

Unit 3: Finite Element Model and Database Model

Contents

- 1.0 Introduction
- 2.0 Intended Learning Outcomes (ILOs)
- 3.0 Main Content
 - 3.1 Finite Element Model
 - 3.2 Overview of Basic FEM
 - 3.3 Discretization
 - 3.4 Interpretation of FEM
 - 3.5 Assembly Procedure
 - 3.6 Boundary Conditions
 - 3.7 Data-based models
 - 3.8 The three perspectives of Data model
 - 3.9 Database Model
- 4.0 Self-Assessment Exercise(s)
- 5.0 Conclusion
- 6.0 Summary
- 7.0 Further Readings



1.0 Introduction

The finite element method is one of the most powerful approaches for approximate solutions to a wide range of problems in mathematical physics. The method has achieved acceptance in nearly every branch of engineering and is the preferred approach in structural mechanics and heat transfer. Its application has extended to soil mechanics, heat transfer, fluid flow, magnetic field calculations, and other areas.

Managing large quantities of **structured** and **unstructured** data is a primary function of **information systems**. Data models describe structure of **data** for **storage** in data management systems such as relational databases. They typically do not describe unstructured data, such as documents, word processing, email messages, pictures, digital audio, and video



2.0 Intended Learning Outcomes (ILOs)

After studying this unit the reader should be able to:

- Define Finite Element Method (FEM)
- Describe the relationship between FEM and Finite element analysis
- State the origin and Applications of FEM
- Describe the Basics of FEM
- Define Data modeling

- Describe the different types and the three perspectives of data models
- Have an overview of database models



3.0 Main Content

3.1 Finite Element Model

Many physical phenomena in engineering and science can be described in terms of partial differential equations (PDE). In general, solving these equations by classical analytical methods for arbitrary shapes is almost impossible. The finite element method (FEM) is a numerical approach by which these PDE can be solved approximately.

The FEM is a function/basis-based approach to solve PDE. FEs are widely used in diverse fields to solve static and dynamic problems – Solid or fluid mechanics, electromagnetic, biomechanics, etc.

The **finite element method (FEM)** (its practical application often known as **finite element analysis (FEA)**) is a numerical technique for finding approximate solutions of partial differential equations (PDE) as well as of integral equations. The solution approach is based either on eliminating the differential equation completely (steady state problems), or rendering the PDE into an approximating system of ordinary differential equations, which are then numerically integrated using standard techniques such as Euler's method, Runge-Kutta, etc.

In solving partial differential equations, the primary challenge is to create an equation that approximates the equation to be studied, but is numerically stable, meaning that errors in the input and intermediate calculations do not accumulate and cause the resulting output to be meaningless. There are many ways of doing this, all with advantages and disadvantages. The steps may be broken down as follows:

1. Definition of the physical problem:- development of the model.
2. Formulation of the governing equations - Systems of PDE, ODE, algebraic equations, define initial conditions and/or boundary conditions to get a well-posed problem,
3. Discretization of the equations.
4. Solution of the discrete system of equations.
5. Interpretation of the obtained results.
6. Errors analysis.

The Finite Element Method is a good choice for solving partial differential equations over complicated domains (like cars and oil pipelines), when the domain changes (as during a solid state reaction with a moving boundary), when the desired precision varies over the entire domain, or when the solution lacks smoothness. For instance, in a frontal crash simulation it is possible to increase prediction accuracy in "important" areas like the front of the car and reduce it in its rear (thus reducing cost of the simulation); another example would be the simulation of the weather pattern on Earth, where it is more important to have accurate predictions over land than over the wide-open sea.

3.1.1 The Finite Element Analysis

Finite Element Analysis is a method to computationally model reality in a mathematical form to better understand a highly complex problem. In the real world *everything* that occurs is as a results of interactions between atoms (and sub-particles of those atoms), billions and billions of them. If we were to simulate the world in a computer, we would have to simulate this interaction based on the simple laws of physics. However, no computer can process the near infinite number of atoms in objects, so instead we model 'finite' groups of them.

For example, we might model a gallon of water by dividing it up into 1000 parts and measuring the interaction of these linked parts. If you divide into too few parts, your simulation will be too inaccurate. If you divide into too many, your computer will sit there for years calculating the result!

