OBJECT ORIENTED PROGRAMMING AND CAD TECHNOLOGY APPLIED TO A FINITE ELEMENT TECHNIQUE FOR THE LIMIT ANALYSIS OF AXISYMMETRICAL PRESSURE VESSELS

José Ricardo Queiroz Franco
Department of Structural Engineering, School of Engineering, Federal University of Minas Gerais, Av. do Contorno, 842, Brazil, CEP: 30110-060 , e-mail:franco@dees.ufmg.br

Felício Bruzzi Barros
Department of Structural Engineering, School of Engineering, Federal University of Minas Gerais, Av. do Contorno, 842, Brazil, CEP: 30110-060 , e-mail:felicio@cadtec.dees.ufmg.br

Alexandre Balabram
EMBRAER, São José dos Campos, S.P., Brazil, e-mail:balabram@embraer.com.br

Fernando Poinho Malard
Department of Structural Engineering, School of Engineering, Federal University of Minas Gerais, Av. do Contorno, 842, Brazil, CEP: 30110-060 , e-mail:fpmalard@cadtec.dees.ufmg.br

Abstract: *This paper concerns with the aspects of the Object Oriented Programming used to develop a Finite Element technique for limit analysis of axisymmetrical pressure vessels. The concepts of inheritance, polimorphism, encapsulation and modularity are extensively explored. A friendly user interface based on the MFC library classes was developed to run the analysis processor integrated with two graphical modules defined as modeler (pre-processing) and post-processor. The problem used to illustrated the limit analysis technique implemented is a standard ASME torispherical vessel subjected to internal pressure. The obtained results are compared tiwh other numerical and analytical solutions.*

Key Words: *Object Oriented Programming, CAD, Shakedown and Limit Analysis, Finite Element Method*

1. Introduction

Object Oriented Programming (OOP) associated to a new CAD technology for modeling the geometry and discretizing axisymmetrical shells into finite elements have been applied for the limit analysis of pressure vessels. The global scope of the work proposes to automate the three modules of the analysis process, which consists of modeling/discretization (pre-processing), analysis (processing) and post-processing graphical results for ultimate loading in pressure vessels. This paper concerns only with the aspects of the OOP used to develop a technique for limit analysis, although the dependency of this module upon an easy communication with the two graphical processors had to be considered. A data file interfacing data from the geometrical modeler to the analysis code and a data file exporting data from this code to the graphical post-processor are essential procedures for the automation of the process as a whole.

The finite element limit analysis formulation was developed according to the work of T. Zimmermann and Bomme (1992); T. Zimmermann (1993), following the principle of OOP and his concepts of classes derivation. This choice was motivated by the extensive documentation available and the possibility of re-using several classes appropriated to finite element analysis. Three types of classes were used in this work. Most of them are originated from T. Zimmermann (1993), without any alteration and other required some changes. New classes had also to be developed to fulfill specific requirements of limit analysis problems.

A friendly user interface based on the MFC library classes was developed to run the analysis processor. A simple button on this graphical interface allows the user to interact with the pre-processor by chosing a data file generated by the modeler. This file contains all the necessary geometrical and discretized data for the limit analysis. A second button can be used to name and export an output file with the analysis results and the necessary data file for the graphical post-processing of the results. Finally, the OOP system is applied to a limit analysis problem involving a torispherical vessel for various thickness and the results are compared with other numerical and analytical solutions.

2. Kinematic Formulation of Limit Analysis

The formulation of the limit analysis problem for pressure vessels has been proposed by Franco and Ponter (1997a). Application of the kinematical principles of the Koiter's theorem to discretized shells can be reduced to a minimization problem briefly discussed as follows.For the limited load factor $\kappa_L \geq \kappa$ minimize:

$$\kappa = \inf_{\dot{\lambda}} \int\limits_{\Omega} (\boldsymbol{\sigma}^c)^T \dot{\boldsymbol{\varepsilon}}^c(\dot{\lambda}) d\Omega \tag{1}$$

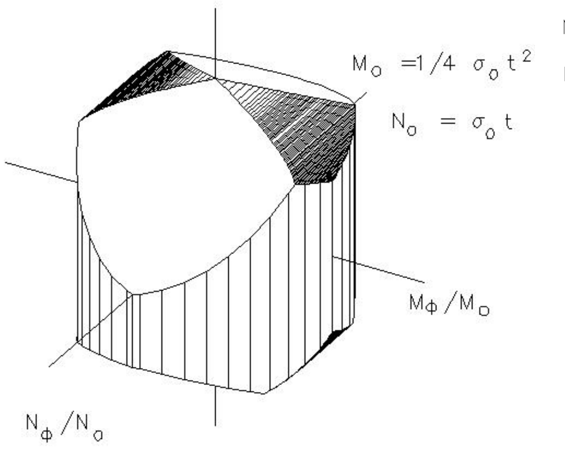
in the domain Ω, where $\boldsymbol{\sigma}^c(t)$ is a state of stress on the yield surface associated with the pure plastic strain rates $\dot{\boldsymbol{\varepsilon}}^p(t)$ and $\hat{\boldsymbol{\sigma}}^\theta(x,t)$ denotes the elastic stresses due to time dependent loads. The optimization problem is constrained by prescribed boundary conditions and the additional global constraint

