International Journal of Civil and Structural Engineering Research ISSN 2348-7607 (Online)
Vol. 5, Issue 1, pp: (73-83), Month: April - September 2017, Available at: www.researchpublish.com

DEVELOPMENT OF EIGHT NODES DISCRETEKIRCHOFF QUADRILATERAL (DKQ8) ELEMENTS IN JAVA

I.E Umeonyiagu[1], N.P Ogbonna[2]

[1]Department of Civil Engineering, ChukwuemekaOdumegwuOjukwu University, Uli, Nigeria
[2]Department of Civil Engineering, ChukwuemekaOdumegwuOjukwu University, Uli, Nigeria

Abstract: **This study Developed Kirchorff Quadrilateral Elements using an object oriented approach and the Java programming language. A review of the work of Kansara (2004) and Nikishkov (2010) was conducted to develop a finite element program using Java programming language which addresses most of deficiencies inherent in the work of Kansara (2004). Input to the program is done through a text file and the loads that can be applied to the structure include concentrated loads, and uniformly distributed surface loads. Various load combinations can also be used. The program computes displacements and stresses at each node of the finite element model. Several test examples were analyzed using the program and results were compared with those obtained from Kansara(2004), the commercial finite element analysis program SAP 2000 and LISA respectively. Results were compared at the points of maximum displacements and stresses. The average stress was taken in to consideration to calculate stresses at specific point. The results obtained from the analysis of the example problems were found to be very accurate when compared to those obtained from the Kansara (2004), SAP 2000 and LISA. The difference in displacements computed by the two programs was less than 1, which is very negligible. The difference in stresses was also quite close. However, stresses in a few cases differed by 5 to 9 %. The difference in displacements was found to be less than 5 % for the eight node quadrilateral plate bending (DKQ8) element.**

Keywords: **Finite element model, Java programming language, eight node quadrilateral plate bending element, Object oriented approach, descrete Kirchoff quadrilateral elements.**

LIST OF SYMBOLS

U_e^b is element strain energy due to bending, L is Length, H *is* Depth, t *is* Thickness , E is Modulus of elasticity n is Poisson's ratio,σ_x, σ_y, σ_x, τ_z, τ_{yz}, τ_{zx} are the state of stress at any point $\epsilon_x, \epsilon_y, \epsilon_z, \gamma_{xy}, \gamma_{yz}, \gamma_{zx}$ are the components of the strain α *is* Coefficient of thermal expansion T *is* Change in temperature, ψ is the non-zero residual due to the approximate representation of a function inside a finite element.

1. INTRODUCTION

Since the evolution of the term finite element by Clough in 1951, there have been significant developments in finite element method. A large number of different finite elements have been developed (Kansara, 2004).McNeal (1978) developed a four node quadrilateral shell element using isoparametric shape functions. This element gives very good results for plate bending. Robinson and Haggenmacher (1979) developed the quadrilateral plate bending element, LORA based on stress parameters rather than displacement fields.

Batoz and Tahar (1982) reviewed the earlier attempts to develop plate bending elements and concluded that these elements were useful for thick plates, but when applied to the thin plates they do not give very good results. Batoz and Tahar (1982) developed a four node quadrilateral element based on the Discrete Kirchoff theory.Forte et al (1990) in one of the first publications on the object oriented approach to the finite element development, presented essential finite

element classes such as elements, nodes, displacement and force boundary conditions, vectors and matrices. Several authors described detailed finite element architecture using the object oriented approach. Nikiskov (2010) developed procedures for programming finite element in Java. He was able to demonstrate on how to solve finite element problems for solid mechanics as well as Heat transfer problems. Kansara (2004) developed membrane, plate and flat shell element in Java Programming language. He created a finite element analysis program using Java Programming Language to check the accuracy of the developed elements. He advocated for a future research work that can handle most of the deficiencies inherent in his work. These were the inability to developing flat shell elements in which the membrane elements have rotational (drilling) degrees of freedom, the use of a band storage scheme for storing the structure stiffness matrix and band solvers in the program to solve large finite element analysis problems, absence of a graphical user interface in the program and, extending the application of the program to include other elements such as truss, and frame elements.

This work improved on the work of Kansara (2004) using the procedures prescribed by Nikishkov (2010). Plate elements developed are the discrete Kirchoff quadrilateral (DKQ8) elements. A finite element analysis program was also developed with Java programming language to check the accuracy of the developed elements. Several test structures will be analyzed using the Java program and the results compared with those from other standard test validation examples.