3.1.2 Why use FEA?

Simulation in general is always a good idea, as it lets you test designs and ideas without spending money or effort actually building anything. By using simulation, you can find fault points within your designs, simulate ideas as you think of them, and even quantitize and optimize them. One can even use simulation to verify theories - if the theoretical simulation matches what actually happens, then the theory is proven!

Sometimes you can hand calculate certain designs. But sometimes a design can be too complex, making FEA great for non-symmetric problems with ultra-complicated geometries.

3.1.3 History of FEM

The finite element method originated from the need for solving complex elasticity and structural analysis problems in civil and aeronautical engineering. The development can be traced back to the work by Alexander Hrennikoff (1941) and Richard Courant (1942). While the approaches used by these pioneers are dramatically different, they share one essential characteristic: mesh discretization of a continuous domain into a set of discrete sub-domains, usually called elements.

Hrennikoff's work discretizes the domain by using a lattice analogy while Courant's approach divides the domain into finite triangular subregions for solution of second order elliptic partial differential equations (PDEs) that arise from the problem of torsion of a cylinder. Courant's contribution was evolutionary, drawing on a large body of earlier results for PDEs developed by Rayleigh, Ritz, and Galerkin.

Development of the finite element method began in earnest in the middle to late 1950s for airframe and structural analysis and gathered momentum at the University of Stuttgart through the work of John Argyris and at Berkeley through the work of Ray W. Clough in the 1960s for use in civil engineering. By late 1950s, the key concepts of stiffness matrix and element assembly existed essentially in the form used today. NASA issued a request

for proposals for the development of the finite element software NASTRAN in 1965. The method was again provided with a rigorous mathematical foundation in 1973 with the publication of Strang and Fix's *An Analysis of The Finite Element Method* has since been generalized into a branch of applied mathematics for numerical modelling of physical systems in a wide variety of engineering disciplines, e.g., electromagnetism, thanks to Peter P. Silvester and fluid dynamics.

3.1.4 Applications of FEM

A variety of specializations under the umbrella of the mechanical engineering discipline (such as aeronautical, biomechanical, and automotive industries) commonly use integrated FEM in design and development of their products. Several modern FEM packages include specific components such as thermal, electromagnetic, fluid, and structural working environments. In a structural simulation, FEM helps tremendously in producing stiffness and strength visualizations and also in minimizing weight, materials, and costs.

3.2 Overview of Basic FEM

The basic steps of the finite element method are discussed next in more generality. Although attention is focused on structural problems, most of the steps translate to other applications problems as noted above. The role of FEM in numerical simulation is schematized in figure 1 below.

This diagram displays the three key simulation steps: idealization, discretization and solution. It also indicates the fact that each step introduces different of errors. For example the discretization error is the discrepancy obtained when the discrete solution is substituted in the mathematical model.

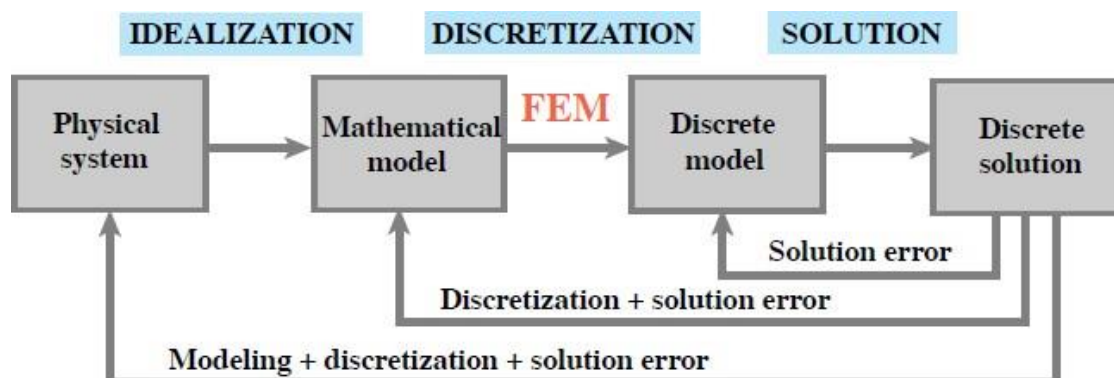


Figure1: Steps of the physical simulation process: idealization, discretization and solution.

3.2.1 Idealization

a. Models

The word “model” has the traditional meaning of a scaled copy or representation of an object. And that is precisely how most dictionaries define it. We use here the term in a more modern sense, increasingly common since the advent of computers:

A model is a symbolic device built to simulate and predict aspects of behaviour of a system.