$$\int_S \boldsymbol{p}_i \dot{\boldsymbol{U}}^{c_i} dS = 1 \tag{2}$$

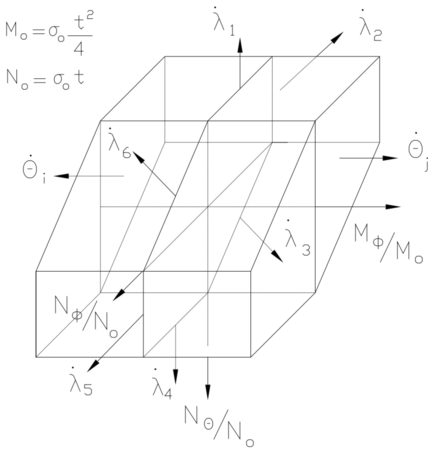
where $\boldsymbol{p}_i$ is the traction, $\dot{\boldsymbol{U}}_i^c$ is the global displacement rate field and the index i refers to the i^{th} element.

2.1. Constitutive Model

The yield surface used for the present analysis Fig. 1(b) was constructed by linearizing the exact yield surface for thin cylindrical shell represented in the Fig. 1(a). This 3D surfaces corresponds to a cylindrical element subjected to a complete set of forces (meridional and circunferencial normal forces, N_ϕ and N_θ, as well as the meridional moment M_ϕ) obtained in Drucker and Shield (1958); Onat (1955). In Fig. 1(b), the primary membrane behaviour of thin shells and the important characteristic that bending occurs only at certain local positions allow the separations of distinct volumes of the shell where either bending or membrane behaviour are independently relevant. These volumes are the plastic hinges where curvatures $(\kappa_\theta, \kappa_\phi)$ are unrestricted and the regions between two adjacent hinges where only circunferencial (ε_θ) and meridional (ε_ϕ) strain occur, (Drucker, 1953). For application of the upper bound formulation (1), a displacement



(a) Exact yield surface



(b) Circunsribed prism yield surface

Figure 1. 3D yield surface for thin cylindrical shell

field has been chosen consisting of discrete hinges at the nodes of a finite element subdivision and zero curvature within the element (hinge-cone displacement field), separating membrane action from bending action consistent with the one moment limited interaction yield surface of Fig. 1(b).

A finite number of yield functions ϕ_k representing the eight planes defining the hexagon prism (Fig. 1(b)) is used to describe the yield conditions as:

$$\phi_k = \phi_k(N_\phi, N_\theta, M_\phi) \leq 0 \quad \text{for } k = 1, 2, \ldots, 8 \tag{3}$$

Yielding will occur when one or more functions ϕ_k equals to zero which corresponds to a state of stress on the yield surface: negative values of all the yield functions define the elastic domain. The flow rule associated with the hexagonal prism yield surface relates the stress rate to plastic strain $(\dot\kappa^p_\phi, \dot\varepsilon^p_\phi, \dot\varepsilon^p_\theta)$ for the separate volumes of the shell as:

$$\dot\kappa^p_\phi = \dot\theta_k \frac{\partial \phi_k}{\partial M_\phi} \quad \text{at hinges circles} \quad \dot\varepsilon^p_\phi = \textstyle\sum_{k=1}^{8} \dot\lambda_k \frac{\partial \phi_k}{\partial N_\phi} \quad \dot\varepsilon^p_\theta = \displaystyle\sum_{k=1}^{8} \dot\lambda_k \frac{\partial \phi_k}{\partial N_\theta} \quad \text{between hinges} \tag{4}$$

where $\dot\theta_k$ represent free positive rotations rate at the hinges circles induced by the plastic unrestricted curvatures rates and $\dot\lambda_k$ are positive plastic multipliers.

2.2. Finite Element Discretization

The relations (4) can be put together by the following representation, described as in Franco and Ponter (1997b), for each finite element i, as shown in Figure 2:

$$\dot{\boldsymbol{\varepsilon}}^p_i = \left\{ \begin{array}{c} \dot\varepsilon^p_\phi(s,\phi) \\ \dot\varepsilon^p_\theta(s,\phi) \end{array} \right\}^i = \boldsymbol{N}\dot{\boldsymbol{\lambda}}^i(s) \text{ for k=1,..6} \tag{5}$$

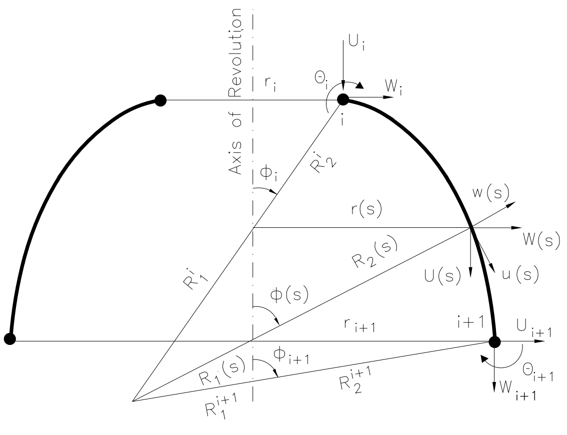


Figure 2. Global and Local Displacements for Thin Shells of Revolution

where $\dot{\boldsymbol{\lambda}}^i(s) = \left\{ \begin{array}{ccc} \dot{\lambda}_1^i(s) & \dots & \dot{\lambda}_8^i(s) \end{array} \right\}^T$ and $\boldsymbol{N}$ is a matrix that relates the plastic strains to the plastic multipliers, according to the flow law, Figure 1(b).

