FINITE ELEMENT ANALYSIS OF GENERAL THREE DIMENSIONAL SPACE FRAME

Hairul Mubarak Bin Hassim

A dissertation submitted in
fulfillment of the requirement for the award of the
Degree of Master of Mechanical Engineering

FACULTY OF MECHANICAL AND MANUFACTURING ENGINEERING
UNIVERSITI TUN HUSSEIN ONN MALAYSIA

JAN 2014

Abstract

Finite Element Analysis to three dimensional space frames is the fundamental of Finite Element Analysis. Because of the shape of the space frame, the space frame is regarded as a line element in the Finite Element Analysis. Each of the elements will have two nodes which is located at its ends. Each of the nodes has six degree of freedom.. The first three degree of freedom are in translation in x, y and z direction and the next three degree of freedom are in rotational in θ_x, $\theta_y$and θ_z direction.

The programming of the Finite Element Analysis can be written either in Fortran, C, C++, Java and etc. Each of the programming languages has its own merit and demerit. The merit and demerit are in term of computing efficiency, computing speed and ease of writing a program in those languages.

The programming of the space frame analysis starts with the data input provided by the user. The required data input are the element connectivity's, the node coordinates, material properties, shape, force and constraint. From input data, a global stiffness matrix$[K]$, force vector$\{F\}$ and displacement vector$\{u\}$ are created. Using the Hooke's Law $F = Ku$, the displacement u of each nodes can be computed. Displacement u of the two ends nodes will results in elongation. Elongation of the space frame will cause the stress and strain in term of tension and compression. Since the stress can be computed and the cross sectional area is constant, the elemental force can be computed as the product of stress and cross sectional area.

Abstrak

Analisa terhadap struktur kerangka adalah asas kepada analisa Finite Element. Disebabkan oleh luas keratan rentas struktur kerangka, struktur kerangka dianggap sebagai satu garisan. Setiap elemen mempunyai dua nod yang terletak di penjuru elemen. Setiap nod mempunyai enam darjah kebebasan. Tiga adalah dari segi pergerakan nod di arah paksi x, y dan z. Tiga lagi adalah dari segi putaran di arah paksi θ_x, $\theta_y$and θ_z.

Pemprograman untuk analisa Finite Element boleh dihasilkan menggunakan bahasa komputer seperti Fortran, C, C++, Java dan lain lain lagi. Setiap bahasa komputer mempunyai kelebihan dan kekurangan masing masing. Kelebihan dan kekurangan adalah dari segi kepantasan pemprosesan dan kemudahan menulis dalam bahasa komputer itu.

Pemprograman dimulakan dengan data yang dibekalkan oleh pengguna. Data yang diperlukan adalah hubungan element dengan nod, koordinat nod, sifat bahan, bentuk, daya dan kekangan. Daripada data yang dibekalkan, matrik kekukuhan $[K]$, vektor daya $\{F\}$ dan vektor anjakan $\{u\}$ dihasilkan. Menggunakan hukum Hooke di mana $F = Ku$, anjakan setiap nod dapat di cari. Anjakan oleh dua nod menghasilkan pemanjangan atau pemendekkan. Pemanjangan dan pemendekkan akan menyebabkan ketegangan dan tekanan. Dengan menggunakan data tekanan dan luas keratan rentas, daya setiap element dapat di cari dengan mendarabkan tekanan dan luas keratan rentas.

Contents

	Declaration	ii
	Dedication	iii
	Acknowledgment	iv
	Abstract	v
	Abstrak	vi
	List of Figures	x
	List of Tables	xii
	List of Appendices	xiii
	List of Symbols	xiv
Chapter 1	**Introduction**	1
	1.1 Analysis Tool	3
	1.2 Finite Element History	4
	1.3 Finite Element Method by Direct Calculation	5
	1.4 Finite Element Analysis Software	6
	1.4.1 Commercial Finite Element Analysis Software	6
	1.4.2 Open Source Finite Element Analysis Software	7
	1.4.3 Integrated FEM-CAD Software	7
	1.5 Programming the Finite Element Analysis Software	8
	1.6 Problem Statement	10
	1.7 Project Objective	11
	1.8 Project Scope	12
Chapter 2	**Literature Review**	**13**

2.1 Space Frame in Finite Element Analysis 13
2.2 Finite Element Analysis 14
2.2.1 Procedural Oriented Finite Element Analysis Software 16
2.2.2 Object-Oriented Finite Element Programming 17
2.2.3 Design Patterns in Object-Orienting Finite Element Programming 20
2.3 Alternative to Finite Element Analysis 24
2.4 Finite Element Software Comparison 25
2.5 Finite Element Software Selection 26

Chapter 3 Methodology 28
3.1 Assumption 28
3.2 Programming the Finite Element Software 32
3.3 Characteristics of stiffness matrix 39
3.4 Verification 41
3.4.1 Lisa software 41

Chapter 4 Programming the Finite Element Analysis 43
4.1 Finite Element Pre-Processor 43
4.2 Finite Element Processor 47
4.2.1 Element Connectivity 47
4.2.2 Material 52
4.2.3 Shape of Element 52
4.2.4 Stiffness Matrix 52
4.2.4.1 Single matrix component 53
4.2.4.2 Multiple matrix component 56
4.2.5 Applied force 64
4.2.6 Constraint of boundary condition 66
4.2.7 Solve 66
4.2.7.1 Penalty Method 67
4.2.7.2 Gauss Elimination Method 69
4.2.7.3 Force 71
4.2.8 Result 71

Chapter 5 Conclusion 74
5.1 Verification 74
5.1.1 Verification of axial element analysis 75
5.1.2 Verification of Truss Two Dimensional Element 76
5.1.3 Verification of Truss Three Dimensional Element 78

5.1.4 Verification of space frame element 79
5.2 Discussion 82
5.3 Conclusion 87
5.4 Future Work 89

References **92**

Vitae **116**

List of Figures

Figure No	Title	Page
1.1	Methodology of product development	3
1.2	Classification of Analysis Methods	4
1.3	Example of Finite Element Programming Language	9
2.1	Finite Element Method Capability	14
2.2	Dimensional Space Frame	15
2.3	Assembly of finite elements	16
2.4	UML diagram of the element class (simplified representation)	18
2.5	Simplified UML diagram of the object oriented framework	20
3.1	Methodologies of developing analysis tool for three dimensional space frame	29
3.2	Analysis of Finite Element on surface area	31
3.3	Difference between Exact analysis and Finite Element Method	31
3.4	Element - node connectives	33
3.5	Constraint and Boundary Condition	38
3.6	Characteristics of stiffness matrix	40
3.7	Comparison of analysis procedure	42
4.1	Element Connectivity	48
4.2	Element - node connectivity matrix	48
4.3	Node coordinate matrix	48
4.4	Element - node coordinate matrix	49
4.5	Matrix $[T]$ for element 1,2 and 3	57
4.6	Matrix before and after transpose	59
4.7	Matrix $[k]$ for element 1 and 2	60
4.8	Multiplication of matrices	61
4.9	Multiplication of matrix	62
4.10	Assembly of global stiffness matrix	63

5.1	Example of axial element verification	75
5.2	Example of truss two dimensional element verification	77
5.3	Example of truss three dimensional element verification	78
5.4	Applying force on center of force	80
5.5	Verification of space frame	81
5.6	Structure analysis	83
5.7	Quad8 and Hex20	91

List of Tables

Table No	Title	Page
3.1	Length and Cosine Angle	33
5.2	Axial element comparison result	76
5.1	Axial element displacement comparison result	76
5.3	Truss two dimensional element displacement comparison	78
5.4	Truss three dimensional element displacement comparison	79
5.5	Output data for verification for circular shape	82
5.6	Output data for verification for annulus shape	84
5.7	Output data for verification for square shape	85
5.8	Result of verification of space frame	86
5.9	Structure analysis - node displacement	87
5.10	Structure analysis - node rotation	88

Chapter 1

Introduction

Three dimensional space frames are widely used mainly in construction and vehicle industries. Space frame are used as the main structure of a construction. The space frame are the structure of a bridge, the structure of a roof, the structure of a crane, the structure of a building and etc. In vehicle industries, the space frame are the main structure of the body of the vehicle. As for ease of explanation, the space frame represents the main frame of a body.