2. LITERATURE REVIEW

2.1 Quadrilateral Plate Bending Element Based on Discrete Kirchoff Theory:

The quadrilateral thin plate bending element is efficient and useful for representing the bending part of flat shell elements (Kansara, 2004). The quadrilateral plate bending element can also be used for the analysis of plate structures such as slabs. Batoz and Tahar (1982) developed the Discrete Kirchoff Quadrilateral (DKQ) plate bending element by considering the Kirchoff assumptions for thin plates. Considering the element is in the xy plane (see Figure 2.1), the degrees of freedom at each node of the element can be described as the transverse displacement *w* in the direction normal to the xy plane, and the inplane rotations θ_x *and* $\theta_y$in the x and y directions respectively.

$$\left.\begin{aligned} w &= w(x,y) \\ \theta_x &= w_{,y} \\ \theta_y &= w_{,x} \end{aligned}\right\} \qquad (2.1)$$

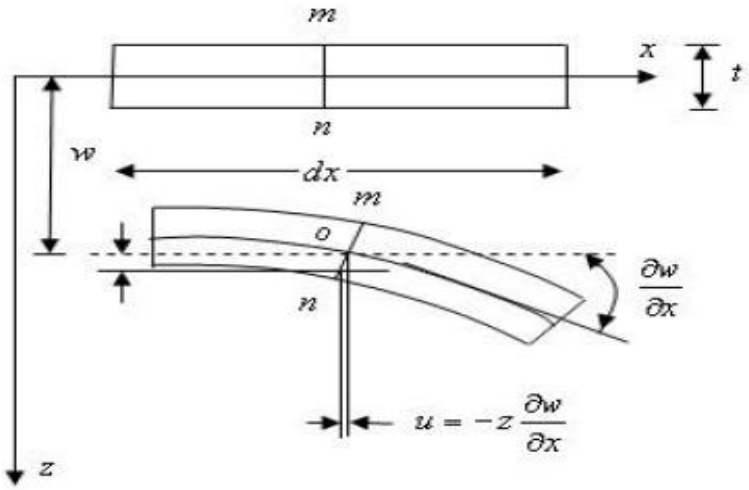

Figure 2.1 Bending of Plate.

The development of the DKQ element by Batoz and Tahar (1982) is described in this section. The formulation of the DKQ element is based on the Kirchoff assumptions. According to these assumptions the shear strain energy is neglected.

The strain energy of the element is,

$$U = \sum_e U_e^b \qquad (2.2)$$

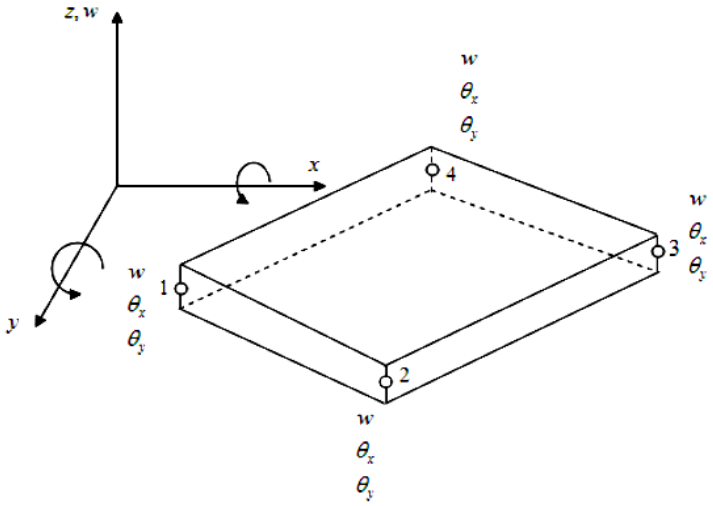

Figure 2.2 Quadrilateral Plate Bending Element

where, U_e^b is element strain energy due to bending and is given by,

$$U_e^b = \frac{1}{2}\int_{A^e} \langle\chi\rangle[D_b]\{\chi\}dxdy \tag{2.3}$$

For homogeneous isotropic material properties, the curvatures are given by,and A^e is the area of an element.

$$\{\chi\} = \begin{Bmatrix} \dfrac{\partial \beta_x}{\partial x} \\ \dfrac{\partial \beta_y}{\partial y} \\ \dfrac{\partial \beta_x}{\partial y} + \dfrac{\partial \beta_y}{\partial x} \end{Bmatrix} \tag{2.4}$$

Here,

β_x = rotation normal to the middle surface of the plate in the xz direction.

β_y = rotation normal to the middle surface of the plate in the yz direction.

