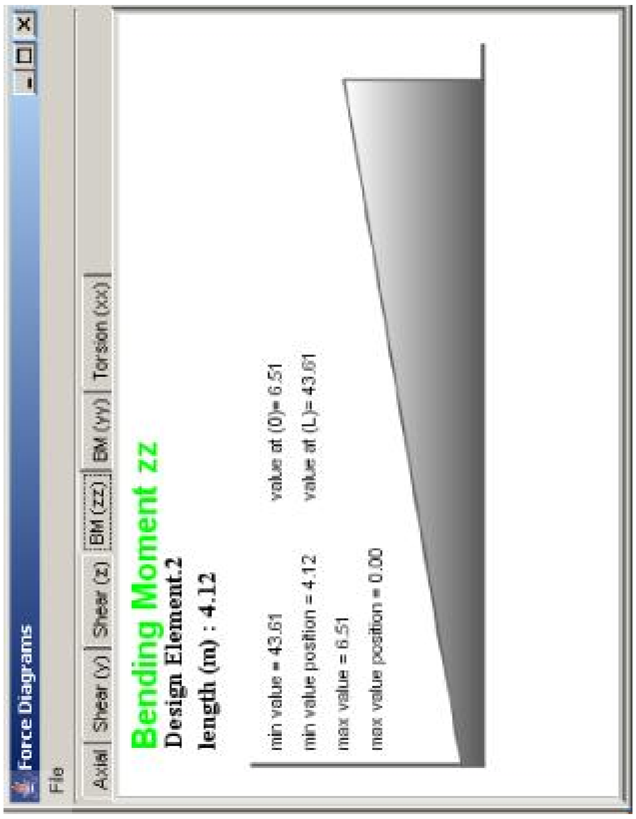
min value = 6.51
min value position = 4.12
max value = -52.39
max value position = 0.00
value at (0)= -52.39
value at (L)= 6.51

Figure 9-30 Axial force and Bending moment diagrams for "DesignElement.0" and “Design Element.1”



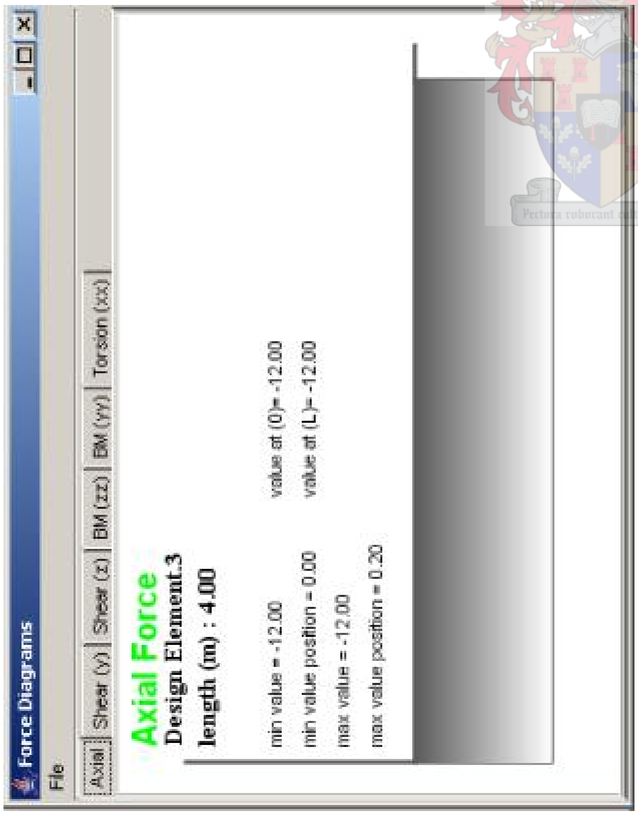
Axial Force
Design Element.2
length (m) : 4.12

min value = -13.49
min value position = 0.21
max value = -13.49
max value position = 0.00
value at (0)= -13.49
value at (L)= -13.49



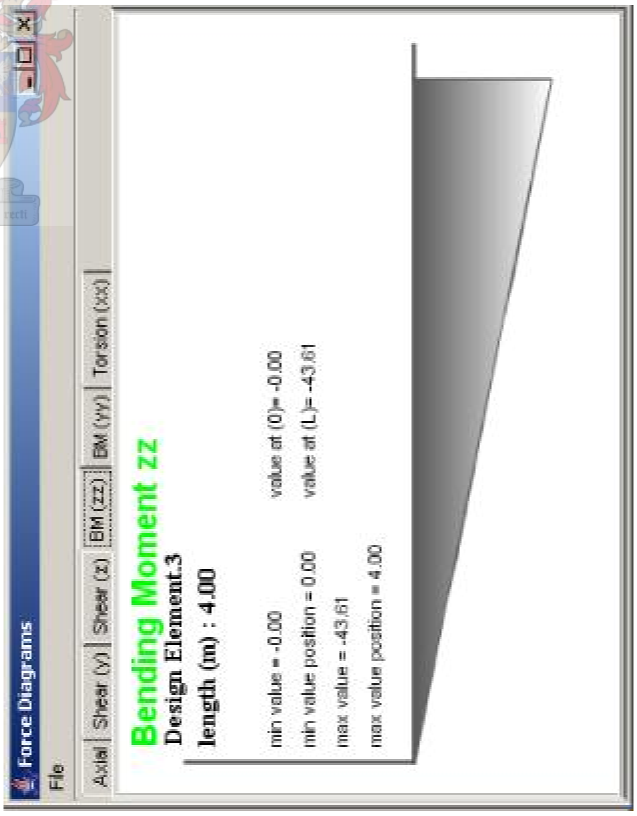
Bending Moment zz
Design Element.2
length (m) : 4.12

min value = 43.61
min value position = 4.12
max value = 6.51
max value position = 0.00
value at (0)= 6.51
value at (L)= 43.61