To avoid a construction part such as building or a bridge from collapse, other that natural disaster effect, a building and a bridge must withstand its own weight and the force that acted on them. In vehicle industries, a vehicle must withstand or acts on the force of collision. These two examples illustrate that an external force will be acted to the space frame of the body.

The space frame of a body may or may not have a grounding point. In a construction side, some of the ends of the structure are fixed. This is to prevent the structure from moving when force is applied to the structure. Unavoidable, not all of the structure can be fixed, and can moves at a certain distance. These movements have to be calculated so that the movements will not have a negative effect to whole structure. For a moving structure such a moving vehicle, there will be at least one fixed point that is the point where the force is applied to the moving vehicle. However, the principle of space frame may differs between moving and static bodies.

In industrial applications, the analysis improves the standard of engineering designs and the methodology of the design process. The analysis can substantially decrease the time taken for a product to be developed from a conceptual design to a finished product.

Without using the analysis, a company needs to construct a certain quantity of prototypes. These prototypes are mainly used for assembly related purposes and reliability functionality confirmation. Modification improvement or amendment are essential in developing a new product. This will cause for another sets of prototypes that need to be constructed for the reconfirmation. This cycling procedures will continue until the prototypes pass all the requirements. This method consumes a lot of design cost, energy and time. In addition, optimization is seldom achieved because it requires another series of testing that involves prototyping. Normally, a company is content to conclude the design development once the prototypes pass all the requirements. Figure 1.1a shows the methodology of current product development

With using the analysis, the design stage mostly is done by using Computer Aided Design (CAD) software. Design modification and design optimization can are carried out using the software. In case of product reliability issues, manual prototyping reliability testing is preceded by computational analysis software such as Finite Element Method (FEM) software. Only after the design and optimization already concluded, prototypes are constructed for final verification. Figure 1.1b illustrates the methodology of a product development using Finite Element Analysis.

In construction side, is unthinkable to have a concept of modification. A modification to a structure of a construction can make the structure losses its strength due the initial alignment of the structure is disturbed. In addition, the material of the modification may not perfectly bond to the initial structure of a construction.

Without using finite analysis, a construction has to have a high safety factor in order to ensure that the construction will not collapse. This safety factor increases the material usage. This material usage added to the cost of the construction in term of price material, delivery cost and manpower cost.

By using a finite element, the structure can be remodeled to have the optimum strength as required by the safety bodies. The safety factor may be added to some point where unnecessary material can be avoided.

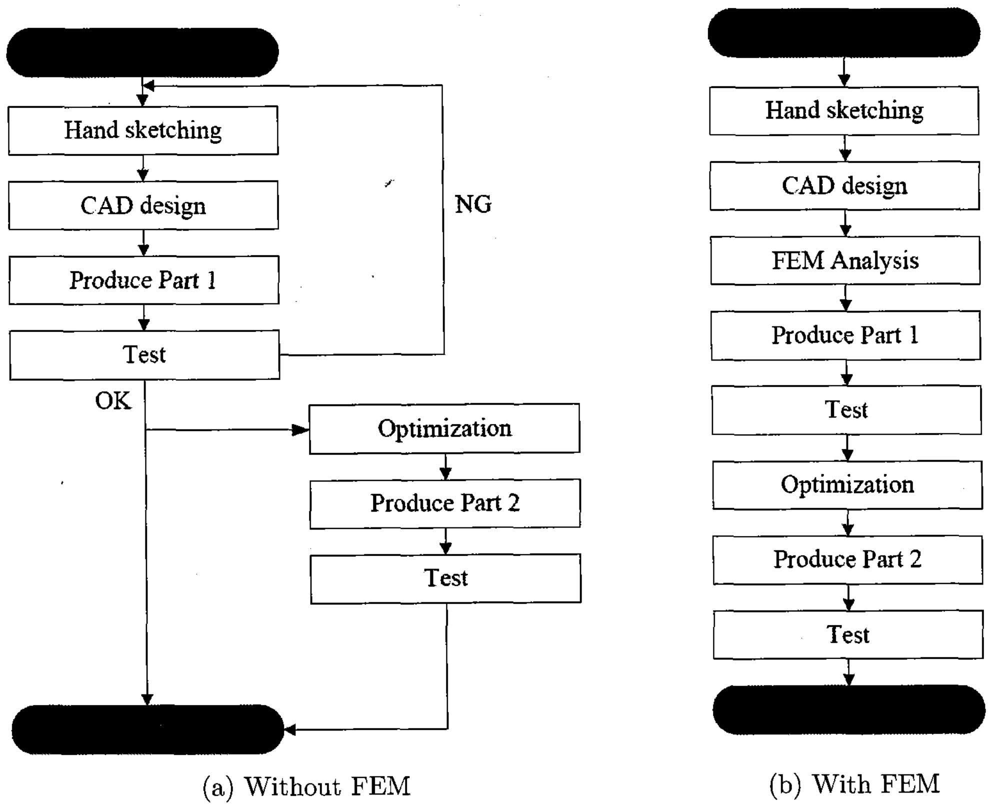


Figure 1.1: Methodology of product development

In summary, the benefits of finite element analysis to a three dimensional space frame are increase in accuracy, enhance in design and better insight into critical design parameters, virtual prototyping, fewer hardware prototypes, a faster and less expensive design cycle, increase in productivity, and increase in revenue.

1.1 Analysis Tool

Barkanov (2001) stated that the analysis method can be classified into two main groups. They are analytical and numerical method as shown in Figure 1.2.

The analytical method is further classified into two groups that are exact and approximate method. The examples of exact method are separation of variables and Laplace transformation method while the examples of approximation method are Rayleigh-Ritz method and Galerkin Method.