The material matrix is,

$$D_b = \frac{Et^2}{12(1-v)^2}\begin{bmatrix} 1 & v & 0 \\ v & 1 & 0 \\ 0 & 0 & \dfrac{1-v}{2} \end{bmatrix} \tag{2.5}$$

$where, v = Poisson's\ ratio, t = thickness\ of\ the\ plate, and\ E = modulus\ of\ elasticity.$

Batoz and Tahar (1982) developed the relationship between the transverse displacement w and the rotations β_x and β_y as follows,

i. They defined β_x and β_y by incomplete cubic polynomials:

$$\left.\begin{aligned} \beta_x &= \sum_{i=1}^{8} N_i \beta_{xi} \\ \beta_y &= \sum_{i=1}^{8} N_i \beta_{yi} \end{aligned}\right\} \tag{2.6}$$

The shape functions for the eight node quadrilateral element are represented in the following form.

$$\left.\begin{aligned} N_2 &= \frac{1}{4}(1-\xi)(1-\eta)(1+\xi+\eta) \\ N_2 &= -\frac{1}{4}(1+\xi)(1-\eta)(1-\xi+\eta) \\ N_3 &= -\frac{1}{4}(1+\xi)(1+\eta)(1-\xi-\eta) \\ N_4 &= -\frac{1}{4}(1-\xi)(1+\eta)(1+\xi-\eta) \\ N_5 &= \frac{1}{2}(1+\xi^2)(1-\eta) \\ N_6 &= \frac{1}{2}(1+\xi)(1-\eta^2) \\ N_7 &= \frac{1}{2}(1+\xi^2)(1-\eta) \\ N_8 &= \frac{1}{2}(1-\xi^2)(1-\eta^2) \end{aligned}\right\} \quad (2.7)$$

ii. They applied Kirchoff assumptions at the corner nodes and the midpoint of the sides, at the corner nodes,

$$\begin{Bmatrix} \beta_{xi} + w_{,xi} \\ \beta_{yi} + w_{,yi} \end{Bmatrix} = \begin{Bmatrix} 0 \\ 0 \end{Bmatrix} \qquad i = 1,2,3,4 \qquad (2.8)$$

At the midpoints:

$$\beta_{sk} + w_{,sk} = 0 \qquad k = 5,6,7,8 \qquad (2.9)$$

iii. A cubic function was used to represent the transverse displacement *w*. Hence the derivative of the transverse displacement *w* with respect to *s* at the mid nodes of the element sides is a quadratic and is represented as,

$$w_{,sk} = -\frac{3}{2l_{ij}}(w_i - w_j) - \frac{1}{4}(w_{,si} + w_{,sj}) \qquad (2.10)$$

$k = 5,6,7,8 \quad for \quad ij = 12,23,34,41 \ and \ l_{ij} =$ length of the line for side connecting nodes ij

iv. The rotation normal to the sides at the mid nodes varies linearly.

$$\beta_{nk} = \frac{1}{2}(\beta_{ni} + \beta_{nj}) = -\frac{1}{2}(w_{,ni} + w_{,nj}) \qquad (2.11)$$

The nodal displacement vector for the Discrete Kirchoff Quadrilateral element is,

$$\langle U_n \rangle = \langle w_1 \quad \theta_{x1} \quad \theta_{y1} \quad \theta_2 \quad \theta_{x2} \quad \theta_{y2} \quad \theta_3 \quad \theta_{x3} \quad \theta_{y3} \quad \theta_4 \quad \theta_{x4} \quad \theta_{y4} \rangle \qquad (2.12)$$

Where,

$$\left.\begin{aligned} \theta_{xi} &= w_{,yi} \\ \theta_{yi} &= w_{,xi} \end{aligned}\right\} \qquad (2.13)$$

$for \ i = 1,2,3,4$

The quantities β_x and β_y are expressed in terms of the nodal displacements using the component vectors of the shape functions as,

$$\left.\begin{aligned} \beta_x &= \langle H^x \quad (\xi,\eta) \rangle \{U_n\} \\ \beta_y &= \langle H^y \quad (\xi,\eta) \rangle \{U_n\} \end{aligned}\right\} \qquad (2.14)$$