The plastic multipliers are interpolated in terms of nodal values $\dot{\lambda}_k^n$ using the matrix $\boldsymbol{\Lambda}(s)$ of lagrangian linear functions:

$$\dot{\boldsymbol{\lambda}}_k^i(s) = \boldsymbol{\Lambda}(s)\dot{\boldsymbol{\lambda}}_k^n \tag{6}$$

The localize nature of the nodal plastic hinges requires no interpolation. The flow rule (first expression of (4)), describes the unrestricted curvature rates $\dot{\kappa}_\phi$ in terms of the rotation rates $\dot{\theta}$, which are defined in terms of the velocity field $\dot{\boldsymbol{U}}_i$. A major set of nodal constraints for the minimization problem is then defined as:

$$\dot{\theta}_i = \dot{U}_i'^{(-)} + \dot{U}_i'^{(+)} \tag{7}$$

This allows the separation of bending at the nodes from membrane behaviour within the element when the shell is discretized.

In order to determine the total strain rate, $\dot{\boldsymbol{\varepsilon}}_i$, within the element only, the following relationship is employed:

$$\boldsymbol{\varepsilon}_i = \left\{ \begin{array}{c} \dot{\varepsilon}_\phi \\ \dot{\varepsilon}_\theta \end{array} \right\}^i = \boldsymbol{B}^i \dot{\boldsymbol{U}}_n^i \tag{8}$$

where $\boldsymbol{B}^i$ is a matrix operator stated in Franco and Ponter (1997b) to relate the velocity field $\dot{\boldsymbol{U}}$ and the total strain rate $\dot{\boldsymbol{\varepsilon}}_i$.

Finally, the reduction of the upper bound theorem, equation (1), to a Linear Programming problem requires that the two separate descriptions of the strain fields given by equations (8) and (5), must be consistent with each other at both nodes and within the element. We force these to be compatible by constructing the L^2-projection of the plastic strain rates onto the space of functions spanned by the displacement-rate shape functions. This simple process proposed by Oden and Brauchli (1971); Oden and Reddy (1973) was extensively discussed in Franco and Ponter (1997a) and used by Franco et al. (1995). We assume that for elastic perfectly plastic materials the stress remains constant during collapse due to the pure plastic strain rates, i.e.,

$$\dot{\boldsymbol{\sigma}}^i = \boldsymbol{D}^i \left(\dot{\boldsymbol{\varepsilon}}_i - \dot{\boldsymbol{\varepsilon}}_i^p \right) = \boldsymbol{0} \tag{9}$$

This assumption requires that $\dot{\boldsymbol{\varepsilon}}_i = \dot{\boldsymbol{\varepsilon}}_i^p$ and the element stress rate field is described in terms of nodal values as

$$\dot{\boldsymbol{\sigma}}^i = \boldsymbol{\phi}^i \dot{\boldsymbol{\sigma}}_n^i \tag{10}$$

where $\boldsymbol{\phi}$ is a suitable interpolation matrix defined by

$$\boldsymbol{\phi}^i = \boldsymbol{R}^i \boldsymbol{H}^i \tag{11}$$

The minimization of the local strain rate residual

$$\boldsymbol{e}_\varepsilon^i = \left(\dot{\boldsymbol{\varepsilon}} - \dot{\boldsymbol{\varepsilon}}^p \right)^i \tag{12}$$

obtained from numerical procedures, is achieved by setting the derivative of the energy norm, with respect to stress nodal values $\dot{\boldsymbol{\sigma}}_n^i$, to zero. The energy norm can be defined locally as

$$I = \| \boldsymbol{e} \|_i^2 = \int_{\Omega_i} \dot{\boldsymbol{\sigma}}^T \boldsymbol{e}_\varepsilon d\Omega = \int_{\Omega_i} \boldsymbol{e}_\varepsilon^T \boldsymbol{D} \boldsymbol{e}_\varepsilon d\Omega \tag{13}$$

Taking the derivative $\dfrac{dI}{d\dot{\boldsymbol{\sigma}}_n^i} = \mathbf{0}$, as shown in the Franco et al. (1997), gives the consistency relationship

$$\dot{\boldsymbol{U}}_n^i = \boldsymbol{L}\dot{\boldsymbol{\lambda}}_k^n \text{ where } \boldsymbol{L} = \boldsymbol{H}_i^T \int\limits_{\Omega_i} \boldsymbol{R}_i^T \boldsymbol{K}(s) d\Omega \tag{14}$$

with

$$\boldsymbol{H}_i = \left\{ \int\limits_{\Omega_i} \boldsymbol{B}_i^T \boldsymbol{R}_i d\Omega \right\}^{-1} \qquad \boldsymbol{K}(s) = \boldsymbol{N}\boldsymbol{\Lambda}(s) \tag{15}$$

being a symmetric, non-singular matrix and the arbitrary matrix $\boldsymbol{R}$ with, the same size as $\boldsymbol{B}$, has components which are a set of functions obtained by a standard Galerkin procedure as shown in Franco and Ponter (1997a); Franco et al. (1995). The solution of the optimization problem obtained using such a consistent relationship gives rise to an improved plastic strain rate field $\dot{\varepsilon}_h^p$ at collapse represented by the plastic multipliers.