The numerical method is also classified into another two groups that are numerical solution and Finite Element Method. The numerical solution is then

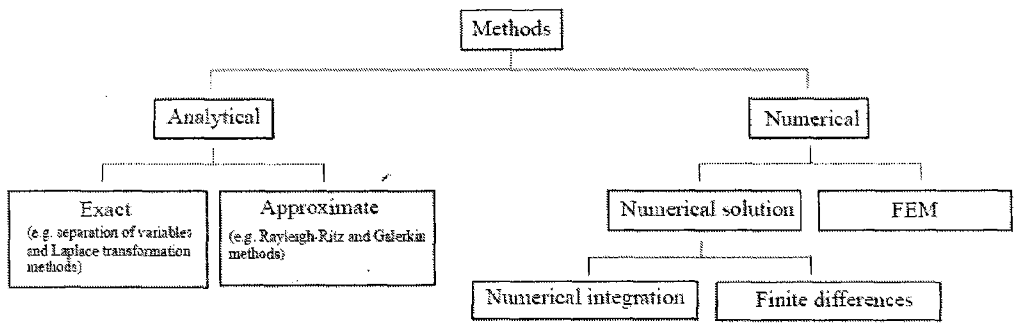


Figure 1.2: Classification of Analysis Methods (Barkanov, 2001)

divided into another two groups that is numerical integration and Finite Differences. From the available analysis methods, Finite Element Method is taken as an analysis tool for this report.

1.2 Finite Element History

The ideas that lead to the development of Finite Element Method were inspired by Euler and Langrange. By using Euler's and Langrange's findings as part of his research, Ritz developed an effective method to determine the approximate solution in the mechanics of deformable solids (Barkanov, 2001). His method also includes an approximation of energy functional of the known functions with unknown coefficients. By using minimization of functional in relation to each unknown, the system of equations from the unknown coefficients can be determined. One of the difficulties faced by Ritz was that the functions in his method should satisfy the boundary condition of the problem.

The boundary condition restriction were solved by Courant in 1943. In his research, he introduced the special linear functions defined over triangular regions and applied the method for the solution of torsion problem. As unknowns, the values of functions in the node points of triangular regions were chosen.

Clough (1960) introduced the term 'finite element' in 1960 (Barkanov, 2001). However, the Finite Element Method proposed by Clough was more or less the same as the Ritz method with the Courant modification. Clough's Finite Element Method become popular at that time due to the possibility to use computers for the big volume of computations required by Finite Element Method.

As researches of Finite Element Method increases, there were needs for textbooks of Finite Element Method to be published. Barkanov (2001) stated that the first book that was examined as a Finite Element Method textbook was published by Zienkiewicz & Cheung (1967).

1.3 Finite Element Method by Direct Calculation

Calculation method is the first Finite Element Method tool to analyze three dimensional space frames. Finite Element Method by calculation can be used to analyze structural analysis, heat transfer, fluid flow, mass transport, and electromagnetic potential to some extent. Due to calculation on Finite Element Method requires mathematically handling; the analysis is limited to non complex analysis.

In Finite Element Method, the whole body is discretized to equivalent system of smaller bodies or units that are called finite elements. As for calculation Finite Element Method, the number of finite elements is limited due to time processing and calculation difficulty limitation. Each of the finite elements needs to be calculated, therefore, larger number of finite element requires longer processing time and become more complex. It is possible to create any number or finite element per unit length. However, normally, one element is assigned per unit length. By using algebraic equation, the analysis data for each of the finite elements can be formulated. The calculation of each of the finite elements is combined to obtain the solution for the whole body. The time processing time can be expedited with the use of spreadsheet such as in Microsoft Excel or Open Office.

This analysis is limited to the analysis of deformation, stress, strain and force applied to each finite element. The analysis output data is discrete data of each elements and not a distribution data throughout the whole body. Therefore, the distribution analysis data of the whole body cannot be obtained. With this disadvantage, calculation Finite Element Method usage is limited to educational purposes to understand the Finite Element Method and not in industries for analysis purposes.

1.4 Finite Element Analysis Software

In industrial segment, Finite Element Analysis software are widely used. In today's computer technology, computer can computes large numbers of finite elements in a short time. This enables the size of the finite elements to become smaller. The acquired data analysis of each element is also a discrete data as the calculation method. However, because the size of each finite element is very small, the discrete data becomes a distribution data.

In structural design, Finite Element Analysis software allows detailed visualization where the structures bend or twist, and indicates the distribution of stresses and displacements. Finite Element Analysis software provides a wide range of simulation options for controlling the complexity of both modeling and analysis of a system. Similarly, the desired level of accuracy required and associated computational time requirements can be managed simultaneously to address most engineering applications. Finite Element Analysis software allow entire designs to be constructed, refined, and optimized before the design is manufactured. In a structural simulation, Finite Element Analysis software help tremendously in producing stiffness and strength visualizations and also in minimizing weight, materials, and costs.

The Finite Element Analysis software packages that are available in the market can be categorized into three categories. They are the Commercial Finite Element Analysis software, Open Source Finite Element Analysis software and Commercial Finite Element Method (FEM) - Computer Aided Design (CAD) software.

1.4.1 Commercial Finite Element Analysis Software

Commercial Finite Element Analysis software are developed by recognized software companies. The examples of commercial Finite Element Analysis software are Abaqus developed by Dassault Systèmes Simulia Corp., ANSYS developed by ANSYS Inc., ALGOR by Autodesk and NASTRAN by MacNeal-Schwendler Corporation. Other examples of commercial Finite Element Analysis software packages are STRAND7, SAP2000, LUSAS, JMAG and ADINA.

In addition to commercial Finite Element Analysis software, there are also general software that can perform finite element analysis. The example of this software is MatLab. Unlike commercial Finite Element Analysis software, MatLab does not give an instruction how to perform finite element analysis. A MatLab user needs to understand the commands in MatLab in order to perform the analysis.

For the commercial software, the developments of the Finite Element Analysis software are sustained by the developers. The users are guided on how to use the software. There is no or a little option for the user to modify the analysis function limited by the developers. By preserving the sustainability of their software, they can guarantee the quality assurance, analysis data verification and validation of their software.

1.4.2 Open Source Finite Element Analysis Software

Open source Finite Element Analysis software packages are usually developed by either Finite Element community users or by universities. As the software are classified as open source software, they are available to be downloaded for free. Examples of the open source software are deal.II developed by Texas M&A University, FreeFem++ developed by Université Pierre et Marie Curie and Laboratoire Jacques-Louis Lions, FEAP by Berkeley University and Calfem by Lund University. Other examples of open source software are GetFem++, OOFEM, ParaFEM and CulculiX.

Identical to MatLab, there are also open source general software that can perform Finite Element Analysis. Examples of the software are FreeMat and Calfem.

As these software are open source software, there are issue regarding the modularity and sustainability of the software. Software updates and additional programming codes can be contributed by the Finite Element community user or by the universities. The question are who and how to sustain the validity verification of the software.

1.4.3 Integrated FEM-CAD Software

For industry segment, Finite Element Method (FEM) - Computer Aided Design (CAD) software are the most sought software in the Finite Element Analysis software category. In this integration software, the Finite Element Method software is embedded to Three Dimensional (3D) CAD design software. There are correlations between 3D design software developer and Finite Element commercial developer. A product will initially be designed in a 3D design environment. Upon completed, it will be analyzed in a Finite Element environment. To increase the product performance or for optimization, the analyzed product design can be redesign in 3D design software and rerun in the finite element software. The examples of integrated software are Creo Parametic by PTC, Autodesk Inventor by Autodesk and Solidwork by Dassault Systèmes.

The Finite Element Analysis in FEM-CAD software is an application software. The user need to provide the necessary requirement to the software such as the 3D design model, mechanical properties and the force applied to the design. With these information, the Finite Element software will provide the analysis output in a graphical user interface (GUI) environment. Since this is application software, the software developer does not include any option to the user to modify the Finite Element Analysis method.