Axial Force
Design Element.3
length (m) : 4.00

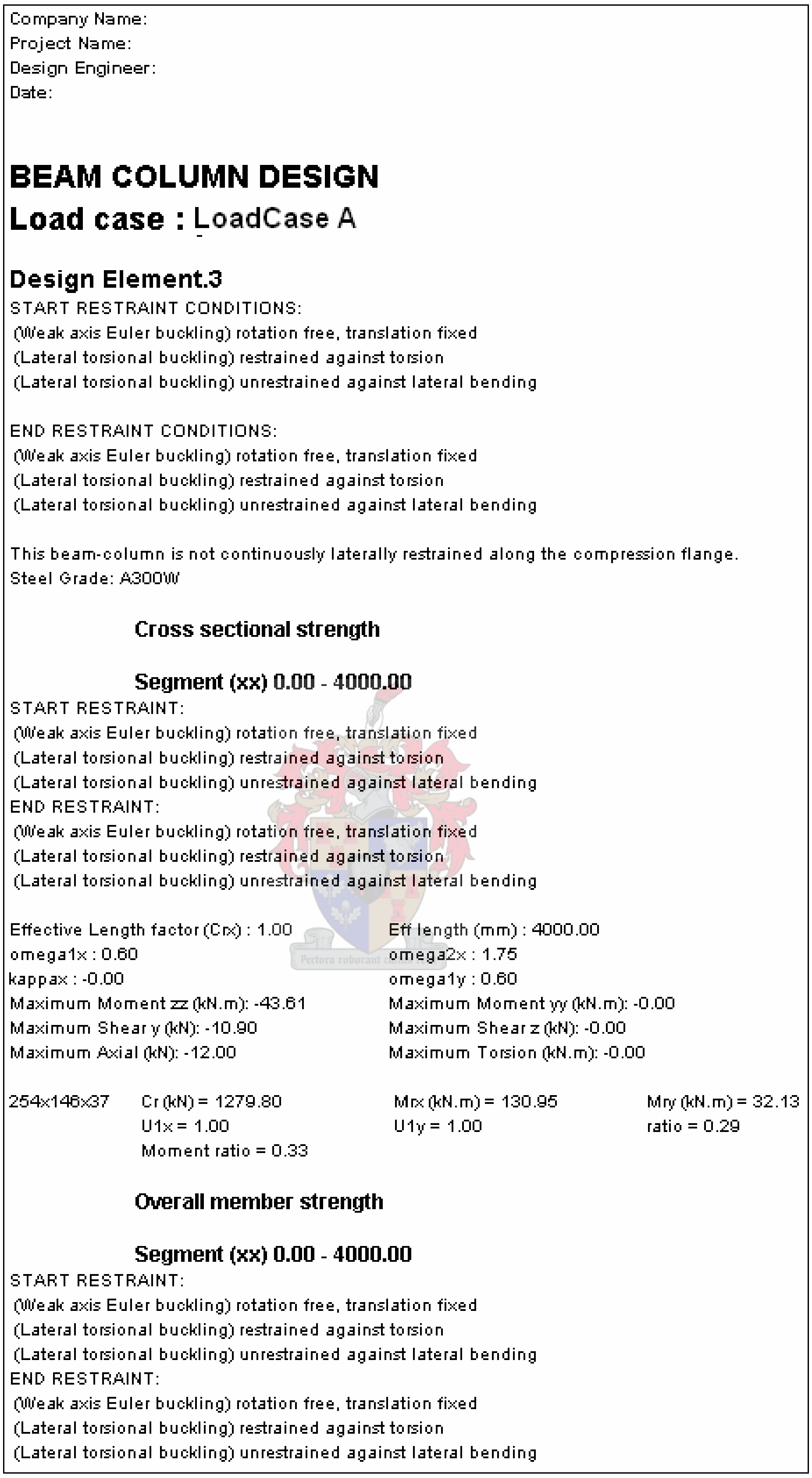
min value = -12.00
min value position = 0.00
max value = -12.00
max value position = 0.20
value at (0)= -12.00
value at (L)= -12.00



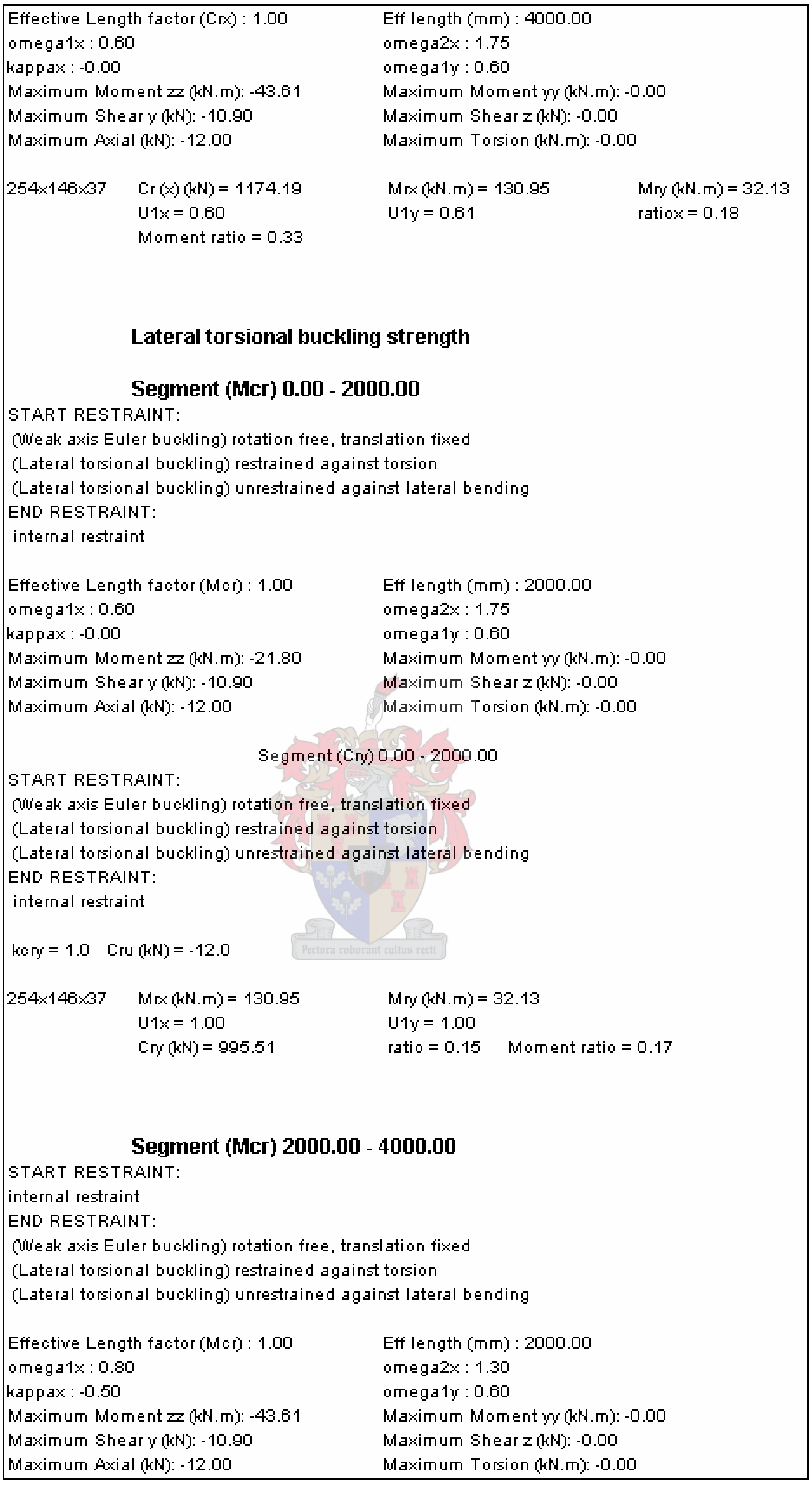
Bending Moment zz
Design Element.3
length (m) : 4.00

min value = -0.00
min value position = 0.00
max value = -43.61
max value position = 4.00
value at (0)= -0.00
value at (L)= -43.61