3. Finite Element Algorithm

The extended upper bound theorem, defined in Section 2, corresponds to a optimization problem. Assuming that k is the optimal shakedown or limit load factor, the cost function Eq. 1 can be scaled by assuming that:

$$\int\limits_S p_i U_i^c dS = 1 \tag{16}$$

which adds to the problem as a global constraint involving all the finite elements. The same problem can now be formulated as; for $k^s \geq k$ **minimize**

$$k^s = \int\limits_0^T dt \int\limits_\Omega \left[\sigma_{ij}^c(t) - \widehat{\sigma}_{ij}^\theta(x,t)\right] \dot{\varepsilon}_{ij}^c(t) d\Omega \tag{17}$$

In addition to the global constraint, the optimization problem is subjected to the following constraints:

$$\boldsymbol{u} = \text{constant} \quad on\ S_u \quad (BoundaryConditions) \tag{18}$$

$$\boldsymbol{\varepsilon}_{ij} = \boldsymbol{BU} \qquad (Compatibility\ Conditions) \tag{19}$$

$$\boldsymbol{\varepsilon}_{ij}^c = \boldsymbol{N\lambda} \quad (Flow\ Law) \tag{20}$$

where $\boldsymbol{\lambda} > \mathbf{0}$.

A major set of nodal constraints for the minimization problem, is defined as

$$\theta_i = U_i^{\prime(-)} + U_i^{\prime(+)} \tag{21}$$

representing unrestricted curvatures(rotations) at each node. This allows the separation of bending at the nodes from membrane behaviour within the element when the shell is discretized.

4. Processing System - F.E. Code

Among several works found in the literature related to F.E. development environment, those proposed by T. Zimmermann and Bomme (1992); T. Zimmermann (1993); Devloo (1997); A. H. van den Boorgaard and Huétink (1998) are worth mentioning. The computer system for this work was conceived based upon the F.E. development environment proposed by T. Zimmermann and Bomme (1992); T. Zimmermann (1993). This system is part of a platform to implement limit analysis techniques for axisymmetrical pressure vessels using the FEM. Various classes proposed by T. Zimmermann and Bomme (1992); T. Zimmermann (1993) have been used such as those which implement matrices, lists, vectors for integers and float numbers, dictionaries, files reading and writing and so on. Further, the F.E. structure for the static analysis of reticular frames types of structures has also been used, since it is analogous to the F.E. discretization of axisymmetrical shells. The goal of this analysis system is to determine the ultimate load of axisymmetrical pressure vessels composed by the combination of primitive entities denominated cylindrical, conical, spherical and toroidal.

4.1. Processing System Requisites

The processing system (compute code) presented here is an intermediate module of a development platform for limit analysis of pressure vessels using a F.E. discretization. This module perform the F.E. limit analysis while the system as whole encompasses two other independent modules defined as the pre and the post graphical processor described in Franco et al. (2003). The facilities defined for the analysis module consist of a simple and intuitive interface for the edition of the vessel parameters and an easy communication with the graphical modules through the importation and exportation of data files.

4.2. Classes of T. Zimmermann and Bomme (1992); T. Zimmermann (1993) used without changes

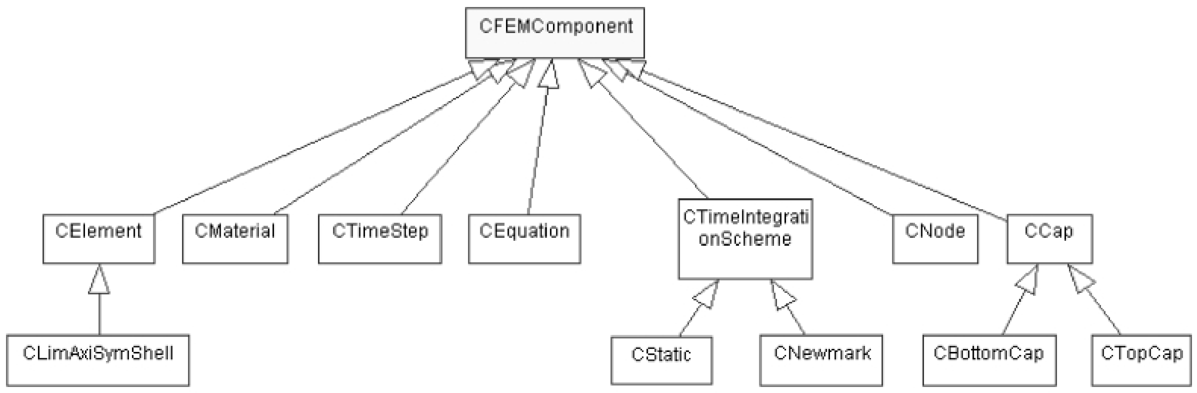


Figure 3. Class Diagram - CFEMComponent

CDictionary : chained list for storaging and reading of pairs of the type name/value. The task of this class is to storage and return the pair (name/value) when requested.