1.5 Programming the Finite Element Analysis Software

All of the commercial FEM-CAD software developers do not offer subroutines or programming codes to their users. The users can only use the limitation analysis that are provided by the software. For some commercial Finite Element Analysis software, they offer until certain level of subroutine, while others do not offer subroutine at all. The main reason is that the developers do not want the users to modify any part of the programming code. This is to ensure that the output analysis verification will be sustained. Any changes to the subroutines may cause the analysis result to differ from the actual result. This can cause the validity of the software to be questioned.

With the advance of technology, more complex and nonlinear situations are required to be analyzed. With the modification restriction imposed by the software developers, there is a need to develop new software to analysis these new conditions. Furthermore, commercial FEM-CAD software and commercialize Finite Element Analysis software create a community of user that are only able to make analysis without understanding how the analysis is carried out. They are required to provide all the necessary information and the software will present the analysis output to them. Therefore, developing a finite element program enables the researcher to comprehend the concept of the finite element and the answer

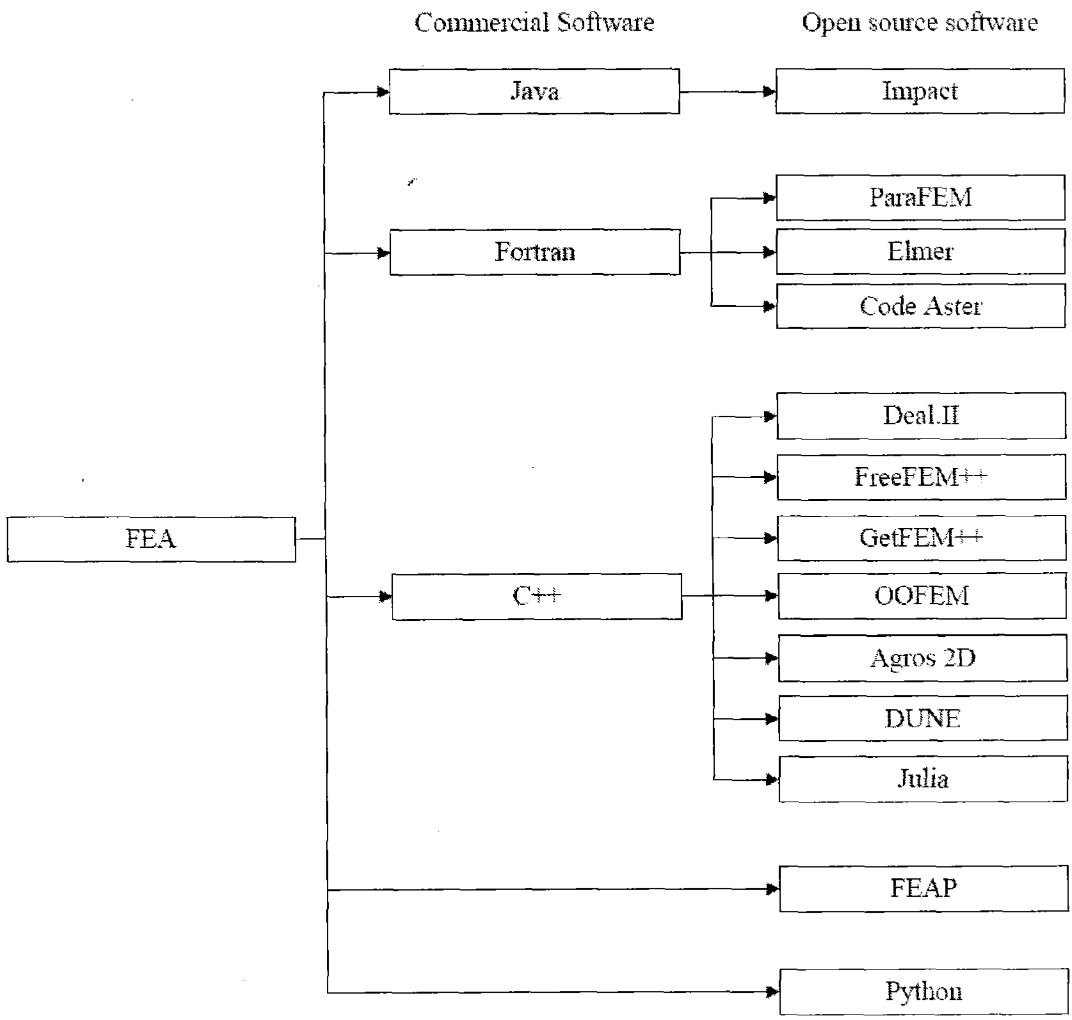


Figure 1.3: Example of Finite Element Programming Language

on why the necessary information need to be provided to the software.

There are many programming languages available to develop Finite Element Analysis software. They are grouped into two categories namely commercial programming language and open source programming language. Figure 1.3 indicates the available programming languages software for commercialize and open source software.

Commercial programming languages are developed by programming software companies. The examples of programming language software to develop Finite Element Analysis software are C++ developed by Bell Laboratories, Java developed by Sun Microsystems Inc. and Fortran developed by IBM. All of the programming software have their own programming codes. However, some of the programming codes are compatible to be used by other programming language. The example is to write a programming code using C syntax in Java environment. In choosing the right programming language, one has to know the software's advantages and disadvantages in terms of difficulty of programming, technical assistance availability, user interface and computer processing time of the analysis.

Since the price of the commercial programming language is relatively expensive, open source programming software offer free option to develop Finite Element Analysis software. Mostly, the open source software language are developed by universities, programming language communities and even software companies.

There are many reasons why open source programming software are developed. Commercial programming software such as Fortran, C++ and Java are general programming language. Thus, there are compromises that are required by the software such as the speed, features and ease of use. This leads to other developers to develop better programming language that better suit Finite Element Analysis as highlighted by Deal.II. In addition, universities develop the programming language for their researches and education purposes. Furthermore, easier programming languages with fewer capabilities are developed for ease of programming purposes.

The main reason why these software offers programming codes is that these software are developed using commercial programming software such as C++, Java and Fortran. The examples of the software are deal.II written in C++, Impact written in Java and ParaFEM written in Fortran. However, there are also open source software that are developed independently such as Python and FEAP.

1.6 Problem Statement

Commercial Finite Element Analysis software is written in a general purpose code. The idea is so that the developer can write analysis software that can be used to analyze a multitude of problems. By writing the software in this manner, the software can cover a larger possibility of finite element analysis and fulfill the needs of the industry. However, the bigger capability of software to perform multitude finite element analysis, the software becomes more complex to be used. Initiative person need to go thorough trainings and learning in order to understand most of the functions of the software in order to perform even an easy analysis. In addition to the complexity in using the software, the software is accompanied with large and complicated data structures. This requires a higher processing capability of computer to perform the analysis faster.

As reliable as commercial Finite Element Analysis software, they come with a price tag. The price of the software includes all the analysis that the software can perform. Even though the Finite Element Analysis software package can be purchased, it is seldom for mediocre user to fully use all the available analysis. It is more relevant to have software that are lower in analysis capabilities but are fully utilized by the user than to have software with lots of analysis capabilities but are not used by the user.