Where, $H^x(\xi,\eta)$ and $H^y(\xi,\eta)$ are the component vectors of the shape functions and are given in Equations (2.15) and (2.16),

$$\langle H^x \quad (\xi,\eta)\rangle = \begin{bmatrix} \frac{3}{2}(a_5N_5 - a_8N_8) \\ b_5N_5 + b_8N_8 \\ N_1 - c_5N_5 - c_8N_8 \\ \frac{3}{2}(a_6N_6 - a_5N_5) \\ b_6N_6 + b_5N_5 \\ N_2 - c_6N_6 - c_5N_5 \\ \frac{3}{2}(a_7N_7 - a_6N_6) \\ b_7N_7 + b_6N_6 \\ N_3 - c_7N_7 - c_6N_6 \\ \frac{3}{2}(a_8N_8 - a_7N_7) \\ b_8N_8 + b_7N_7 \\ N_4 - c_8N_8 - c_7N_7 \end{bmatrix} \tag{2.15}$$

$$\langle H^y \quad (\xi,\eta)\rangle = \begin{bmatrix} \frac{3}{2}(d_5N_5 - d_8N_8) \\ -N_1 - e_5N_5 - e_8N_8 \\ -b_5N_5 - b_8N_8 \\ \frac{3}{2}(d_6N_6 - d_5N_5) \\ -N_2 - e_6N_6 - e_5N_5 \\ -b_6N_6 - b_5N_5 \\ \frac{3}{2}(d_7N_7 - d_6N_6) \\ -N_3 - e_7N_7 - e_6N_6 \\ b_7N_7 - b_6N_6 \\ \frac{3}{2}(d_8N_8 - d_7N_7) \\ -N_4 - e_8N_8 - e_7N_7 \\ -b_8N_8 - b_7N_7 \end{bmatrix} \tag{2.16}$$

$$\left.\begin{aligned} a_k &= \frac{x_{ij}}{l_{ij}^2} \\ b_k &= \frac{3}{4}\frac{x_{ij}y_{ij}}{l_{ij}^2} \\ c_k &= \frac{\left(\frac{1}{4}x_{ij}^2 - \frac{1}{2}y_{ij}^2\right)}{l_{ij}^2} \\ d_k &= \frac{-y_{ij}^2}{l_{ij}^2} \\ e_k &= \frac{\left(\frac{1}{4}y_{ij}^2 - \frac{1}{2}x_{ij}^2\right)}{l_{ij}^2} \\ x_{ij} &= x_i - x_j \\ y_{ij} &= y_i - y_j \\ l_{ij}^2 &= \left(x_{ij}^2 + y_{ij}^2\right) \end{aligned}\right\} \tag{2.17}$$

$k = 5,6,7,8$ when $ij = 12,23,34,41$

3. MATERIALS AND METHOD

3.1 Creation of Robot Finite Element Analysis Program (Rfea):

The required input to Rfea is in the form of a text file and the results from the program saved in an output file in text format. A simple GUI was used during visualization of finite element models and results. Rfea performs three tasks of the finite element analysis: preprocessing (finite element model generation), processing (problem solution); and post processing (results calculation and visualization). Rfea finite element system is organized into eight class packages. The various packages and their individual classes are presented as follows:

3.1.1 Package fea:

This package shall contain the main classes which include FE, JFEM, JMGEN and JVIS.

3.1.2 Package model:

This package contain finite element model and loading classes

3.1.3 Package util:

Utility classes which include the FePrintWriter, FeScanner, GaussRule, UTIL

3.1.4 Package elem:

This package containsclasses such Element DKQ8., ShapeDKQ8.

3.1.5 Package material:

This package contains the constitutive relations for materials.

3.1.6 Package solver:

Classes for the Assembly and solution of global finite element equation systems:

3.1.7 Package gener:

The classes of this package will be used for generation of meshe

3.1.8 Package visual:

Contains classes visualization of models and results. Detailed description of some Rfea classes imported from Nikishkov (2010), were presented in the last sections of chapter two of this thesis work. The new Java Programming classes developed to modify Kansara (2004) work using Nikishkov's approach will be explained in the successive sections of this thesis work

3.2 Implementation of Discrete Kirchoff quadrilateral (DKQ8) Elements:

3.2.1 Formulation of Shape functions:

The shape functions for the eight node quadrilateral element are represented in equation (2.7). Now applying Kirchoff assumptions at the corner nodes and the midpoint of the sides, at the corner nodes, gives

$$\begin{Bmatrix} \beta_{xi} + w_{,xi} \\ \beta_{yi} + w_{,yi} \end{Bmatrix} = \begin{Bmatrix} 0 \\ 0 \end{Bmatrix} \qquad i = 1,2,3,4 \qquad (3.1)$$

At the midpoints:

$$\beta_{sk} + w_{,sk} = 0 \qquad k = 5,6,7,8 \qquad (3.2)$$

The quantities β_x and β_y are expressed in terms of the nodal displacements using the component vectors of the shape functions as,

$$\begin{matrix} \beta_x = \langle H^x \quad (\xi,\eta) \rangle \{U_n\} \\ \beta_y = \langle H^y \quad (\xi,\eta) \rangle \{U_n\} \end{matrix} \qquad (2.3)$$