CFEMComponent : superclass of the F.E. components model (node, element, . . .), Figure 3, that groups attributes and methods for these components, such as:

- to implement random access to the data file;
- access to the class CDomain (to be described later).

CFileReader : allows randomic access to the data file.

CFloatArray : implements a float number vector. The responsibility of its methods are:

- to storage and to return a coefficient;
- to perform standard vector algebraic operations;
- to expand the vector size to accommodate more coefficients.

CIntArray : analogous to the previous class for integers.

CList : objects vector for child classes of CFEMComponent. Its tasks are:

- to sorage, to return and to erase a component;
- to expand its size to accommodate more coefficients.

CMaterial : responsible to inform material properties.

CMatrix : base class for different type of matrices, which implements basic matrices operations.

CPair : association key/value.

4.3. Classes of T. Zimmermann and Bomme (1992); T. Zimmermann (1993) used with changes

CFloatMatrix : implements a rectangular matrix of float numbers. This class is derived from CMatrix and has the following tasks:

- to storage and to inform certain coefficient;
- to perform standard operations such as multiplication, invertion, transposing for the equations (14) and (15).

CGaussPoint : groups attributes (coordinate and weight) of Gauss points.

CDomain : the attributes of this class are: list of elements, nodes, materials, equation of general restriction (2) and the linear programming system (17). The tasks of this class are:

- receive the user command (solveYourself method) to initiate the operation;
- create the objects of accessing the data file.

In fact, this class only sends messages to objects (elements, nodes, linear programming system), passing the responsibility for execution of tasks. The solveYourself method triggers three other methods:

- assembleLinProgSystemAt - responsible for the optimization problem assembly (CLinearSystem class). This method requests the objects assemblage from the Nodal Restriction Equation (21). It also assembles objects from the Constraint Equation (in this case the Bundary Condition Equations (18)), from the General Constraint Equation (2) and from the Cost Function (17), all classes, but the last one, derived from CEquation (described later);
- solveYourself from the class CLinearSystem (described later) - this method is responsible for the solution of the optimization problem through Linear Programming using SIMPLEX;
- terminate - this method perform the post-processing and ends the analysis.

CLinearSystem : this class is related to the optimization problem originated from the limit analysis problem. It deals with the storage of constraint equations system into objects of the type FloatArray. The others attributes of FloatArray type related to this class are the Cost Function coefficients (17), a set of plastic multipliers and the optimized value of the Cost Function, i.e., the upper bound of the limit load k^s. The main method of this class is solveYourself described previously with the class CDomain.

CNOde : a node is an attribute of one or more elements. It is the child class of the class CFEMComponent with the following attributes:

- a vector with nodal coordinates;
- the distance to the axis of symmetry;
- the thickness;
- the number of plastic multipliers and of the variables associated to the rotation at the plastic hinges (21);
- the vectors to storage the optimized plastic multipliers and rotation rate at the plastic hinges;
- the displacement vector in global and local coordinates.

This class CNode is responsible for

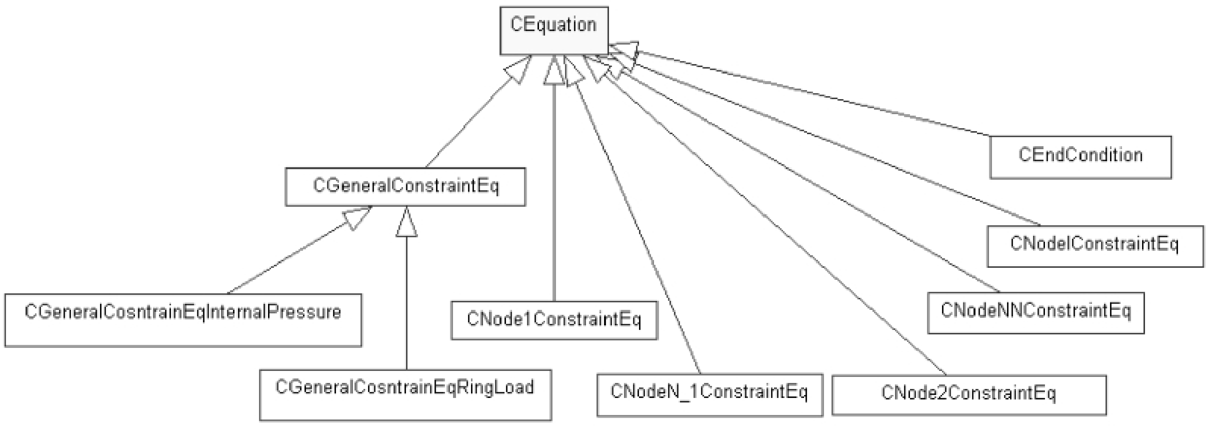
- extracting information from the data file to storage and manage them;
- calculating and informing the required values for the Global Constraint Equations.