Figure 9-31 Axial force and Bending moment diagrams for "DesignElement.2" and “Design Element.3"



```
Company Name:
Project Name:
Design Engineer:
Date:


BEAM COLUMN DESIGN
Load case : LoadCase A

Design Element.3
START RESTRAINT CONDITIONS:
 (Weak axis Euler buckling) rotation free, translation fixed
 (Lateral torsional buckling) restrained against torsion
 (Lateral torsional buckling) unrestrained against lateral bending

END RESTRAINT CONDITIONS:
 (Weak axis Euler buckling) rotation free, translation fixed
 (Lateral torsional buckling) restrained against torsion
 (Lateral torsional buckling) unrestrained against lateral bending

This beam-column is not continuously laterally restrained along the compression flange.
Steel Grade: A300W

                 Cross sectional strength

                 Segment (xx) 0.00 - 4000.00
START RESTRAINT:
 (Weak axis Euler buckling) rotation free, translation fixed
 (Lateral torsional buckling) restrained against torsion
 (Lateral torsional buckling) unrestrained against lateral bending
END RESTRAINT:
 (Weak axis Euler buckling) rotation free, translation fixed
 (Lateral torsional buckling) restrained against torsion
 (Lateral torsional buckling) unrestrained against lateral bending

Effective Length factor (Crx) : 1.00         Eff length (mm) : 4000.00
omega1x : 0.60                               omega2x : 1.75
kappax : -0.00                               omega1y : 0.60
Maximum Moment zz (kN.m): -43.61             Maximum Moment yy (kN.m): -0.00
Maximum Shear y (kN): -10.90                 Maximum Shear z (kN): -0.00
Maximum Axial (kN): -12.00                   Maximum Torsion (kN.m): -0.00

254x146x37     Cr (kN) = 1279.80             Mrx (kN.m) = 130.95             Mry (kN.m) = 32.13
               U1x = 1.00                    U1y = 1.00                      ratio = 0.29
               Moment ratio = 0.33

                 Overall member strength

                 Segment (xx) 0.00 - 4000.00
START RESTRAINT:
 (Weak axis Euler buckling) rotation free, translation fixed
 (Lateral torsional buckling) restrained against torsion
 (Lateral torsional buckling) unrestrained against lateral bending
END RESTRAINT:
 (Weak axis Euler buckling) rotation free, translation fixed
 (Lateral torsional buckling) restrained against torsion
 (Lateral torsional buckling) unrestrained against lateral bending
```



```
Effective Length factor (Crx) : 1.00            Eff length (mm) : 4000.00
omega1x : 0.60                                  omega2x : 1.75
kappax : -0.00                                  omega1y : 0.60
Maximum Moment zz (kN.m): -43.61                Maximum Moment yy (kN.m): -0.00
Maximum Shear y (kN): -10.90                    Maximum Shear z (kN): -0.00
Maximum Axial (kN): -12.00                      Maximum Torsion (kN.m): -0.00

254x146x37     Cr (x) (kN) = 1174.19            Mrx (kN.m) = 130.95         Mry (kN.m) = 32.13
               U1x = 0.60                       U1y = 0.61                  ratiox = 0.18
               Moment ratio = 0.33
```

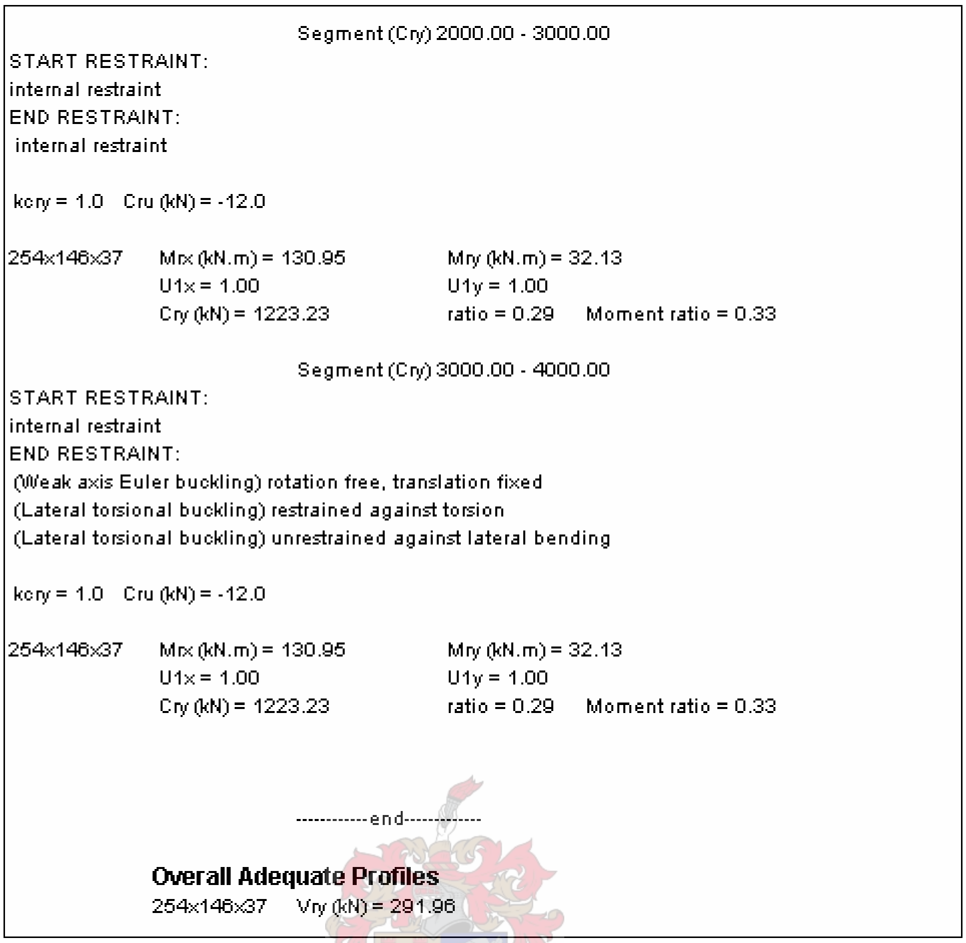
Lateral torsional buckling strength

Segment (Mcr) 0.00 - 2000.00

```
START RESTRAINT:
 (Weak axis Euler buckling) rotation free, translation fixed
 (Lateral torsional buckling) restrained against torsion
 (Lateral torsional buckling) unrestrained against lateral bending
END RESTRAINT:
 internal restraint

Effective Length factor (Mcr) : 1.00            Eff length (mm) : 2000.00
omega1x : 0.60                                  omega2x : 1.75
kappax : -0.00                                  omega1y : 0.60
Maximum Moment zz (kN.m): -21.80                Maximum Moment yy (kN.m): -0.00
Maximum Shear y (kN): -10.90                    Maximum Shear z (kN): -0.00
Maximum Axial (kN): -12.00                      Maximum Torsion (kN.m): -0.00

                              Segment (Cry) 0.00 - 2000.00
START RESTRAINT:
 (Weak axis Euler buckling) rotation free, translation fixed
 (Lateral torsional buckling) restrained against torsion
 (Lateral torsional buckling) unrestrained against lateral bending
END RESTRAINT:
 internal restraint

 kcry = 1.0   Cru (kN) = -12.0

254x146x37     Mrx (kN.m) = 130.95              Mry (kN.m) = 32.13
               U1x = 1.00                       U1y = 1.00
               Cry (kN) = 995.51                ratio = 0.15     Moment ratio = 0.17
```

Segment (Mcr) 2000.00 - 4000.00

```
START RESTRAINT:
internal restraint
END RESTRAINT:
 (Weak axis Euler buckling) rotation free, translation fixed
 (Lateral torsional buckling) restrained against torsion
 (Lateral torsional buckling) unrestrained against lateral bending

Effective Length factor (Mcr) : 1.00            Eff length (mm) : 2000.00
omega1x : 0.80                                  omega2x : 1.30
kappax : -0.50                                  omega1y : 0.60
Maximum Moment zz (kN.m): -43.61                Maximum Moment yy (kN.m): -0.00
Maximum Shear y (kN): -10.90                    Maximum Shear z (kN): -0.00
Maximum Axial (kN): -12.00                      Maximum Torsion (kN.m): -0.00
```



```
                              Segment (Cry) 2000.00 - 3000.00
START RESTRAINT:
internal restraint
END RESTRAINT:
 internal restraint

 kcry = 1.0   Cru (kN) = -12.0

254x146x37     Mrx (kN.m) = 130.95         Mry (kN.m) = 32.13
               U1x = 1.00                  U1y = 1.00
               Cry (kN) = 1223.23          ratio = 0.29     Moment ratio = 0.33

                              Segment (Cry) 3000.00 - 4000.00
START RESTRAINT:
internal restraint
END RESTRAINT:
 (Weak axis Euler buckling) rotation free, translation fixed
 (Lateral torsional buckling) restrained against torsion
 (Lateral torsional buckling) unrestrained against lateral bending

 kcry = 1.0   Cru (kN) = -12.0

254x146x37     Mrx (kN.m) = 130.95         Mry (kN.m) = 32.13
               U1x = 1.00                  U1y = 1.00
               Cry (kN) = 1223.23          ratio = 0.29     Moment ratio = 0.33



                              ------------end-------------

               Overall Adequate Profiles
               254x146x37     Vry (kN) = 291.96
```

Figure 9-32 Text design results for "DesignElement.3"

As can be seen from the textual design results for DesignElement.3, the “254x146x37” steel profile is adequate for the given conditions.