$where, H^x(\xi,\eta)$and $H^y(\xi,\eta)$ $are\ the\ component\ vectors\ of\ the\ shape\ \ functions, with$

3.2.2 Calculation of strain-displacement matrix:

The strain-displacement matrix is obtained from the component vectors of the shape functions as,

$$[B] = \begin{bmatrix} \langle H^x{}_{,x} \rangle \\ \langle H^y{}_{,y} \rangle \\ \langle H^x{}_{,y} + H^y{}_{,x} \rangle \end{bmatrix} = \begin{bmatrix} j_{11}\langle H^x{}_{,\xi} \rangle + j_{12}\langle H^x{}_{,\eta} \rangle \\ j_{21}\langle H^y{}_{,\xi} \rangle + j_{22}\langle H^y{}_{,\eta} \rangle \\ j_{11}\langle H^y{}_{,\xi} \rangle + j_{12}\langle H^y{}_{,\eta} \rangle + j_{21}\langle H^x{}_{,\xi} \rangle + j_{22}\langle H^x{}_{,\eta} \rangle \end{bmatrix} \quad (3.5)$$

The Jacobian matrix is given by,

$$[B] = \begin{bmatrix} J_{11} & J_{12} \\ J_{21} & J_{22} \end{bmatrix} = \frac{1}{4}\begin{bmatrix} x_{21} + x_{34} + \eta(x_{12} + x_{34}) & y_{21} + y_{34} + \eta(y_{12} + y_{34}) \\ x_{32} + x_{41} + \xi(x_{12} + x_{34}) & y_{32} + y_{41} + \xi(y_{12} + y_{34}) \end{bmatrix} \quad (3.6)$$

In the equation for the strain-displacement matrix (Equation 3.5), the terms j_{11} , j_{12} , j_{21} , j_{22} are components of the inverse of the Jacobian matrix represented in Equation (3.6). These terms are obtained (Kansara, 2004) as follows,

$$\left.\begin{aligned} j_{11} &= \frac{1}{det[J]}J_{11} \\ j_{12} &= \frac{-1}{det[J]}J_{12} \\ j_{21} &= \frac{-1}{det[J]}J_{21} \\ j_{22} &= \frac{1}{det[J]}J_{22} \end{aligned}\right\} \quad (3.7)$$

The determinant of the Jacobian matrix is,

$$det[J] = \frac{1}{8}(y_{42}x_{31} - y_{31}x_{42}) + \frac{\xi}{8}(y_{34}x_{21} - y_{21}x_{34}) + \frac{\eta}{8}(y_{41}x_{32} - y_{32}x_{41}) \quad (3.8)$$

3.2.3 Calculating Element Stiffness Matrix:

Element stiffness matrices are calculated as follows:

$$[k^e] = \int_A [B]^T[D_b][B]dxdy \quad (3.9)$$

Equation (3.9) can be written in terms of natural coordinates as,

$$[K^e] = \int_{-1}^{+1}\int_{-1}^{+1} [B]^T[D_b][B]det[J]d\xi d\eta \quad (3.10)$$

Equation (3.10) can be computed using a standard 2 × 2 numerical integration scheme (Batoz et al, 1982),. This is found to be sufficient for the solution although theoretically 3 × 3 numerical integration scheme is required to integrate the quadratic functions. The weight functions and roots for two point Gauss quadrature are given in Table 3.1

Table 3.1 Weight functions and roots for 2 × 2 Gauss quadrature

Roots	**Weight Functions** *w*
±0.577350269189626	1.0

Equation (3.1) can be rewritten as,

$$[k]_{12\times12} = \sum_{j=1}^{2}\sum_{i=1}^{2} w_j w_i \left[B(\xi_i, n_j)\right]^T{}_{12\times3} [D_b]_{3\times3}\left[B(\xi_i, n_j)\right]_{3\times12} \left|J(\xi_i, n_j)\right| d\xi d\eta \quad (3.11)$$

3.2.4 Class Element DKQ8:

Class ElementDKQ8 is created to extend class Element and to implement methods for computing element matrices and vectors. The following arrays are declared similar to QUAD8:

an – shape functions; dnxy – derivatives of shape functions with respect to global coordinates *x*, *y* (first index is related to node number, second index to *x* and *y*); bmat – displacement differentiation matrix, emat – elasticity matrix; ept – vector of thermal strains.