To sustain the validation of the result of Finite Element Analysis, most of the commercial software have fewer tendencies to be able to be altered. This is not a drawback to a user or an analyst. A user or an analyst require highly robust, well documented, fully verified codes with good technical support to solve their problems. They do not interested in knowing how the computations are carried out. They only interested on the result of the analysis. On the other hand, difficulty in modifying the programming codes is a major drawback for researchers and developers. This group of people are desire to have an access to a well-established reliable source code, which can be used as a foundation and building blocks to their development of existing problem or new problems that has never been solved.

1.7 Project Objective

The primary objective of this report is to produce a tool that can perform mechanical analysis to one dimensional to three dimensional space frame. Finite Element Method already been verified to be one of best tools to analyze mechanical properties. For this reason, the author proposes to develop an analysis tool using Finite Element Method as its backbone. The main objectives in constructing the analysis tool are:

1. To develop a Finite Element Analysis tool that performs computational static analysis of one to three dimensional space frame structure. This tool will use Finite Element Analysis platform specifically for frame structure. This analysis tool does not integrate with Computer Aided Design (CAD) software.

2. To develop a reliable and validate Finite Element Analysis software package.

1.8 Project Scope

Finite Element Analysis software covers a huge aspect of analysis. This report will only cover some portion of Finite Element Analysis capabilities. This is to be coherent with the objectives of this project so that a user is able to perform Finite Element Analysis up to static level of Finite Element Analysis of three dimensional space frame.

As with other available Finite Element software, the validation of the result of the analysis is one of the top priorities. To validate the produced software, collection of problems will be selected from one dimensional to three dimensional conditions. The result of each analysis will be compared to the analysis result from other commercial and open source Finite Element Analysis software. For validation, the result for all comparisons must be the same.

Final validation can only be performed for complete finished software, as the distribution data will only available for complete finished software. If there is different of analysis result, a software developer needs to thoroughly examine all the tedious programming codes in order to correct them. To avoid this tedious amendment, the programming coding will be made segment by segment. For each finished segment, it will be validated by calculation method in a spreadsheet.

The complete relationship between the stress σ, and strain ε, is given by the equation $\sigma = E\{\varepsilon - \varepsilon^t\}$ where E is the Young modulus while ε^t is the thermal strain. However, in this project, the temperature difference is not taken into consideration. The stress and strain relationship is taken as $\sigma = E\varepsilon$.

To some extend, due to the loading or force applied to the space frame element, the material properties will change from elastic properties to elastic plastic properties. For this program, only the elastic properties is taken into consideration.

Chapter 2

Literature Review

This chapter will provide the reviews of the concept of three dimensional space frames. It will also review the development of the Finite Element Analysis in term of numerical method and software improvement. In conclusion, explanation on why the programming language is chosen to write the Finite Element Analysis software will be given.

2.1 Space Frame in Finite Element Analysis

Space frame is a truss-like, lightweight rigid structure constructed from interlocking struts in a truss. A space frame is strong because of the inherent rigidity of the triangle, flexing loads (bending moments) are transmitted as tension and compression loads along the length of each strut. Numerical Finite Element Method is able to analyze a space frame that is straight in a unit length. Finite Element Analysis software is able to perform analysis of any shape. This is illustrated in Figure 2.1 where numerical method can only perform analysis for a straight measured subject.

One of the disadvantages of numerical finite element method is that it is better to be used to analyze object with constant area along its length. Numerical finite element method can be used to analyze space frame since the area along its length is constant. Finite Element Analysis software does not have the constant

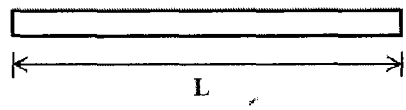
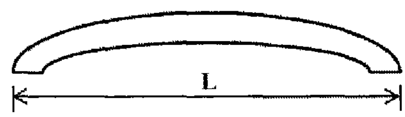
No.	Shape	Numerical Method	FEA software
1	Straight in a unit length 	**CAN**	**CAN**
2	Not straight in a unit length 	**CANNOT**	**CAN**

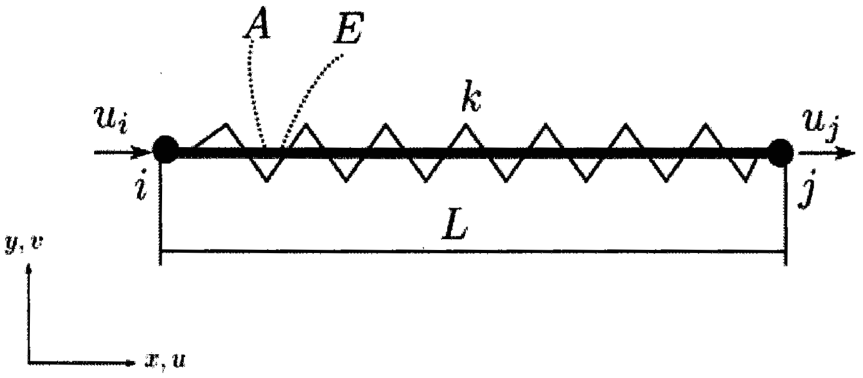
Figure 2.1: Finite Element Method Capability

area limitation. It can analyzes any shape of the object. Space frame can be treated as one dimensional, two dimensional and three dimensional line elements. For simplicity of explanation, in one dimensional, the axis of the analysis object is in x-direction and the force applied is also in x-direction. There is no force and analysis object movement in y and z direction. In two dimensional, force is applied at an angle to the axis causing a force and analysis object movement in x and y direction. In three dimensional, force is applied at an angle to the axis causes a force in x, y and z direction. The analysis object can also move in x, y and z direction. Clearly stated that three dimensional space frame element has three rotational and three transitional degree of freedom. Figure 2.2 illustrates the movement difference of one, two and three dimensional space frame.

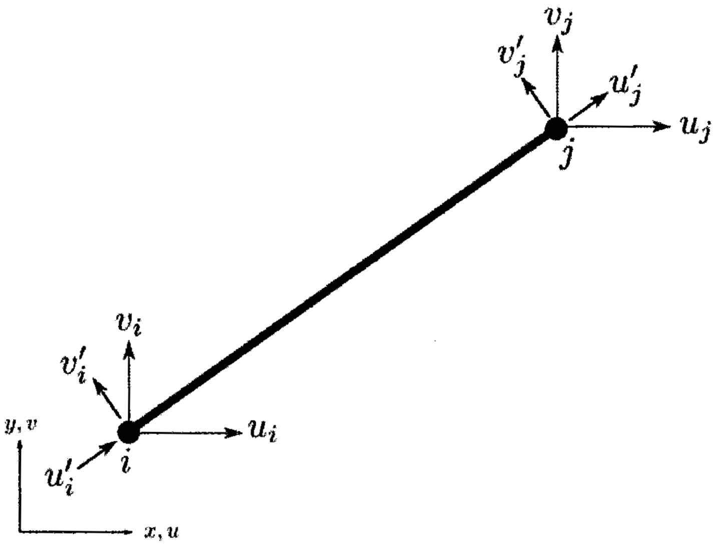
2.2 Finite Element Analysis

Finite Element Analysis (FEA) is also known as Finite Element Method (FEM). Finite Element Analysis is a computational technique for solving problems that are described by partial differential equations or can be formulated as a functional of minimization. A main interest of Finite Element Analysis is represented as an assembly of finite elements. A continuous physical problem is transformed into a discretized finite element problem with unknown nodal values. Figure 2.3 illustrates single and global finite elements. 1,2 and 3 are the finite elements while 1 to 7 are the nodes. An element can share the same nodes with other elements. Two features of the FEM that are worth mentioning are (Nikishkov, 2010):