Hand calculated design: This section shows the hand calculated design results and comparison with the results from the application.

For the purpose of the hand calculated results, only the results of “DesignElement.3” will be investigated. The remaining members are similar to the previous example as shown in example 5. Thus “DesignElement.3” and the loaded profile of “254x146x37” will be hand calculated.

<u>Classification of profiles</u>

“254x146x37”

h = 256

b = 146.4

t_f = 10.9

t_w = 6.3

$$\frac{b}{2\cdot t_f}=\frac{146.1}{(21.8)}=<\frac{145}{\sqrt{300}}\rightarrow \text{class 1 flange}$$

C_u = 12 kN

$$C_y = A\cdot f_y = 4.07\times10^3(300) = 1221\text{kN}$$

$$\frac{h-2t_f}{t_w}=\frac{256-2(10.9)}{6.3}=37.17\le\frac{1100}{\sqrt{300}}\left(1-0.39\frac{12}{\Phi\cdot1221}\right)\rightarrow \text{class 1 web}$$

The profile is therefore a class 1 profile.

DesignElement.3

This element is not continuously braced along its weak axis, resulting in the test for lateral torsional buckling strength to be included in the design process. The member is taken to be part of a braced frame for the purpose of this example.

The effective length of the member for strong axis Euler buckling is taken as 1.0 due to the finite element results assumed to be from a second order analysis.. [1.0]

The effective length factor of the member for weak axis Euler buckling is 1.0 due to the restraint conditions applied [1.0]

The effective length factor of the member for lateral torsional buckling is 1.0 due to the restraint conditions applied. [1.0]

The capacity of the member is examined for the following strength cases and is based on the following interaction formula, $\frac{C_u}{C_r}+\frac{0.85\cdot U_{1x}\cdot M_{ux}}{M_{rx}}\le 1.0$.

- *Cross – sectional strength*

$C_r = \Phi\cdot A\cdot f_y = 0.9(4.74\times10^3)(300) = 1279.8\,\text{kN}$ [1279.8 kN]

$M_{rx} = \Phi\cdot Z_{plx}\cdot f_y = 0.9(485\times10^3)(300) = 130.95\,\text{kN.m}$ [130.95 kN.m]

$$U_{1x}=\frac{\omega_1}{1-\frac{C_u}{C_{ex}}}$$

$$\omega_1 = 0.6-0.4\cdot\kappa\ge0.4$$

The value of κ is equal to the ratio of the smaller end moment to the larger end moment. As can be seen from the bending moment diagram of this example, shown in Figure 9-31, the largest end moment appears at the end of the member with the other end equal to zero. This results in a value for κ to be equal to zero. [0.0]

$\therefore \omega_1 = 0.6 - 0.4(0.0) = 0.6$ [0.6]

$$C_{ex} = \frac{\pi^2 \cdot E \cdot I_x}{L_x} = \frac{\pi^2 (200\times10^3)(55.5\times10^6)}{(4000)^2} = 6847.04\,\text{kN}$$

$$\therefore U_{1x} = \frac{0.6}{1 - \dfrac{12}{6847.04}} = 0.601$$

This value may not be less than 1.0 for the cross sectional analysis. [1.0]

The result of the interaction is therefore,

$$\frac{12}{1279.8} + \frac{0.85(1.0)(43.61)}{130.95} = 0.29 \le 1.0$$ [0.29]

$$\frac{43.61}{130.95} = 0.33 \le 1.0$$ [0.33]

The profile is thus adequate in terms of cross sectional strength as illustrated by the interaction formula.

- *Overall member strength*

The compressive resistance of the member is calculated according to buckling about the strong axis, due to uniaxial strong axis bending.

$$C_{rx} = \Phi \cdot A \cdot f_y \left(1 + \lambda^{2n}\right)^{\frac{-1}{n}}$$

$$\lambda = \frac{K \cdot L_x}{r_x} \cdot \sqrt{\left(\frac{f_y}{\pi^2 \cdot E}\right)} = \frac{1.0(4000)}{108}\sqrt{\left(\frac{300}{\pi^2 (200\times10^3)}\right)} = 0.456$$

$$\therefore C_{rx} = 0.9(4.74\times10^3)(300)(1 + 0.456^{2\cdot1.34})^{\frac{-1}{1.34}} = 1174.19\,\text{kN}$$ [1174.19 kN]

$$M_{rx} = \Phi \cdot Z_{plx} \cdot f_y = 0.9(485\times10^3)(300) = 130.95\,\text{kN.m}$$ [130.95 kN.m]

$$U_{1x} = \frac{\omega_1}{1 - \dfrac{C_u}{C_{ex}}}$$

$$\omega_1 = 0.6 - 0.4 \cdot \kappa \ge 0.4$$

The value of κ is equal to the ratio of the smaller end moment to the larger end moment. As can be seen from the bending moment diagram of this example, shown in Figure 9-31, the largest end moment appears at the end of the member with the other end equal to zero. This results in a value for κ to be equal to zero. [0.0]

$\therefore \omega_1 = 0.6 - 0.4(0.0) = 0.6$ [0.6]

$$C_{ex} = \frac{\pi^2 \cdot E \cdot I_x}{L_x} = \frac{\pi^2(200\times10^3)(55.5\times10^6)}{(4000)^2} = 6847.04\,\text{kN}$$

$$\therefore U_{1x} = \frac{0.6}{1-\frac{12}{6847.04}} = 0.601$$ [0.60]

The result of the interaction formula is therefore,

$$\frac{12}{1174.19} + \frac{0.85(0.60)(43.61)}{130.95} = 0.18 \leq 1.0$$ [0.18]

$$\frac{43.61}{130.95} = 0.33 \leq 1.0$$ [0.33]

The profile is thus adequate in terms of cross sectional strength as illustrated by the interaction formula.

- *Lateral torsional buckling strength*

The compressive resistance of the member, as used in the interaction formula, calculated is dependent on weak axis or torsional flexural buckling. Due to the presence of internal restraints present along the length of the member, the Design Element is divided up into segments at the positions of the internal restraints. Figure 9-33 illustrates these segments.

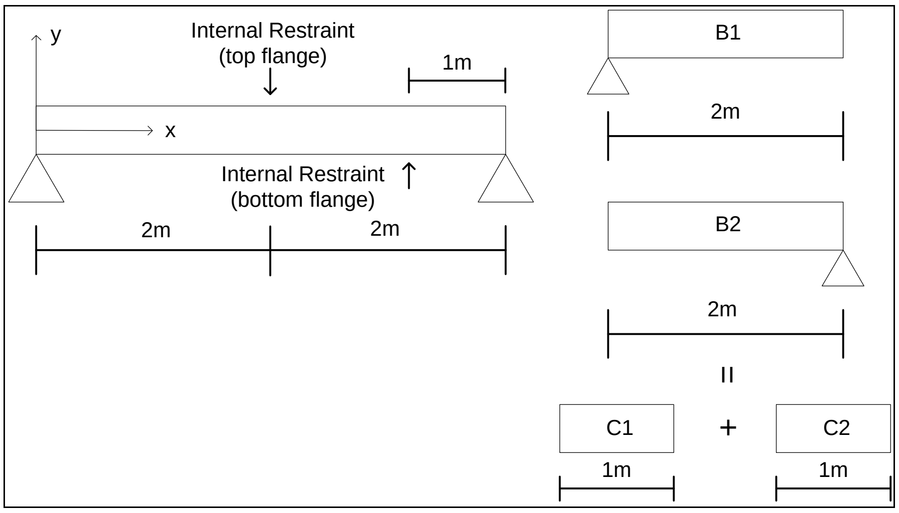


Figure 9-33 Illustration of the segments of "DesignElement.3"

As can be seen from the above figure, the internal restraint at midpoint divides the member into two main segments, B1 and B2. These segments represent the lateral torsional buckling lengths as well as Euler buckling lengths about the weak axis of the member. The second internal restraint does not create additional bending segments due to the fact that there is no compressive bending at the bottom flange of the steel member at that point. The bending segment B2 is has two compressive segments, namely C1 and C2. These represent the Euler buckling lengths of B2 due to the second internal restraint. All these buckling lengths are illustrated in the textual design results for "DesignElement.3".