CElement : this class groups the common aspects of a finite element for the limit analysis. It is the base class of the class CLimAxiSymShell. The main attribute of this class are the node numbers, the integration points, the list of node and of integration points and the material number. A special attribute is the matrix $\boldsymbol{L}$ from equation (14), an object of the type FloatMatrix, which relates the displacement rates to the plastic multipliers. It is responsible for the reading of information from the data file, storaging and returning of attributes. It also performs the calculation of the $\boldsymbol{L}$ matrix.

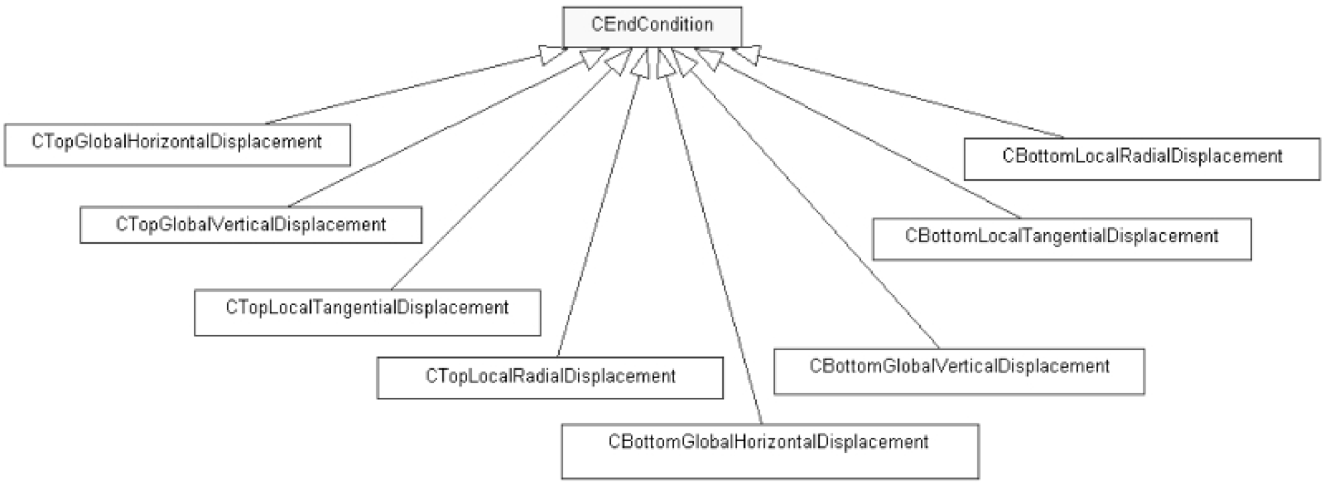
4.4. New Developed Classes

CLimAxiSymShell : implements a two nodes element for the limit anslysis of thin axisymmetrical shells. This class inherits methods and properties of the class CElement, Figure 3. Its data members are the length and the meridional curvature (associated with the direction ϕ) of the element. The tasks of this class are:

- to calculate the position and weight of gaussian points;
- to storage and return its attributes;
- to calculate the element volume;
- to calculate and to inform the following quantities associated to each gaussian points:
  - thickness;
  - the angle between the normal to the shell medium surface and the axis of symmetry (angle ϕ);
  - the distance from the gaussian points to the axis of symmetry;
  - the circunferencial radius of curvature;
  - the linear interpolation function of the element;
  - the element volume associated to each Gauss point;
  - the matrices $\boldsymbol{B}$ (8), $\boldsymbol{R}$ (11), $\boldsymbol{\Lambda}$ (6) e $\boldsymbol{K}$ (15);
- to obtain the necessary quantities to establish the Global Constraint Equations (16) and the Cost Function (17).



(a) CEquation


CEndCondition
CTopGlobalHorizontalDisplacement
CBottomLocalRadialDisplacement
CTopGlobalVerticalDisplacement
CBottomLocalTangentialDisplacement
CTopLocalTangentialDisplacement
CTopLocalRadialDisplacement
CBottomGlobalVerticalDisplacement
CBottomGlobalHorizontalDisplacement

(b) CEndCondition

Figure 4. Class Diagrams

CEquation : this class implements an abstract constraint equation. The set of different types of equations implemented through classes derived from CEquation, Figure 4(a), constitutes the system of constraint equations of the linear programming problem. The system data members are the number of equations and pointers to the left and right hand side of the equations. The task of the class is to request its attribute values from the optimization problem (CLinearSystem). Every time the optimization problem is notified about the definition of a new constraint equation, an appropriate space is allocated and the updated dimension of the system and the new equation number are returned. The methods of such class are virtual functions overrided in the derived classes, in which the coefficient of the system of restriction equations are calculated. The definition of these terms requires the determination of the plastic multipliers from the rate of displacements using Equation (14). Each one of the derived classes is restricted to one kind of the constraint equation and it is described in the following items. When the system of constraint equations is assembled (method CEquation :: assembleLhs ()), the pure virtual method CEquation::ComputeLhs() is called over the pointers to the objects of each equation. Here the late-binding mechanism is performed and one of the methods of the corresponding derived classes is called aiming to calculate the coefficients of the plastic multipliers.