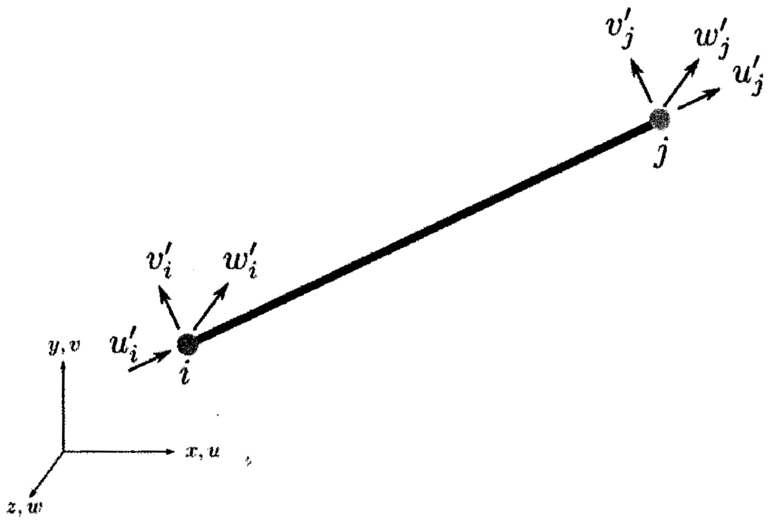
1. Piece-wise approximation of physical fields on finite elements provides good precision even with simple approximating (interpolation) functions. By increasing the number of elements and nodes precision of results can be



(a) One Dimensional Space Frame.

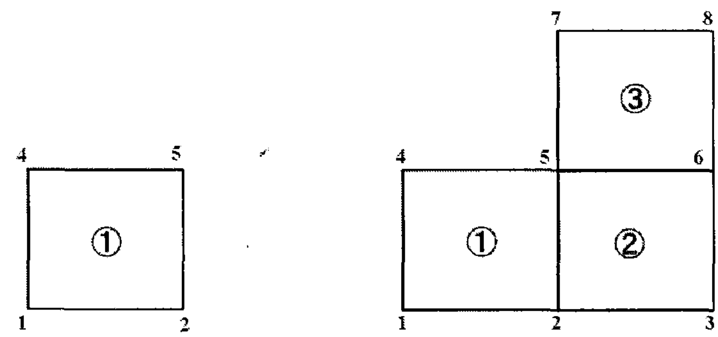


(b) Two Dimensional Space Frame



(c) Three Dimensional Space Frame

Figure 2.2: Dimensional Space Frame. (Siswanto & Darmawan, 2012)



(a) Single Element (b) Global Elements

Figure 2.3: Assembly of finite elements

achieved.

2. Locality of approximation leads to sparse equation systems for a discretized problem. This helps to solve problems with a very large number of nodal unknowns using reasonable memory and computing time.

Finite Element Analysis Software has evolved through years from its initial creation. Programming language software limitation has forced Finite Element Analysis software to be initially written in procedural oriented approach. With the advancement of technology, Finite Element Analysis is improved with the use of object oriented programming. It is further enhanced with the development of design pattern programming.

As the Finite Element Method reaches its maturity, there is an urge to develop better analysis software to compliment the weakness of Finite Element Analysis. Isogeometric Analysis is said to be the next best thing after Finite Element Analysis.

2.2.1 Procedural Oriented Finite Element Analysis Software

Initially Finite Element Analysis software was written using Fortran and C programming software Pantale *et al.* (2004). It was written in a procedural oriented approach in which the finite element algorithm is broken down into procedures that manipulate the data. For software with large programming codes, the procedures are used as modules and wrapped in libraries. With the existence of libraries, it is possible to link different libraries to produce complete programming software.

Although it was a success to write the software using procedural oriented approach, there are many drawbacks to this approach. Most of the programming codes are associated to each others. Changing any code in between the program will affect to whether the complete program can be functional or not. One has to modify almost the entire program in order to make any modification code. In addition, interdependency in the program architecture are always hidden and difficulty to determined. This requires the programmer to fully understand the programming codes of the program in order to modify it.

2.2.2 Object-Oriented Finite Element Programming

In order to improve the development of Finite Element Analysis software, Object-Oriented Finite Element Programming was implemented. Mackerle (2004); Phongthanapanich & Dechaumphai (2006) stated that object-oriented programming improves the efficiency, extendability, re-usability and increased maintainability of large finite element software systems. This leads to smaller programs and provides better data management. This statement is agreed by Martha & Junior (2002) with addition that object-oriented programming (OOP) leads to closer integration between theory and computer implementation.

There are two main features of object oriented programming as stated by Martha & Junior (2002). The first feature is the capability of treating multi-dimensional finite element models in the same object oriented, generic fashion. This is accomplished through the definition of two object oriented programming classes namely Analysis Model and Shape. Analysis model is responsible for the handling of specific aspects related to the differential equations that governs the element behaviour while Analysis shape handles the geometric and field interpolation aspect of the element. The second feature is the generic handling of natural boundary conditions. This is implemented with the automated creation of fictitious elements responsible for translating these boundary conditions into nodal coefficients of the solution matrices and forcing vectors.

Pantale *et al.* (2004) emphasis that there are three main features of Finite Element Analysis object oriented programming. The three main features are:

1. Inheritance is a mechanism that allows the exploitation of commonality between objects. Figure 2.4 illustrates the unified modeling language diagram of many classes derived from the class Element which differ by the level of specialization that they present. Therefore, only the highly specialized

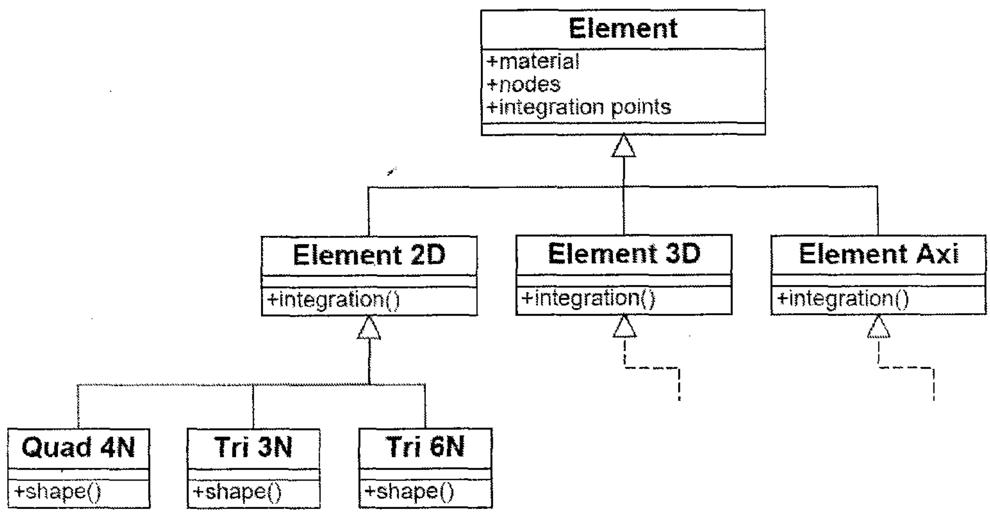


Figure 2.4: UML diagram of the element class (simplified representation) (Pantale *et al.*, 2004)

code, as shape functions calculations for example, are implemented in those derived classes.