The Euler buckling strength of segment B1 is calculated as follows

$$C_{ry} = \Phi \cdot A \cdot f_y \left(1+\lambda^{2n}\right)^{\frac{-1}{n}}$$

$$\lambda = \frac{K \cdot L_y}{r_y} \cdot \sqrt{\left(\frac{f_y}{\pi^2 \cdot E}\right)} = \frac{1.0(2000)}{34.7}\sqrt{\left(\frac{300}{\pi^2(200\times10^3)}\right)} = 0.711$$

$$\therefore C_{ry} = 0.9(4.74\times10^3)(300)(1+0.711^{2\cdot1.34})^{\frac{-1}{1.34}} = 995.51\,\text{kN} \qquad [995.51\text{ kN}]$$

The bending resistance of the member is based on lateral torsional buckling and not on cross sectional resistance.

$$M_{cr} = \frac{\omega_2 \cdot \pi}{K \cdot L}\sqrt{EI_y GJ + \left(\frac{\pi E}{K \cdot L}\right)^2 I_y C_w}$$

The value of κ is calculated from the end moments.

$$\kappa = \frac{0}{21.81} = 0\,. \qquad [0.0]$$

$$\omega_2 = 1.75 + 1.05 \cdot \kappa + 0.3 \cdot \kappa^2 = 1.75 + 1.05(0) + 0.3(0) = 1.75 \qquad [1.0]$$

$$\therefore \frac{1.75(\pi)}{(2000)}\sqrt{200\times10^3(5.71\times10^6)(77\times10^3)(155\times10^3) + \left(\frac{(\pi)(200\times10^3)}{(1.0)2000}\right)^2 (5.71\times10^6)(85.7\times10^9)} =$$

684.06 kN.m > 0.67 M_p

This value is larger than the value of 0.67M_p that is calculated as 97.49 kN.m. The bending resistance of the segment is then calculated as follows:

$$M_r = 1.15 \cdot \Phi \cdot M_p \cdot \left(1 - \frac{0.28 \cdot M_p}{M_{cr}}\right) \leq \Phi \cdot M_p$$

$$\therefore = 1.15(0.9)(145.5)\left(1-\frac{0.28(145.5)}{684.06}\right) = 141.62 \text{ kN.m} > 130.95.\text{kN.m}$$

Due to the fact that the value calculated is larger than Φ M_p, the value of bending resistance is calculated as Φ $M_{p,}$ as discussed in chapter 4.

$\therefore M_{rx} = \Phi \cdot M_p = 0.9(145.5) = 130.95\text{kN.m}$ [130.95 kN.m]

$$U_{1x} = \frac{\omega_1}{1-\dfrac{C_u}{C_{ex}}}$$

$$\omega_1 = 0.6 - 0.4\cdot\kappa \geq 0.4$$

The value of κ is equal to the ratio of the smaller end moment to the larger end moment. This results in a value for κ to be equal to zero. [0.0]

$\therefore \omega_1 = 0.6 - 0.4(0.0) = 0.6$ [0.6]

$$C_{ex} = \frac{\pi^2\cdot E\cdot I_x}{L_x} = \frac{\pi^2(200\times10^3)(55.5\times10^6)}{(2000)^2} = 27388.15\,\text{kN}$$

$$\therefore U_{1x} = \frac{0.6}{1-\dfrac{12}{27388.15}} = 0.6$$

This value may not be less than 1.0 for this particular test. [1.0]

The result of the interaction is therefore,

$\dfrac{12}{995.51} + \dfrac{0.85(1.0)(21.8)}{130.95} = 0.15 \leq 1.0$ [0.15]

$\dfrac{21.8}{130.95} = 0.166 \leq 1.0$ [0.17]

This segment is therefore adequate in terms of lateral torsional buckling strength with the chosen profile provided.

The lateral torsional buckling strength of segment B2 is calculated as follows

$$C_{ry} = \Phi\cdot A\cdot f_y\left(1+\lambda^{2n}\right)^{\frac{-1}{n}}$$

As can be seen in Figure 9-33, B2 is further subdivided into two smaller segments C1 and C2. These segments have the same length of 2m. The calculation of the compressive resistance of the member is based on these smaller segments.

The compressive resistance of C1 and C2 is as follows:

$$\lambda = \frac{K \cdot L_y}{r_y} \cdot \sqrt{\left(\frac{f_y}{\pi^2 \cdot E}\right)} = \frac{1.0(1000)}{34.7}\sqrt{\left(\frac{300}{\pi^2(200\times10^3)}\right)} = 0.356$$

$$\therefore C_{ry} = 0.9(4.74\times10^3)(300)(1+0.356^{2\cdot1.34})^{\frac{-1}{1.34}} = 1223.09\,\text{kN}$$ [1223.23 kN]

The bending resistance of the member is based on lateral torsional buckling and not on cross sectional resistance.

$$M_{cr} = \frac{\omega_2 \cdot \pi}{K \cdot L}\sqrt{EI_yGJ + \left(\frac{\pi E}{K \cdot L}\right)^2 I_yC_w}$$

The value of κ is calculated from the end moments.

$$\kappa = -\frac{21.81}{43.61} = -0.5$$ [-0.5]

$$\omega_2 = 1.75 + 1.05 \cdot \kappa + 0.3 \cdot \kappa^2 = 1.75 + 1.05(-0.5) + 0.3(-0.5)^2 = 1.3$$ [1.3]

$$\therefore \frac{1.3(\pi)}{(2000)}\sqrt{200\times10^3(5.71\times10^6)(77\times10^3)(155\times10^3) + \left(\frac{(\pi)(200\times10^3)}{(1.0)2000}\right)^2(5.71\times10^6)(85.7\times10^9)}$$

= 508 kN.m > 0.67 M_p

This value is larger than the value of 0.67M_p that is calculated as 97.49 kN.m. The bending resistance of the segment is then calculated as follows:

$$M_r = 1.15 \cdot \Phi \cdot M_p \cdot \left(1 - \frac{0.28 \cdot M_p}{M_{cr}}\right) \le \Phi \cdot M_p$$

$$\therefore = 1.15(0.9)(145.5)\left(1 - \frac{0.28(145.5)}{508}\right) = 138.51\text{ kN.m} > 130.95\text{ kN.m}$$

Due to the fact that the value calculated is larger than Φ M_p, the value of the bending resistance is calculated as Φ M_p as discussed in chapter 4.

$$\therefore M_{rx} = \Phi \cdot M_p = 0.9(145.5) = 130.95\,\text{kN.m}$$ [81.27 kN.m]

$$U_{1x} = \frac{\omega_1}{1 - \dfrac{C_u}{C_{ex}}}$$

$$\omega_1 = 0.6 - 0.4 \cdot \kappa \ge 0.4$$

The value of κ is equal to the ratio of the smaller end moment to the larger end moment. This value was calculated to be -0.5.