Note the careful distinction made between “behaviour” and “aspects of behaviour.” To predict everything, in all physical scales, you must deal with the actual system. A model

abstracts aspects of interest to the modeler. The term “symbolic” means that a model represents a system in terms of the symbols and language of another science. For example, engineering systems may be (and are) modeled with the symbols of mathematics and/or computer sciences.

b. Mathematical Models

Mathematical modelling, or *idealization*, is a process by which the engineer passes from the actual physical system under study, to a *mathematical model* of the system, where the term *model* is understood in the wider sense defined above.

The process is called *idealization* because the mathematical model is necessarily an abstraction of the physical reality. (Note the phrase *aspects of behaviour* in the definition.) The analytical or numerical results obtained for the mathematical model are re-interpreted in physical terms only for those aspects.

Why is the mathematical model an abstraction of reality?

Engineering systems such as structures tend to be highly complex. To simulate its behaviour it is necessary to reduce that complexity to manageable proportions. Mathematical modelling is an abstraction tool by which complexity can be brought under control. This is achieved by “filtering out” physical details that are not relevant to the analysis process. For example, a continuum material model necessarily filters out the aggregate, crystal, molecular and atomic levels of matter. If you are designing a bridge or building such levels are irrelevant. Consequently, choosing a mathematical model is equivalent to choosing an information filter.

3.2.2 Implicit vs. Explicit Modelling

Suppose that you have to analyze a structure and at your disposal is a “black box” general-purpose finite element program. This is also known in the trade as a “canned program.” Those programs usually offer a *catalog* of element types; for example, bars, beams, plates, shells, axisymmetric solids, general 3D solids, and so on. The moment you choose specific elements from the catalog you automatically accept the mathematical models on which the elements are based. This is *implicit modelling*, a process depicted in Figure 2.

Ideally you should be fully aware of the implications of your choice. Providing such “finite element literacy” is one of the objectives of this course.

Unfortunately many users of commercial programs are unaware of the “implied consent” aspect of implicit modelling.

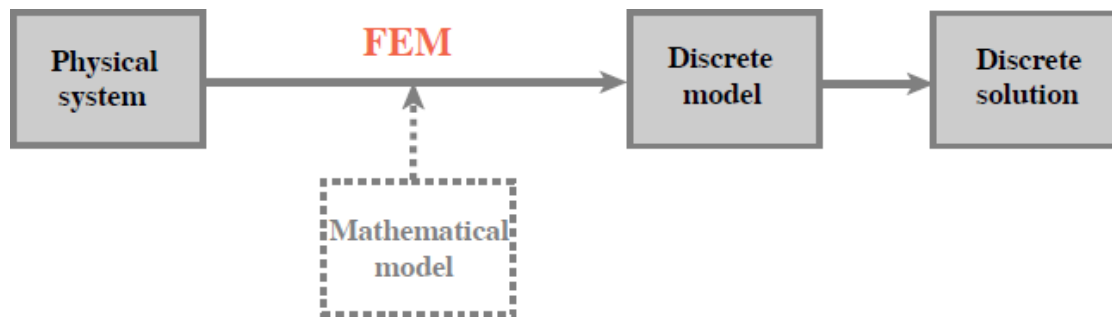


Figure 2: Implicit modelling: picking elements from an existing FEM code implicitly accepts an idealization. Read the fine print.

The other extreme occur when you select a mathematical model of the physical problem with your eyes wide open and *then* either shop around for finite element programs that implements that model, or write the program yourself. This is *explicit modelling*. It requires far more technical expertise, resources, experience and maturity than implicit modelling. But for problems that fall out of the ordinary it may be the right thing to do.

In practice a combination of implicit and explicit modelling is quite common. The physical problem to be solved is broken down into subproblems. Those subproblems that are conventional and fit existing programs may be treated with implicit modelling. Those subproblems that require special handling may yield only to explicit modelling treatment.

3.3 Discretization

3.3.1 Purpose

Mathematical modelling is a simplifying step. But models of physical systems are not necessarily easy to solve. They usually involve coupled partial differential equations in space and time subject to boundary and/or interface conditions. Such analytical models have an *infinite* number of degrees of freedom.

At this point one faces the choice of trying for analytical or numerical solutions. Analytical solutions, also called “closed form solutions,” are more intellectually satisfying, particularly if they apply to a wide class of problems. Unfortunately they tend to be restricted to regular geometries and simple boundary conditions. Moreover a closed- form solution, expressed for example as the inverse of an integral transform, often has to be numerically evaluated to be useful.