2. Member and operator overload allows an easy writing of mathematical functions such as matrix products using a generic syntax of the form $A = B * C$ where A; B and C are three matrices of compatible sizes. The same kind of operation also is possible when the parameters are instances of different classes.

3. Template classes are generic ones, for example generic lists of any kind of object (nodes, elements integration points, etc.). Templates are the fundamental enabling technology that supports construction of maintainable highly abstract, high performance scientific codes in C++.

It can be concluded that the features of the object oriented programming are depends on the developers. The environment of the programming is the same but the approach to the programming is different. Karaoulanis *et al.* (2006) summed up the features of the object-oriented programming as below:

1. *Abstraction* of the data into objects, usually called classes, which are described by their attributes and their methods and they are capable of performing predefined actions.

2. *encapsulation*, which isolates the objects and promotes the code reuse.

3. *inheritance* of attributes or methods which permits the creation of new objects based on those already defined.

4. *polymorphism*, either *ad-hoc* by function overloading or *parametric*, by templates, which describes the ability of a single message to activate different actions when addressed to different objects. (This feature only applies to C++ programming language).

Finite Element Method is represented by the class Domain that is mainly composed by the modules represented by the abstract classes Node, Element, Material, Interface and ioDomain as shown in Figure 2.5

1. The class Node contains nodal data, such as node number, nodal coordinates, etc. Two instances of the NodalField class containing all nodal quantities at each node are linked to each node of the structure. Boundary conditions through the BoundaryCondition class affect the behaviour of each node. Those boundary conditions appears through a dynamic list attached to each node, thus, one may attach or detach any type of condition during the main solve loop.

2. The class Element is a virtual class that contains the definition of each element of the structure. This class serves as a base class for a number of other classes depending on the type of analysis and the nature of elements needed. Each element of the structure contains a number of nodes, depending on its shape, may have an arbitrary number of integration points and refers an associate constitutive law through the Material class.

3. The Interface class contains all definitions concerning the contact interfaces of the model including the contact law through the ContactLaw class and the contact definition through the Side class.

4. The class ioDomain is used to serve as an interface between the Domain and input/output files. The class ioDomain serves as a base class for many other derived classes which implement specific interfaces for various file formats. The most important of them is the class InputData used to read the model from the specific pre-processor language.

5. The class Material is used for the definition of the materials used in various models. This class is a generalization for all possible kinds of material definition.

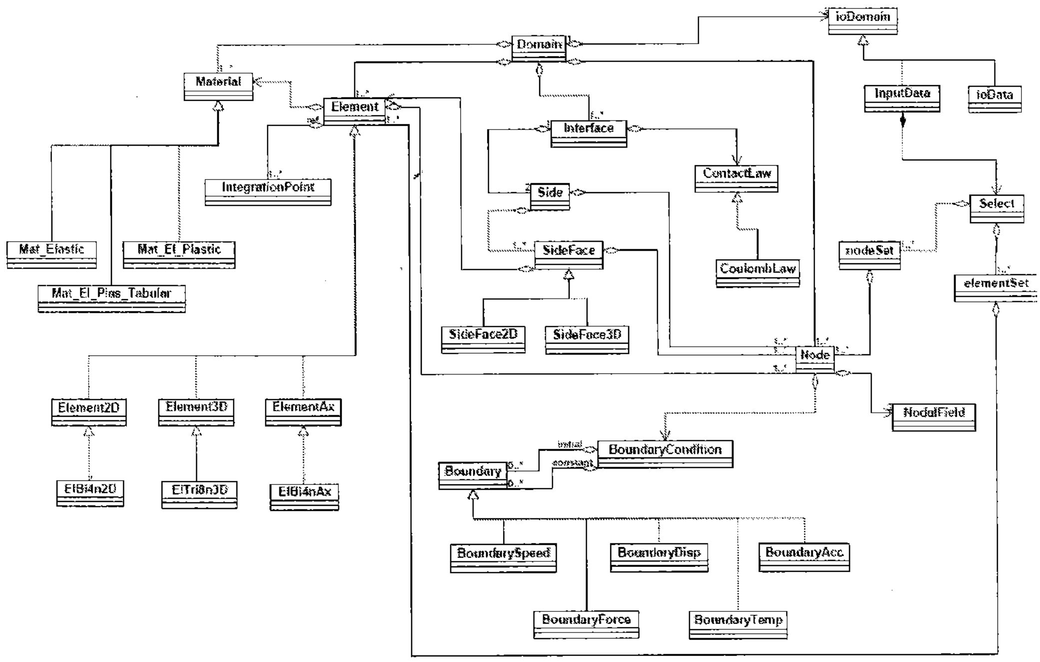


Figure 2.5: Simplified UML diagram of the object oriented framework (Pantale *et al.*, 2004)

2.2.3 Design Patterns in Object-Orienting Finite Element Programming

Object-Orienting implementation of the Finite Element Method has been around for more than 25 years. Even though the goal of developing Finite Element Analysis software is to achieve the same analysis output result, numerous approaches on how the program should be developed had been proposed and implemented. The different in software design is affected by number of factors, such as software requirement, language of the programming such as C, C++ and Fortran and the executing environment such as Windows, Macs and Linux. Developer's methodology and viewpoints also plays very important factors towards the different approach of the development of the software.

These approaches resulting in differences and similarities in each of the program design. The similarities in language independent and reusable format will be captured and used as design pattern. It is said that the design pattern is the abstraction of recurring solution to a design problem (Heng & Mackie, 2009). It captures relationships between objects participating in the solution and describes their collaboration. By reusing proven solution, design patterns help to improve software quality and reduce development time.

Gamma *et al.* (1995) documented 23 general designs patterns that are well accepted by object oriented programming developer (Heng & Mackie, 2009). However, there are problem with using the general design patterns. It requires much time and effort to identify the specific areas in which these patterns can be used. Heng & Mackie (2009) proposed five design patterns to identify the best practices in object-oriented finite element programming. The advantages of the five design patterns are listed as below:

1. Model-Analysis separation
   (a) Decompose of model and analysis subsystems produce a clearer system design. This makes both maintenance and subsequent extension of the system easier.
   (b) Minimizing dependencies across subsystem boundaries. This reduces coupling and help propagation of changes from one subsystem to another subsystem.
   (c) The changes in analysis subsystem have a little impact to the model subsystem.
2. Model –UI Separation
   (a) Clearer division of responsibilities. Model classes can remain coherent.
   (b) Changes to UI classes do not affect the model subsystem.
3. Modular element
   (a) Encapsulation is improved with element class is able to access its function in its component class through interface. Changes made in element class will not affect the component class and vice versa.
   (b) Each component object has its own few responsibilities. This keeps the class structure small and easy to be managed.
   (c) New element types can be defined by composing existing objects with ease.
   (d) New component subclasses can implemented without affecting existing subclasses or the element class.

(e) Better code reuse is achieved since the implementation encapsulated in a component class is available to all element subclasses, whether or not they share a common parent. In fact, component classes can be used by non-element classes as well.

(f) The benefits of object composition are often counter-balanced by the difficulty of abstracting functionality from the target class into logical and coherent component classes. Comprehensibility may suffer as a result of haphazard abstractions.