CGeneralConstraintEq : this class is derived from CEquation to implement the General Constraint Equation (16). The main task of this class is to create an object of a derived subclass according to the type of loading defined for the analysis. Such subclass corresponds to the general restriction equations for the internal pressure or ring load case of loading. In the future new CGeneralConstraintEq derived classes will be create for other types of loading cases such as thermal loading and vertical loading. The method CGeneralConstraintEq::ofType(char* aClass) uses the information pointed to by the variable aClass to determine the type of general equation to be built, i.e, if it is resulting from the energy balance with internal pressure or with ring load. Depending on the kind of information sent to such method one of the constructors of the derived classes CGeneralConstraintEqInternalPressures or CGeneralConstraintEqRingLoad is called to build the object that represents the corresponding general constraint.

CGeneralConstraintEqInternalPressures : subclass of CGeneralConstraintEq, specific for the case of internal pressure loading.

CGeneralConstraintEqRingLoad : subclass of CGeneralConstraintEq, specific for the case of ring load.

Classes to implement nodal restriction equations : there are five classes, figure 4(a), used to represent the nodal restriction equations, all of them directly derived from CEquation. The class CNodeIConstraintEq implements a restriction equation associated to any generic node, except the initial and final nodes. The nodal constraint equations coefficients are obtained by substituting (14) into (7), which involves quantities corresponding to generic node and quantities related to the two previous and two subsequent nodes. For this reason there are four other special versions of the class, for the first two nodes ((CNode1ConstraintEq,CNode2ConstraintEq) and for the last two nodes (CNodeN_1ConstraintEq CNodeNNConstraintEq).

CEndCondition e derivatives : class derived from CEquation that was developed to implement an axisymmetrical restriction to the ends of a thin shell. It is responsible to create a derived class, Figura (4(b)), compatible with the end conditions. In terms of local coordinates, the types of restriction that can be imposed are:

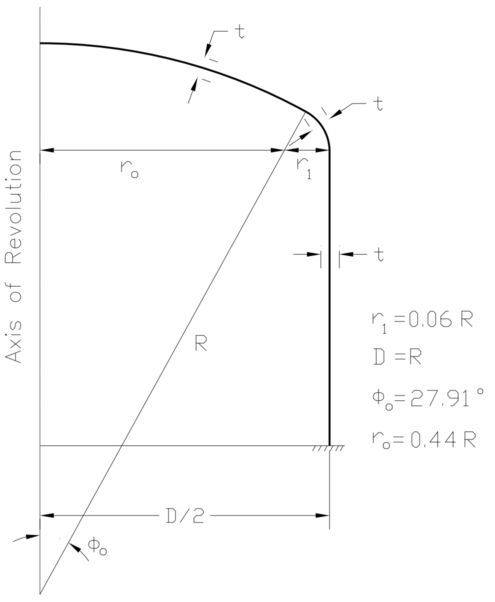
- restriction to the tangential displacement at the top nodal point (class CTopLocalTangentialDisplacement);
- restriction to the radial displacement at the top nodal point (class CTopLocalRadialDisplacement);
- restriction to the tangential displacement at the bottom nodal point (class CBottomLocalTangentialDisplacement);
- restriction to the radial displacement at the bottom nodal point (class CBottomLocalRadialDisplacement);

There is also the possibility to define end conditions in terms of global coordinates. In this case the implementation of the corresponding classes is direct since there is no need to apply any coordinate transformation. The restrictions are:

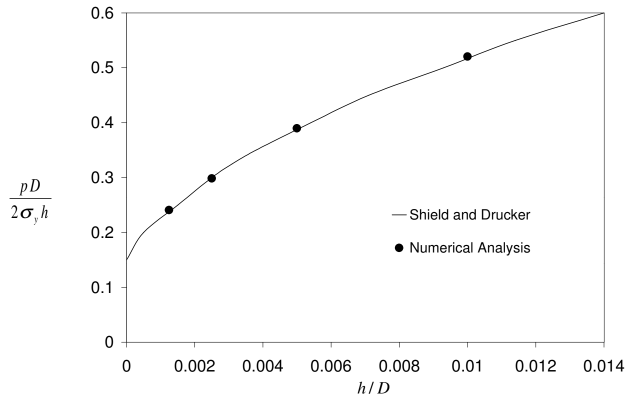
- restriction to the vertical displacement at the top nodal point (class CTopGlobalVerticalDisplacement);
- restriction to the horizontal displacement at the top nodal point (class CTopGlobalHorizontalDisplacement);
- restriction to the vertical displacement at the bottom nodal point (class CBottomGlobalVerticalDisplacement);
- restriction to the horizontal displacement at the bottom nodal point (class CBottomGlobalVErticalDisplacement);

CCap e derivadas : CCap is an abstract associated to the closed ends of thin shells. Its only attribute is the radius of a circular cap. There are two derived classes, CBottonCap and CTopCap corresponding to the cap at the ends of the vessel. These classes exist to help the class CGeneralConstraintEqInternalPressure in the calculation of the General Constrain Equation.

5. Numerical Example



(a) Geometry



(b) Limit analysis results for the ASME standard vessel.