Most problems faced by the engineer either do not yield to analytical treatment or doing so would require a disproportionate amount of effort. The practical way out is numerical simulation. Here is where finite element methods and the digital computer enter the scene.

To make numerical simulations practical it is necessary to reduce the number of degrees of freedom to a *finite* number. The reduction is called *discretization*. The end result of the discretization process is the *discrete model* depicted in Figures 1 and 2.

Discretization can proceed in space dimensions as well as in the time dimension. Because the present course deals only with static problems, we need not consider the time dimension and are free to concentrate on *spatial discretization*.

3.3.2 Error Sources and Approximation

Figure 1 tries to convey graphically that each simulation step introduces a source of error. In engineering practice modelling errors are by far the most important. But they are difficult and expensive to evaluate, because such *model validation* requires access to and comparison with experimental results.

Next in order of importance is the *discretization error*. Even if solution errors are ignored and usually they can, the computed solution of the discrete model is in general only an approximation in some sense to the exact solution of the mathematical model. A quantitative measurement of this discrepancy is called the *discretization error*.

The characterization and study of this error is addressed by a branch of numerical mathematics called approximation theory.

Intuitively one might suspect that the accuracy of the discrete model solution would improve as the number of degrees of freedom is increased, and that the discretization error goes to zero as that number goes to infinity. This loosely worded statement describes the *convergence* requirement of discrete approximations. One of the key goals of approximation theory is to make the statement as precise as it can be expected from a branch of mathematics.

3.3.3 Finite and Boundary Element Methods

The most popular discretization procedures in structural mechanics are finite element methods and boundary element methods. The finite element method (FEM) is by far the most widely used. The boundary element method (BEM) has gained in popularity for special types of problems, particularly those involving infinite domains, but remains a distant second.

In non-structural application areas such as fluid mechanics and thermal analysis, the finite element method is gradually making up ground but faces stiff competition from both the classical and energy-based *finite difference* methods. Finite difference and finite volume methods are particularly well entrenched in computational fluid dynamics.

3.4 Interpretation of FEM

The finite element method (FEM) is the dominant discretization technique in structural mechanics. FEM can be interpreted from either a physical or mathematical standpoint. The treatment has so far emphasized the former.

The basic concept in the physical interpretation of the FEM is the subdivision of the mathematical model into disjoint (non-overlapping) components of simple geometry called *finite elements* or *elements* for short. The response of each element is expressed in terms of a finite number of degrees of freedom characterized as the value of an unknown function,

or functions, at a set of nodal points.

The response of the mathematical model is then considered to be approximated by that of the discrete model obtained by connecting or assembling the collection of all elements.

The disconnection-assembly concept occurs naturally when examining many artificial and natural systems. For example, it is easy to visualize an engine, bridge, building, airplane, or skeleton as fabricated from simpler components.

Unlike finite difference models, finite elements *do not overlap* in space. In the mathematical interpretation of the FEM, this property goes by the name *disjoint support*.

FEM Element Attributes

Just like the members in the truss example, one can take finite elements of any kind one at a time. Their local properties can be developed by considering them in isolation, as individual entities. This is the key to the programming of element libraries.

In the Direct Stiffness Method, elements are isolated by disconnection and localization. This procedure involves the separation of elements from their neighbors by disconnecting the nodes, followed by the referral of the element to a convenient local coordinate system. After these two steps we can consider *generic* elements: a bar element, a beam element, and so on. From the standpoint of computer implementation, it means that you can write one subroutine or module that constructs all elements of one type, instead of writing one for each element instance.