4. Composite element

    (a) Clients of the IElement interface can be simplified since they handle substructures and simple elements uniformly.

    (b) CompositeElem facilitates the grouping of elements into substructures for analysis using domain-decomposition methods. CompositeElem objects can be treated as independent entities for concurrent processing and distributed analysis.

    (c) With CompositeElem, it becomes easier to manage groups of elements. For example, changing the material properties of a group of elements can be as simple as updating the CompositeElem to which they belong.

5. Modular Analyzer

    (a) Decomposing the analysis subsystem into components facilitates code reuse without complicating the main hierarchy. The main procedure encapsulated in a CalcCon class can be reused with different solution strategies. Mathematical classes like CGSolver and UtDUSolver can be used with Calc- ConStaticDD as well as CalcConStatic.

    (b) The use of object composition and interfaces also increases flexibility. Existing classes are not affected by the addition of new solution strategies. Similarly, extending the system to perform, say, dynamic transient analysis has no impact on clients of the ICalcCon interface.

    (c) There is greater coherence since component objects have only a small set of responsibilities. This eases maintenance.

(d) Since mathematical classes are independent of model objects, they may be replaced by general library classes implemented by specialists with minimal impact on the rest of the system.

With many improvements that already been applied and under process of application to make the Finite Element Analysis software an ideal software for analysis purpose, the Finite Element Analysis still exhibits noticeable disadvantages. Mackie (1999) indicates that improvements need to be made of the following disadvantages.

1. Finite Element Analysis software is developed to deals with a vast range of problems covering all aspects of structure, fluids, heat transfer etc. This causes the complexity to use the software and the increase the processing time to produce an analysis output.

2. The solution methods for some area of analysis such as non-linear problems, sub-structuring and fluid-solid interaction can be very complicated.

3. Many of the associated algorithms to the Finite Element Analysis such as mesh generation are written are made of complex software.

4. A user only deals with the pre- and post-processors of the whole system. For conveniences, there is a need for a graphical user interface (GUI) at this area. Implementation of GUI will further increase the degree of difficulty of the programming.

5. Integration between FEA software and CAD software is becoming a necessity for today's industries. Examples of integration software are Pro Engineer Wildfire, Autodesk Inventor and Solid Works. Since both FEA and CAD software are developed independently to each other, compromise is required at certain aspect of the programs when the software are integrated.

6. As data are expanding, there is a need of databases. These data need to be cohesively integrated with the whole design, analysis and construction process.

2.3 Alternative to Finite Element Analysis

The evolution of the Finite Element Method had matured itself. Even though the development in improving the Finite Element Method to match today's technology is still undergoing, it already met stumbling block at certain area. Rypl & Patzak (2012) indicate that Isogeometric Analysis (IGA) introduced by Hughes *et al.* (2005) will eventually replace Finite Element Method in CAD software related environment

The main drawback with current Finite Element Method to current technology is the disability of Finite Element Method to fully integrate with CAD software. This is due to the fact that Finite Element Method and Computer Aided Design are developed by different expertise that has a different point of view on the direction of the software. Geometry model created in CAD contains ambiguities such as gap and overlaps and the level of details is not appropriate for FEM. The geometry model needs to remodel to an Analysis Suitable Geometry (ASG) that need to subjected discretization into finite element meshes. When there is a change in the design, a new ASG is required that need to once again discretization the model to finite element meshes. For FEM, meshing is the crucial aspect because it is the longest processing time is a complete FEM.

Design change in model geometry in IGA on the other hand, does not require new ASG to be remodelled. This skips the process of new discretization and re-meshing the finite element. For IGA, even though there is a change in the geometry model, the meshing shape will retain the shape as before. IGA is developed to improve the gap existing between the CAD and FEA. In doing this, IGA retains most of the features of FEA such as the modularity, extensibility, maintainability and robustness. IGA is built in the concept of isoparametric element, the same functions are used to approximate the geometry and the solution on a single finite element. Although IGA may outperform FEA in many ways, the software is still in a development stages and need more refining (Rypl & Patzak, 2012).

Although IGA may outperform FEA in many ways, the author reckons that it will take some time for the analysis software to be accepted by user. This is mainly due to Finite Element Analysis is widely used over the world and it will take time for the whole user to change to Isogeometric Analysis.

References

Barkanov, E. (2001). *INTRODUCTION TO THE FINITE ELEMENT METHOD*. Riga Technical University.

Broeckhove, J. & Vanmechelen, K. (2013). An adaptor for C++ callbacks with C and Fortran libraries. *Computer Physics Communications*, 184(3), pp. 824 – 832.

Clough, R.W. (Ed.) (1960). *The finite element method in plane stress analysis.*, volume 23, Pittsburgh: 2nd ASCE Conference on Electronic Computations.

Gamma, E., Helm, R., Johnson, R. & Vlissides, J. (1995). *Design patterns: elements of reusable object-oriented software*. Reading, Massachusetts: Addison-Wesley.

Heng, B.C.P. & Mackie, R.I. (2009). Using design patterns in object-oriented finite element programming. *Computers & Structures*, 87(15-16), pp. 952 – 961.

Hughes, T.J.R., Cottrell, J.A. & Bazilevs, Y. (2005). Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. *Computer Methods in Applied Mechanics and Engineering*, 194(39-41), pp. 4135 – 4195.

Karaoulanis, F.E., Panagiotopoulos, C.G. & Paraskevopoulos, E.A. (2006). Recent developments in Finite Element programming. In: *First South-East European Conference on Computational Mechanics*.

Mackerle, J. (2004). Object-oriented programming in FEM and BEM: a bibliography (1990-2003). *Advances in Engineering Software*, 35(6), pp. 325 – 336.

Mackie, R.I. (1999). Object-oriented finite element programming-the importance of data modelling. *Advances in Engineering Software*, 30(9-11), pp. 775 – 782.

Martha, L.F. & Junior, E.P. (2002). An Object-Oriented Framework for Finite Element Programming. In: *Fifth World Congress on Computational Mechanics*.

Nikishkov, G.P. (2006). Object oriented design of a finite element code in Java. *Computer Modeling in Engineering and Sciences*, 11, pp. 81–90.

Nikishkov, G. (2010). *Programming Finite Elements in Java*. Springer.

Pantale, O., Caperaa, S. & Rakotomalala, R. (2004). Development of an object-oriented finite element program: application to metal-forming and impact simulations. *Journal of Computational and Applied Mathematics*, 168(1-2), pp. 341 – 351.

Phongthanapanich, S. & Dechaumphai, P. (2006). EasyFEM-an object-oriented graphics interface finite element/finite volume software. *Advances in Engineering Software*, 37(12), pp. 797 – 804.

Rypl, D. & Patzak, B. (2012). From the finite element analysis to the isogeometric analysis in an object oriented computing environment. *Advances in Engineering Software*, 44(1), pp. 116 – 125.

Siswanto, W.A. & Darmawan, A.S. (2012). Teaching Finite Element Method of Structural Line Elements Assisted by Open Source FreeMat. *Research Journal of Applied Sciences, Engineering and Technology*, 4(10), pp. 1277–1286.

Zienkiewicz, O.C. & Cheung, Y.K. (1967). *The Finite Element Method in Structural and Continuum Mechanics*. McGraw-Hill.