Gauss Rule objects for integration of the element stiffness matrix, thermal vector, surface load, and equivalent stress vector are created.

Constructor ElementDKQ8(){} calls the constructor of parent class Element and passes to it the element name dkq, the number of element nodes (8) and the number of points for storing stresses and strains. Method stiffness Matrix DkQ() performs the computation of the element stiffness matrix according to Equation (3.9) and the stiffness matrix kmat is set to zero.

Integer variable ld represents a length of the strain or stress vector. For plane problems ld is equal to 3. Extraction of Material object mat is done from the hash table materials using the material name. Numerical integration of the element stiffness matrix is performed in a loop with a parameter ip denoting integration point number. Integration is performed inside a single loop.

Other important methods and their description are given in table 3.2

Table 3.2 Methods in the ElementDkQ8 class

Method	Description
setBmatrixDKQ8 ()	Performs computation of a displacement differentiation matrix bmat for specified local coordinates xi and et and returns the determinant of the Jacobian matrix.
DeriveDKQ8 ()	Computes the derivatives of shape functions dnxy with respect to global coordinates *x*, *y*
elasticityMatrixDKQ8 ()	Sets the elasticity matrix emat.
equivFaceLoadDKQ8 ()	Computing a nodal equivalent of surface load
rearrangeDKQ8 ()	Used to put load information in order, corresponding to local element numbering
shapeDerivFaceDKQ8 ()	Provides one-dimensional shape functions an and derivatives of shape functions xin with respect to local coordinate ξ changing along the considered element side.
equivStressVectorDKQ8 ()	Computation of a nodal force vector, which is equivalent to element stress field.
extrapolateToNodesDKQ8 ()	Stress extrapolation from integration points to nodes
getElemFacesDKQ8 ()	returns local numbers for four element sides specified in array faceInd
getStrainsAtIntPointDKQ8()	returns strains at the requested integration pointip.

3.2.5 Class ShapeDKQ8:

Class ShapeDKQ8 is placed in package elem. It is created to calculate the shape functions for DkQ8 elements. Element nodes are numbered in an anticlockwise direction starting from any corner node.

Method shapeDKQ8() computes element shape functions an for specified local coordinates xi (ξ) and et (η). Connectivity numbers ind are used as information on the existence of midside nodes.

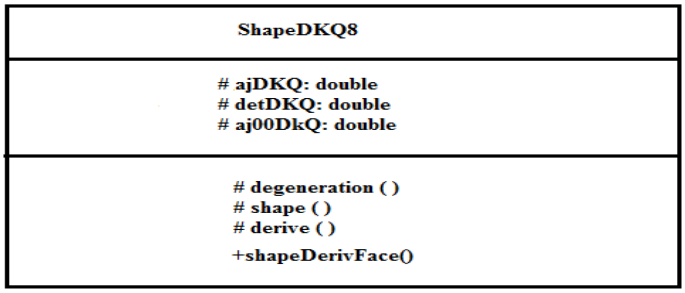

Figure 3.1: UML diagram for ShapeDKQ8 with its attributes and operations

Derivatives of shape functions with respect to global coordinates x, y are obtained by multiplication of the Jacobian matrix and derivatives with respect to local coordinates ξ , η

Method shapeDerivFaceDKQ8() calculates three shape functions an and their derivatives dndxi with respect to the local coordinate ξ.

4. NUMERICAL ANALYSIS

4.1 Verification of Eight Node Quadrilateral Plate Bending (DKQ8) Element:

The structure shown in figure 4.1 was previously modeled using four node quadrilateral plate bending (DKQ4) elements. In this case, we will use eight node quadrilateral plate bending (DKQ8) elements. The description of the example problems and comparison of results is discussed in the following sections.

4.1.1 Finite Element Analysis of a Square Plate using 64 Eight node quadrilateral plate bending (DKQ8) elements:

The finite element model was generated using 64 eight node quadrilateral plate bending (DKQ8) elements (See Figure 4.1).

Geometric Data: Length L = 1.440 m, Width = 1.44.0 m, Thickness t = 0.06 m.

Material Properties: Modulus of elasticity E = 3600 N/mm^2, Poisson's ratio η= 0.3

Boundary Conditions: All four edges of the plate are fixed.

Loading: A uniform surface load of 0.01 kN/m is applied to the plate.