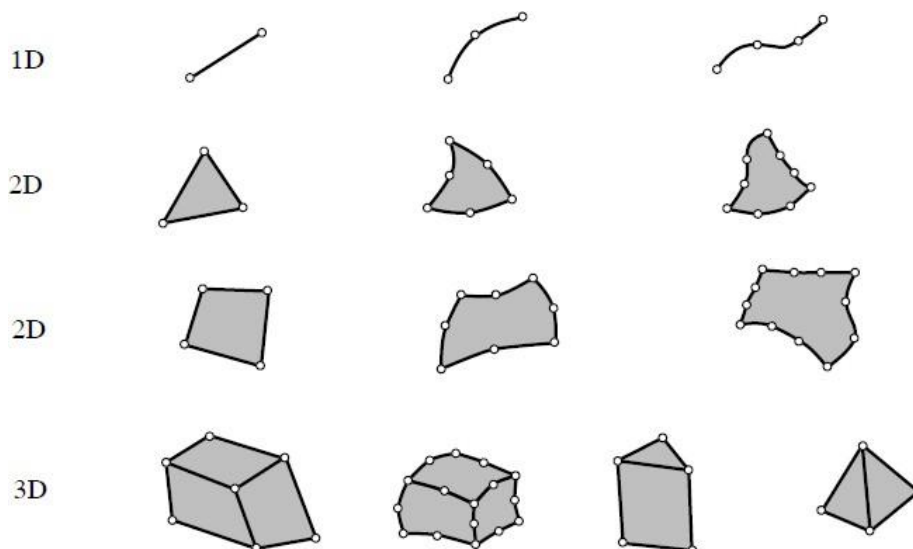


Figure 3: Typical finite element geometries in one through three dimensions

The following is a summary of the data associated with an individual finite element. This data is used in finite element programs to carry out element level calculations.

Dimensionality. Elements can have one, two or three space dimensions. (There are also

special elements with zero dimensionality, such as lumped springs.)

Nodal points. Each element possesses a set of distinguishing points called *nodal points* or *nodes* for short. Nodes serve two purposes: definition of element geometry, and home for degrees of freedom. They are located at the corners or end points of elements (see Figure 3); in the so-called refined or higher-order elements nodes are also placed on sides or faces.

Geometry. The geometry of the element is defined by the placement of the nodal points. Most elements used in practice have fairly simple geometries. In one-dimension, elements are usually straight lines or curved segments. In two dimensions they are of triangular or quadrilateral shape.

In three dimensions the three common shapes are tetrahedra, pentahedra (also called wedges or prisms), and hexahedra (also called cuboids or “bricks”). See Figure 3.

Degrees of freedom. The degrees of freedom (DOF) specify the *state* of the element. They also function as “handles” through which adjacent elements are connected. DOFs are defined as the values (and possibly derivatives) of a primary field variable at nodal points.

The actual selection depends on criteria studied at length in Part II. Here we simply note that the key factor is the way in which the primary variable appears in the mathematical model. For mechanical elements, the primary variable is the displacement field and the DOF for many (but not all) elements are the displacement components at the nodes.

Nodal forces. There is always a set of nodal forces in a one-to-one correspondence with degrees of freedom. In mechanical elements the correspondence is established through energy arguments.

Constitutive properties. For a mechanical element these is the relation that specifies the material properties. For example, in a linear elastic bar element it is sufficient to specify the elastic modulus E and the thermal coefficient of expansion.

Fabrication properties. For a mechanical element these are fabrication properties which have been integrated out from the element dimensionality. Examples are cross sectional properties of MoM elements such as bars, beams and shafts, as well as the thickness of a plate or shell element.

This data is used by the element generation subroutines to compute element stiffness relations in the local system.



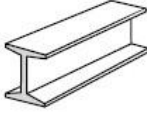

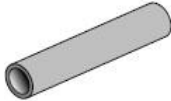




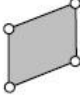
Physical Structural Component	Mathematical Model Name	Finite Element Discretization
	bar	
	beam	
	tube, pipe	
	spar (web)	
	shear panel (2D version of above)	

Figure 4. Examples of primitive structural elements

3.5 Assembly Procedure

The assembly procedure of the Direct Stiffness Method for a general finite element model follows rules identical in principle to those discussed for the truss example. As in that case the process involves two basic steps:

Globalization. The element equations are transformed to a common *global* coordinate system.

Merge. The element stiffness equations are merged into the master stiffness equations by appropriate indexing and entry addition.

The computer implementation of this process is not necessarily as simple as the hand calculations of the truss example suggest. The master stiffness relations in practical cases may involve thousands (or even millions) of degrees of freedom. To conserve storage and processing time the use of sparse matrix techniques as well as peripheral storage is required. But this inevitably increases the programming complexity.

3.6 Boundary Conditions

A key strength of the FEM is the ease and elegance with which it handles arbitrary boundary and interface conditions. This power, however, has a down side. One of the biggest hurdles a FEM newcomer faces is the understanding and proper handling of boundary conditions. Surprisingly, prior exposure to partial differential equations, without a balancing study of variational calculus, does not appear to be of much help in this regard.