$\therefore \omega_1 = 0.6 - 0.4(1.3) = 0.8$ [0.8]

$$C_{ex} = \frac{\pi^2 \cdot E \cdot I_x}{L_x} = \frac{\pi^2(200\times10^3)(55.5\times10^6)}{(2000)^2} = 27388.15\,kN$$

$$\therefore U_{1x} = \frac{0.8}{1-\frac{12}{27388.15}} = 0.8$$

This value may not be less than 1.0 for this particular test. [1.0]

The result of the interaction is therefore,

$$\frac{12}{1223.23} + \frac{0.85(1.0)(43.61)}{130.95} = 0.29 \le 1.0$$ [0.29]

$$\frac{43.16}{130.95} = 0.33 \le 1.0$$ [0.33]

This segment is therefore adequate in terms of lateral torsional buckling strength with the chosen profile provided.

- Shear

The shear resistance of the chosen profile is calculated as follows:

$V_r = \Phi \cdot A_v \cdot f_s$

The effective shear area, A_v, is calculated by the formula $h(t_w)$. This is illustrated as follows:

$A_v = h \cdot t_w = 256(6.3) = 1612.8\,mm^2$

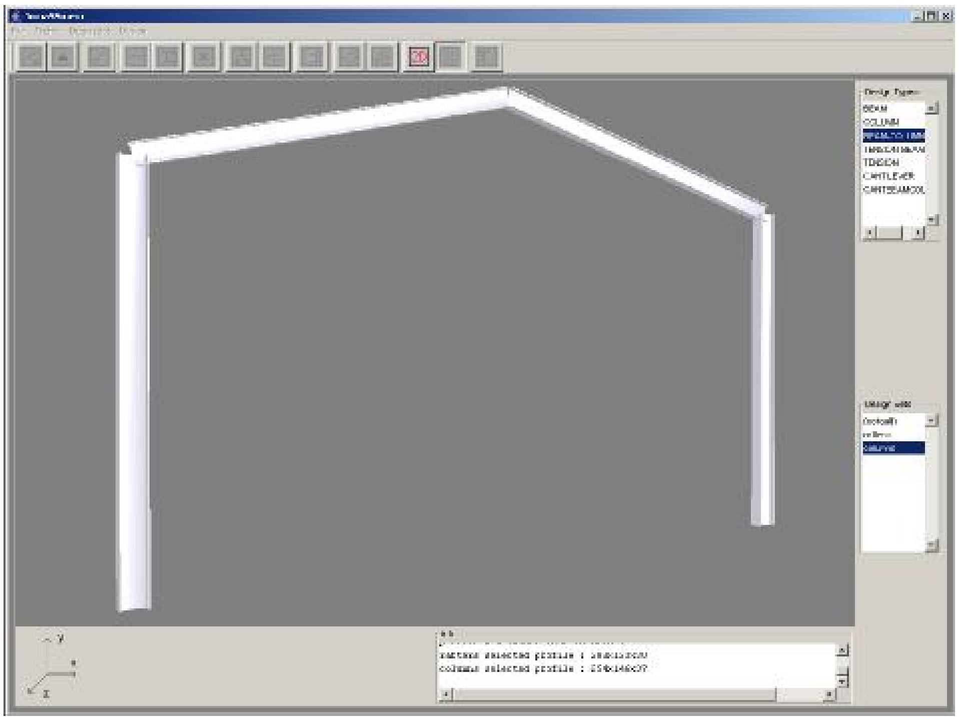
The shear strength, f_s, is calculated by the following formula:

$f_s = 0.66 \cdot f_y = 0.66(300) = 198\,MPa$

$\therefore V_r = 0.9(1612.8)(198) = 290\,kN > 10.9\ kN$ [291.96 kN]

From all the hand calculated tests, the chosen profile of “254x146x37” is adequate in terms of strength. This is indicated under the “overall adequate profiles” heading of the text results.

Figure 9-34 illustrates the chosen profiles in 3D.



```
rafters selected profile : [illegible]
columns selected profile : 254x146x37
```

Figure 9-34 Illustration of the 3D view for example 6

10 Conclusions and Recommendations

10.1 Conclusions

The criteria for success defined in the synopsis were met in this thesis.

A detailed specification for designing and representing structural steel members was developed. The design methods of the specification were developed according to the new South African code, namely *SANS 10162 Code of Practice for the Structural Use of Steel: Part1: Limit States Design of hot – rolled steelwork – 2005.*

An object oriented framework and associated graphical user interface (GUI) for designing the structural members were developed and implemented. The primary objectives of the framework and GUI defined in the synopsis were achieved. They were as follows:

Member Design Model: The development of a separate structural member design model that focuses solely on structural member design. This model makes use of the data from the associated finite element model and applies it in the relevant manner to the design code.

Structural Members: Special elements were developed to represent structural members from a design perspective. These special elements were called Design Elements. Design Elements were created to be independent of the topology of the finite element model, yet dependent on the geometry of that model. This enables more control over physical member lengths and end points. These special elements were developed to be generic in terms of material type and thus independent of design code. Consequently, Design Elements can be specialized for any material type, e.g. steel, concrete, timber etc.

Structural Steel Members: An extension of a Design Element, namely an SSDesignElement, was created for the specific design of hot – rolled structural steel members. This element was created to be capable of being accountable for the design procedures prescribed in *SANS 10162: Part1: Limit States Design of hot – rolled steelwork – 2005.*

End conditions: End conditions for structural steel members (SSDesign Elements) were developed to be independent of the end conditions as stipulated in the finite element model. Finite element end conditions refer to the restricting of finite element degrees of freedom. This was viewed as inadequate for design purpose. Restraint conditions conforming to the implemented design standards and physical end connection were implemented.

Framework: The framework and GUI were built on an existing architecture that allows for structural analysis, structural steel member design and connection design within a single application. Forces and moments for both member design and connection design are obtained directly from the structural analysis model, and member data required for connection design is, in turn, directly available from the member design model. This ensures the consistency and effectiveness of the complete design process. The underlying architecture also supports collaboration in a communication network.

Conclusion: From the above it is concluded that the objectives of the study have been met.

10.2 Recommendations

Recommendations stemming from the results of this study are:

- A Design Element could be produced to represent common repetitive members in a structure. This Design Element should be able to be “copied” over the repetitive members in a structure in place of creating new ones.
- Axial resistance of a Design Element could include the resistance of its end conditions due to the connections used.
- Allowance for changes in the topology in a finite element model, e.g. number of finite elements used to represent a member, number of nodes, etc, without having to recreate the design model. The Design Model could therefore update itself and allow for a non static finite element model.
- The introduction of the cost and availability of steel profiles that are to be used in a structure. This factor could be included in the overall design of a structure and thus determine the overall outcome.
- The automatic determination of design type. The design type of Design Elements could be determined automatically through inspection of the internal loading and other factors without having to manually select a design type.
- Allowance for manually setting effective length factors of the design members.

References

1. South African Institute of Steel Construction. **South African Steel Construction Handbook**. Third edition, Johannesburg, SAISC, 1999.
2. SANS 10162: Part 1: 2005: **Code of practice for the structural use of steel: Limit States Design of hot rolled steelwork**, South Africa, SANS, 2005.
3. Horstmann, Cay S. **Computing Concepts with JAVA 2 Essentials**. John Wiley and Sons, 2000.
4. Eckstein, R, Loy, M and Wood, D. **JAVA Swing**. Sebastopol, O'Reilly & Associates, 1998.
5. Eckel, B. **Thinking in JAVA** second edition. Prentice Hall 2000.
6. Cook, D. R., Malkus, D.S., Plesha, M.E., Witt, R.J. **Concepts and Applications of Finite Element Analysis** fourth edition. John Wiley and sons, 2002.
7. Popov, E.P. **Engineering Mechanics of Solids** second edition. Prentice Hall, 1999.
8. Lafore, R. **Data Structures and Algorithms in Java**, SAMS 1998.