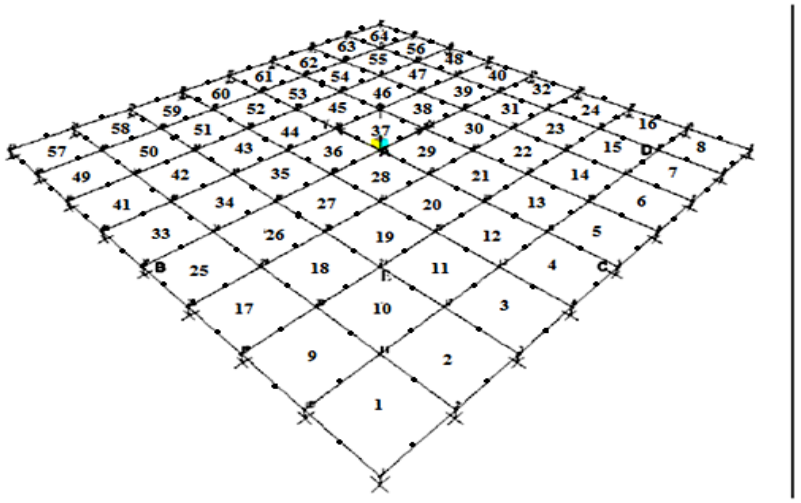

Figure 4.1: FE Model for a Square Plate using 64 right node quadrilateral plate bending (DKQ8) elements.

Comparison of Results:

Table 4.1 shows of displacements at points A and E and stresses at point B, C and D obtained from the program Rfea. The displacements at points A and E are approximately the same as those from Kansara (2004). The difference is 0.002 % t0 0.007 in the normal stresses in x and y direction at the fixed support (where the stresses are maximum). The difference in shearing stress from the program and that from the Kansara (2004), is about 5 %.

Table 4.1 Displacements and Stresses for a Square Plate using 64 right node quadrilateral plate bending (DKQ8) elements

Location		ProgrameRfea	Kansara (2004)	Difference (%)
POINT A (NODE 41)	**UZ**	-0.84010	-0.84010	0.107021826
POINT E (NODE 21)	**UZ**	-0.305787	-0.304787	0.327030373
POINT B (NODE 37)	**S11**	17.51186	17.51232	-0.002626804
POINT C (NODE 5)	**S22**	17.51114	17.51232	-0.006738572
POINT D (NODE 17)	**S12**	-2.69875	-2.82026	-4.502151033

4.4.2 Finite Element Analysis of a rectangular plate with seventy two - Eight node quadrilateral (DKQ8) elements

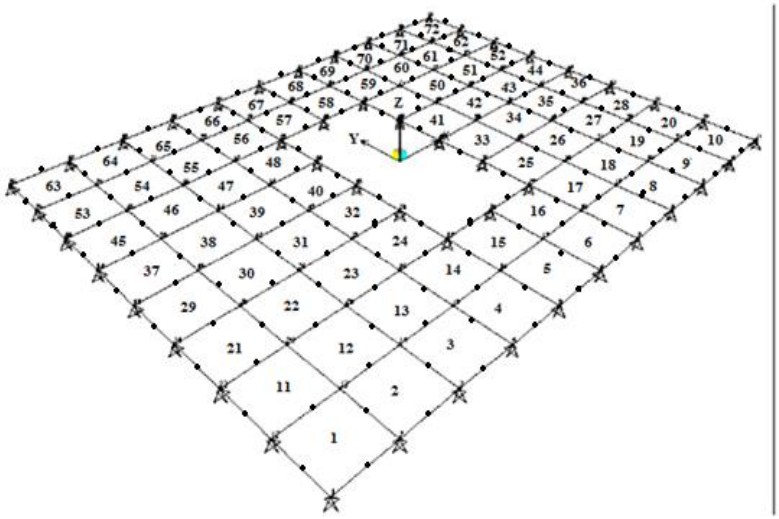

Figure 4.2: FE Model for Rectangular Plate with Hole.

The finite element discretization for a rectangular plate (See Figures 4.2 and 4.3), includes 72 eight node quadrilateral (DKQ8) elements.