In the present Section we summarize some basic rules for treating boundary conditions.

3.6.1 Essential and Natural B.C.

The important thing to remember is that boundary conditions (BCs) come in two basic flavors:

Essential BCs are those that directly affect the degrees of freedom, and are imposed on the left-hand side vector \mathbf{u} .

Natural BCs are those that do not directly affect the degrees of freedom, and are imposed on the right-hand side vector \mathbf{f} .

The mathematical justification for this distinction requires use of the variation calculus, and is consequently relegated to Part II of the course. For the moment, the basic recipe is:

1. If a boundary condition involves one or more degrees of freedom in a *direct* way, it is essential. An example is a prescribed node displacement.
2. Otherwise it is natural.

The term “direct” is meant to exclude derivatives of the primary function, unless those derivatives also appear as degrees of freedom, such as rotations in beams and plates.

3.6.2 Boundary Conditions in Structural Problems

In mechanical problems, essential boundary conditions are those that involve *displacements* (but not strain-type displacement derivatives). The support conditions for the truss problem furnish a particularly simple example. But there are more general boundary conditions that occur in practice.

A structural engineer must be familiar with displacement B.C. of the following types.

Ground or support constraints. Directly restraint the structure against rigid body motions.

Symmetry conditions. To impose symmetry or antisymmetry restraints at certain points, lines or planes of structural symmetry. This allows the discretization to proceed only over part of the structure with a consequent savings in the number of equations to be solved.

Ignorable freedoms. To suppress displacements that are irrelevant to the problem. (In classical dynamics these are called *ignorable coordinates*.) Even experienced users of finite element programs are sometimes baffled by this kind.

Connection constraints. To provide connectivity to adjoining structures or substructures, or to specify relations between degrees of freedom. Many conditions of this type fall under the label.

multipoint constraints or *multifreedom constraints*, which can be notoriously difficult to handle from a numerical standpoint.

FEM allows detailed visualization of where structures bend or twist, and indicates the distribution of stresses and displacements. FEM software provides a wide range of simulation options for controlling the complexity of both modelling 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. FEM allows entire designs to be constructed, refined, and optimized before the design is manufactured.

This powerful design tool has significantly improved both the standard of engineering designs and the methodology of the design process in many industrial applications. The introduction of FEM has substantially decreased the time to take products from concept to the production line. It is primarily through improved initial prototype designs using FEM that testing and development have been accelerated. In summary, benefits of FEM include increased accuracy, enhanced design and better insight into critical design parameters, virtual prototyping, fewer hardware prototypes, a faster and less expensive design cycle, increased productivity, and increased revenue.

3.7 Data-based models

Data modelling is a method used to define and analyze data requirements needed to support the business processes of an organization. The data requirements are recorded as a conceptual data model with associated data definitions. Actual implementation of the conceptual model is called a logical data model. To implement one conceptual data model may require multiple logical data models.

Data modelling defines not just data elements, but their structures and relationships between them. Data modelling techniques and methodologies are used to model data in a standard, consistent, predictable manner in order to manage it as a resource. The use of data modelling standards is strongly recommended for all projects requiring a standard means of defining and analyzing data within an organization, e.g., using data modelling:

- to manage data as a resource;
- for the integration of information systems;
- for designing databases/data warehouses (aka data repositories)

Data modelling may be performed during various types of projects and in multiple phases of projects. Data models are progressive; there is no such thing as the final data model for a business or application. Instead a data model should be considered a living document that will change in response to a changing business. The data models should ideally be stored in a repository so that they can be retrieved, expanded, and edited over time.

Types of Data Models

Whitten (2004) determined two types of data modelling:

- Strategic data modelling: This is part of the creation of an information systems strategy, which defines an overall vision and architecture for information systems defined. Information engineering is a methodology that embraces this approach.
- Data modelling during systems analysis: In systems analysis logical data models are created as part of the development of new databases.

Data modelling is also a technique for detailing business requirements for a database. It is sometimes called *database modelling* because a data model is eventually implemented in a database.

The main aim of data models is to support the development of data-base by providing the definition and format of data. According to West and Fowler (1999) "if this is done consistently across systems then compatibility of data can be achieved. If the same data structures are used to store and access data then different applications can share data.