Appendix A Model Files

Model file for *beam.model*.

```
node n1 {000. 000. 0.};
node n2 {200. 000. 0.};
node n3 {400. 000. 0.};
material mat 200e9 0.3 7850.;
section sec 2.85e-3 19.4e-6 1.42e-6;
frame f1 {n1,n2} mat sec 0 true;
frame f2 {n2,n3} mat sec 0 true;
support s1 n1 {x,y,z,xx};
support s2 n3 {x,y,z};
nload l1 n2 50000 {0,-1.,0};
lcase LoadCase A;
addloadtoloadcase l1 LoadCase A;
analyse;
setresult LoadCase A;
store beam.model;
```

Model file for *column.model*.

```
node n1 {000. 000. 0.};
node n2 {000. 200. 0.};
node n3 {000. 400. 0.};
material mat 200e9 0.3 7850.;
section sec 2.85e-3 19.4e-6 1.42e-6;
frame f1 {n1,n2} mat sec 0 false;
frame f2 {n2,n3} mat sec 0 false;
support s1 n1 {x,y,yy};
nload l1 n3 50000 {0,-1.,0};
lcase LoadCase A;
addloadtoloadcase l1 LoadCase A;
analyse;
setresult LoadCase A;
store column.model;
```

Model file for *column2.model*.

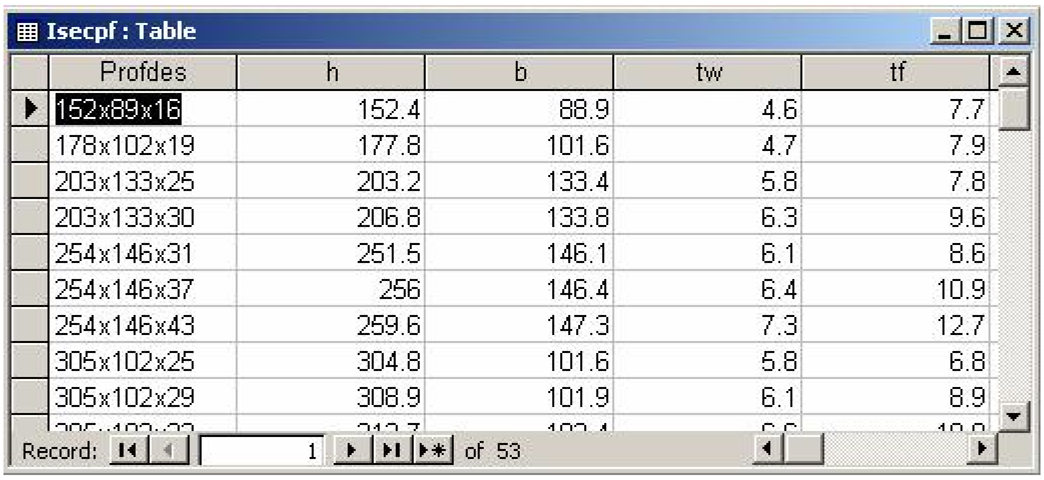
```
node n1 {000. 000. 0.};
node n2 {000. 200. 0.};
node n3 {000. 400. 0.};
material mat 200e9 0.3 7850.;
section sec 2.85e-3 19.4e-6 1.42e-6;
frame f1 {n1,n2} mat sec 0 false;
frame f2 {n2,n3} mat sec 0 false;
support s1 n1 {x,y,zz};
nload l1 n3 50000 {0.2,-1.,0};
lcase LoadCase A;
addloadtoloadcase l1 LoadCase A;
analyse;
setresult LoadCase A;
store column2.model;
```

Model file for *frame.model*.

```
node n1 {000. 000. 0.};
node n2 {000. 200. 0.};
node n3 {000. 400. 0.};
node n4 {200. 450. 0.};
node n5 {400. 500. 0.};
node n6 {600. 450. 0.};
node n7 {800. 400. 0.};
node n8 {800. 200. 0.};
node n9 {800. 000. 0.};
material mat 200e9 0.3 7850.;
section sec 2.85e-3 19.4e-6 1.42e-6;
frame f1 {n1,n2} mat sec 0 false;
frame f2 {n2,n3} mat sec 0 false;
frame f3 {n3,n4} mat sec 0 false;
frame f4 {n4,n5} mat sec 0 false;
frame f5 {n5,n6} mat sec 0 false;
frame f6 {n6,n7} mat sec 0 false;
frame f7 {n7,n8} mat sec 0 false;
frame f8 {n8,n9} mat sec 0 false;
support s1 n1 {x,y};
support s2 n9 {x,y};
nload l1 n3 80000 {0.3,-1.,0};
lcase LoadCase A;
addloadtoloadcase l1 LoadCase A;
analyse;
setresult LoadCase A;
store frame.model;
```

Appendix B Database Tables

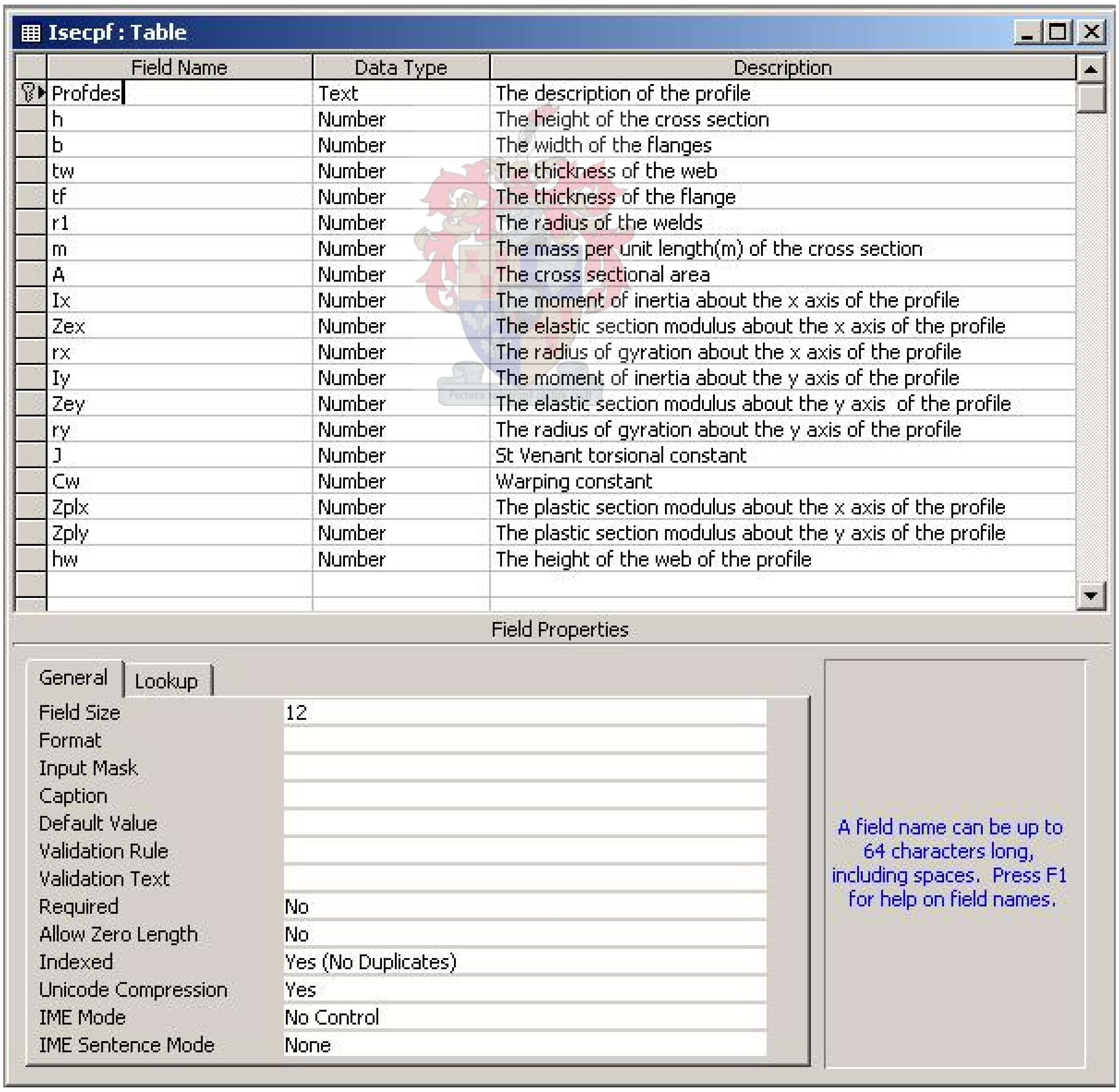
Steel Profiles: (a parallel flange I profile is displayed here as an example**)**