Figure 5. ASME Torispherical Vessel

The select problem chosen to illustrated the limit analysis technique implemented under the OOP paradigm is a standard ASME torispherical vessel subjected to internal pressure shown in Figure 5(a). This problem has been analyzed by Shield and Drucker (1959). The results are compared with those obtained by Shield and Drucker in Figure (5(b)), where the curve relates the quantities $pD/2\sigma_y h$ with h/D. The numerical results obtained using the present technique are also marked in Figure (5(b)). These results were extracted from Table 1 for various thicknesses h and a constant diameter $D = 2.0 \times 10^0$ m. Finally, a collapse mechanism is shown in Figure 5(b) for a vessel with the following parameter $h/R = 0.005$. The final mesh shown in Figure 6 is the result of a mesh refinement based on an error estimator developed for this purpose, (Franco et al., 1997).

t/R	$P_L R/2t\sigma_y$
0.00125	0.2405
0.0025	0.2984
0.005	0.3895
0.01	0.5205

Table 1. Limit analysis results

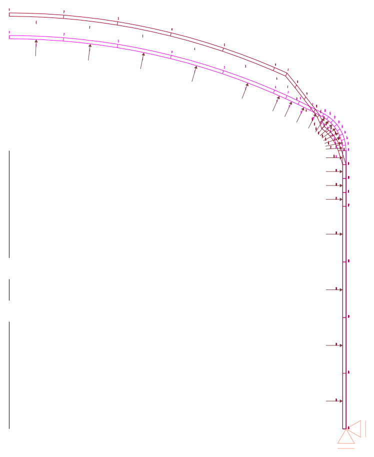


Figure 6. Geometry and collapse mechanism - post-processing

6. Conclusions

Object Oriented Programming, associated to a new CAD technology has been used to implement a Finite Element Technique for limit analysis of axisymmetrical pressure vessels. The new procedure allowed the automation of the process of limit analysis of shells integrating graphical pre-processing (modeling), processing (analysis) and visual post-processing of results. This new programming paradigm opens challenging perspectives to explore the proposed Finite Element Technique of limit and shakedown analysis. The obtained results show a great consistency with other analytical and numerical solutions.

7. Acknowledgments

The authors gratefully acknowledge the support of the Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) at Brazil and the Fundação de Amparo à Pesquisa do Estado de Minas Gerais (FAPEMIG).

8. References

A. H. van den Boorgaard, N. v. V., Huétink, J., 1998. Object oriented design of a thermo-mechanical fem code. CD-ROM Computational Mechanics- New Trends and Applications, 1998.

Devloo, P. R. B., 1997. Pz: An object oriented environment for scientific programming. Computer methods in applied mechanics and engineering 150, 133–153.

Drucker, D. C., 1953. Limit analysis of cylindrical shells under axially-symmetric loading. Proc. $1^s t$ Midwest Conf. Solids Mech I11, 158–163.

Drucker, D. C., Shield, R. T., 1958. Limit analysis of symmetrically loaded thin shells of revolution. J. Appl. Mech., Trans. ASME 25, 61–68.

Franco, J. R. Q., Barros, F. B., Malard, F. P., Balabram, A., 2003. Advances in finite element 3d modeling and meshing of axysmmetrical pressure vessels for the computation of limit loads. to appear in International Journal of Computational Engineering Science .

Franco, J. R. Q., Oden, J., Ponter, A. R. S. ., Barros, F., 1997. A posteriori error estimator and adaptive procedures for computation of shakedown and limit loads on pressure vessels. Computer methods in applied mechanics and engineering 150, 155–171.

Franco, J. R. Q., Ponter, A. R. S., 1997a. A general technique for the finite element shakedown and limit analysis of axisymmetrical shells - part 1 - theory and fundamental relations. International Journal for Numerical Methods in Engineering 40, 3495–3514.

Franco, J. R. Q., Ponter, A. R. S., 1997b. A general technique for the finite element shakedown and limit analysis of axisymmetrical shells - part 2 - numerical algorithm. International Journal for Numerical Methods in Engineering 40, 3515–3536.

Franco, J. R. Q., Ponter, A. R. S., Oden, J. T., 1995. Métodos adaptativos de elementos finitos para a computação de problemas de cargas limite e de shakedown de cascas axi-simétricas. Revista Internacional de Métodos Computacionais en Ingeniería 11 (4), 683–693.

Oden, J. T., Brauchli, H. J., 1971. Calculation of consistent stress distribution in finite element approximations. International Journal for Numerical Methods in Engineering 3, 317–325.

Oden, J. T., Reddy, J. N., 1973. Note on the approximate method for computing consistent conjugate stresses in elastic finite elements. International Journal for Numerical Methods in Engineering 6, 55–61.

Onat, E., 1955. The plastic collapse of cylindrical shells under axially symmetrical loading. Quartely of Applied Mathematics 33, 63–72.

Shield, R. T., Drucker, D. C., 1959. Limit strength of thin walled pressure vessels with an asme standard torispherical head. Congress. of Applied Mechanics , 665–672.

T. Zimmermann, Y. D.-P., 1993. Object-oriented finite element programming: Iii. an efficient implementation in c++. Computer methods in applied mechanics and engineering 108, 165–183.

T. Zimmermann, Y. D.-P., Bomme, P., 1992. Object-oriented finite element programming: I. governing principles. Computer methods in applied mechanics and engineering 98, 291–303.