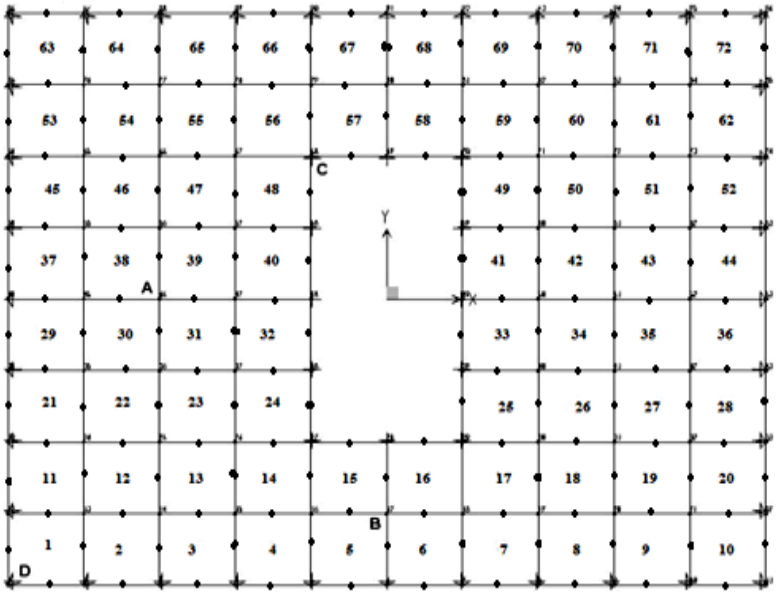

Figure 4.3: FE Model for a Rectangular Plate with Hole.

Geometric Data: Outside length L_1 = 0. 96.m, Outside width B_1= 1.44 m, Inside length L_2= 0.48 m, Inside width B_2= 0. 24 m Thickness t = 0.10m.

Material Properties: Modulus of elasticity E = 3600 N/mm^2, Poisson's ratio $\eta = 0.2$

Loading: A uniform surface load of 0.2 kN/m^2 is applied to the plate.

Boundary Conditions: All inside and outside edges of the plate are simply supported.

Comparison of Results:

The displacements and stresses obtained from the developed program Rfea and from SAP 2000 are tabulated in Table 4.2. The displacements are compared at points A and B while the stresses are compared at points C and D as shown in Figures 4.2 and 4.3. As can be seen from Table 4.2 the difference in displacements is very small. The stresses are also very close; however there is a difference of 5 % in shearing stress.

Table 4.2 Displacements and Stresses for a rectangular plate with seventy two - eight node quadrilateral (DKQ8) elements

Location		ProgrameRfea	Kansara (2004)	Difference (%)
POINT A (NODE 46)	**UZ**	-0.031026	-0.030033	-0.024830
POINT E (NODE 17)	**UZ**	-0.001538	-0.001278	-0.006480
POINT B (NODE 68)	**S11**	3.7208	3.72090	-0.0100
	S22	1.63774	1.637857	-0.003530
POINT C (NODE 1)	**S12**	1.104	1.149308	-1.28260

5. CONCLUSIONS

This research presented the development of Discrete Kirchoff Quadrilateral (DKQ8) elements using the object oriented programming concept in Java as an alternative to the traditional procedural programming approach. A finite element analysis program was developed to verify the accuracy of the results.

Some test example problems were analyzed using the developed program.The results from these analyses were compared with those obtained from Kansara (2004), the commercial finite element analysis program SAP 2000 and LISA respectively in order to verify the accuracy of the developed program. The results obtained from the analysis of the example problems were found to be very accurate when compared to those obtained from SAP 2000. The difference in displacements computed by the two programs was less than 1, which is very negligible. The difference in stresses was also quite close. However, stresses in a few cases differed by 5 to 9 %. The difference in displacements was found to be less than 5 % for the eight node quadrilateral plate bending (DKQ8) element.

REFERENCES

[1] Adam, F. M. and Mohamed, A. E. (2013), Finite Element Analysis of Shell structures, LAP LAMBERT Academic Publishing.

[2] B. Irons and S. Ahmad,(1980) Techniques of Finite Elements. Ellis Horwood, Chichester, UK.

[3] Djermane, M., Chelghoum, A., Amieur, B. and Labbaci, B. (2006), Linear and Nonlinear Thin Shell Analysis Using A Mixed Finite Element with Drilling Degrees of Freedom,International Journal of Applied Engineering Research, Volume 1 Number 2 pp. 217-236

[4] Fathelrahman. M. Adam, Abdelrahman. E. Mohamed, A. E. Hassaballa (2013) Degenerated Four Nodes Shell Element with Drilling Degree of Freedom,, IOSR Journal of Engineering (IOSRJEN).

[5] Gallagher R. H., (1975) Finite Element Analysis Fundamentals, Prentice-Hall.

[6] G.P. Nikishkov, (2010) Programming Finite Elements in Java™, Springer-Verlag London Limited.

[7] Kansara, K. (2004), Development of Membrane, Plate and Flat Shell Elements in Java, M.sc. Thesis, Faculty of the Virginia Polytechnic Institute & State University, 2004.

[8] McNeal R. H., (1978) A Simple Quadrilateral Shell Element, Computers and Structures, Vol. 8, pp. 175-183.