Isecpf : Table

Profdes	h	b	tw	tf
152x89x16	152.4	88.9	4.6	7.7
178x102x19	177.8	101.6	4.7	7.9
203x133x25	203.2	133.4	5.8	7.8
203x133x30	206.8	133.8	6.3	9.6
254x146x31	251.5	146.1	6.1	8.6
254x146x37	256	146.4	6.4	10.9
254x146x43	259.6	147.3	7.3	12.7
305x102x25	304.8	101.6	5.8	6.8
305x102x29	308.9	101.9	6.1	8.9

Record: 1 of 53



Isecpf : Table

Field Name	Data Type	Description
Profdes	Text	The description of the profile
h	Number	The height of the cross section
b	Number	The width of the flanges
tw	Number	The thickness of the web
tf	Number	The thickness of the flange
r1	Number	The radius of the welds
m	Number	The mass per unit length(m) of the cross section
A	Number	The cross sectional area
Ix	Number	The moment of inertia about the x axis of the profile
Zex	Number	The elastic section modulus about the x axis of the profile
rx	Number	The radius of gyration about the x axis of the profile
Iy	Number	The moment of inertia about the y axis of the profile
Zey	Number	The elastic section modulus about the y axis of the profile
ry	Number	The radius of gyration about the y axis of the profile
J	Number	St Venant torsional constant
Cw	Number	Warping constant
Zplx	Number	The plastic section modulus about the x axis of the profile
Zply	Number	The plastic section modulus about the y axis of the profile
hw	Number	The height of the web of the profile

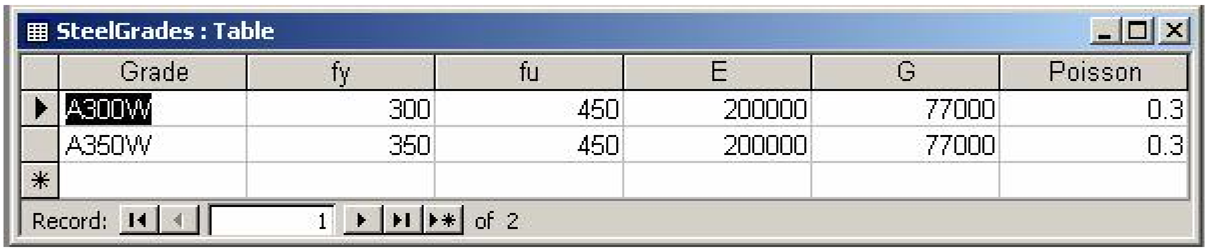
Field Properties

General | Lookup

Field Size	12
Format	
Input Mask	
Caption	
Default Value	
Validation Rule	
Validation Text	
Required	No
Allow Zero Length	No
Indexed	Yes (No Duplicates)
Unicode Compression	Yes
IME Mode	No Control
IME Sentence Mode	None

A field name can be up to 64 characters long, including spaces. Press F1 for help on field names.

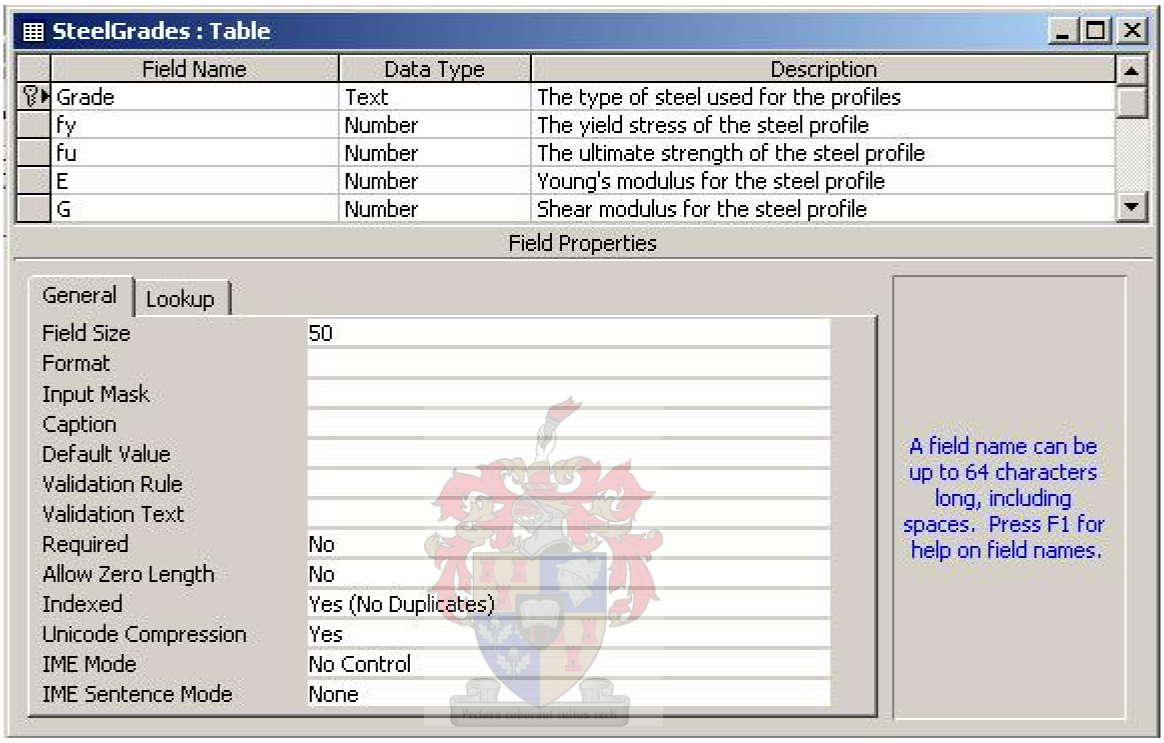
Steel Grades:



SteelGrades : Table

Grade	fy	fu	E	G	Poisson
A300W	300	450	200000	77000	0.3
A350W	350	450	200000	77000	0.3

Record: 1 of 2



SteelGrades : Table

Field Name	Data Type	Description
Grade	Text	The type of steel used for the profiles
fy	Number	The yield stress of the steel profile
fu	Number	The ultimate strength of the steel profile
E	Number	Young's modulus for the steel profile
G	Number	Shear modulus for the steel profile

Field Properties

General | Lookup

Field Size	50
Format	
Input Mask	
Caption	
Default Value	
Validation Rule	
Validation Text	
Required	No
Allow Zero Length	No
Indexed	Yes (No Duplicates)
Unicode Compression	Yes
IME Mode	No Control
IME Sentence Mode	None

A field name can be up to 64 characters long, including spaces. Press F1 for help on field names.