However, systems and interfaces often cost more than they should, to build, operate, and maintain. They may also constrain the business rather than support it. A major cause is that the quality of the data models implemented in systems and interfaces is poor. As a consequence:

- Business rules, specific to how things are done in a particular place, are often fixed in the structure of a data model. This means that small changes in the way business is conducted lead to large changes in computer systems and interfaces
- Entity types are often not identified, or incorrectly identified. This can lead to replication of data, data structure, and functionality, together with the attendant costs of that duplication in development and maintenance
- Data models for different systems are arbitrarily different. The result of this is that complex interfaces are required between systems that share data. These interfaces can account for between 25-70% of the cost of current systems
- "Data cannot be shared electronically with customers and suppliers, because the structure and meaning of data has not been standardised. For example, engineering design data and drawings for process plant are still sometimes exchanged on paper

The reason for these problems is a lack of standards that will ensure that data models will both meet business needs and be consistent.

3.8 The three perspectives of Data model

The perspectives shows that a data model can be an external model (or view), a conceptual model, or a physical model. This is not the only way to look at data models, but it is a useful way, particularly when comparing models.

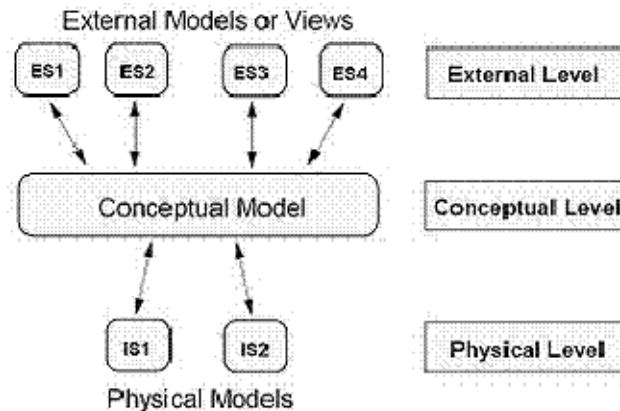


Fig. 6: ANSI/SPARC **three level architecture**.

A data model *instance* may be one of three kinds according to ANSI in 1975:

Conceptual schema - A conceptual schema specifies the kinds of facts or propositions that can be expressed using the model. In that sense, it defines the allowed expressions in an artificial 'language' with a scope that is limited by the scope of the model. It describes the semantics of a domain. For example, it may be a model of the interest area of an organization or industry. This consists of entity classes, representing kinds of things of significance in the domain, and relationships assertions about associations between pairs of entity classes. The use of conceptual schema has evolved to become a powerful communication tool with business users. Often called a subject area model (SAM) or high-level data model (HDM), this model is used to communicate core data concepts, rules, and definitions to a business user as part of an overall application development or enterprise initiative. The number of objects should be very small and focused on key concepts. Try to limit this model to one page, although for extremely large organizations or complex projects, the model might span two or more pages

Logical schema - describes the semantics, as represented by a particular data manipulation technology. This consists of descriptions of tables and columns, object oriented classes, and XML tags, among other things.

Physical schema - describes the physical means by which data are stored. This is concerned with partitions, CPUs, tablespaces, and the like.

The significance of this approach, according to ANSI, is that it allows the three perspectives to be relatively independent of each other.

Storage technology can change without affecting either the logical or the conceptual model. The table/column structure can change without (necessarily) affecting the conceptual model. In each case, of course, the structures must remain consistent with the other model. The table/column structure may be different from a direct translation of the entity classes and attributes, but it must ultimately carry out the objectives of the conceptual entity class structure. Early phases of many software development projects

emphasize the design of a **conceptual data model** . Such a design can be detailed into a **logical data model** . In later stages, this model may be translated into **physical data model**. However, it is also possible to implement a conceptual model directly

3.9 Database Model

A **database model** is a theory or specification describing how a database is structured and used. Several such models have been suggested. Common models include:

Flat model: This may not strictly qualify as a data model. The flat (or table) model consists of a single, two-dimensional array of data elements, where all members of a given column are assumed to be similar values, and all members of a row are assumed to be related to one another.

Hierarchical model: In this model data is organized into a tree-like structure, implying a single upward link in each record to describe the nesting, and a sort field to keep the records in a particular order in each same-level list.

Network model: This model organizes data using two fundamental constructs, called records and sets. Records contain fields, and sets define one-to-many relationships between records: one owner, many members.

Relational model: is a database model based on first-order predicate logic. Its core idea is to describe a database as a collection of predicates over a finite set of predicate variables, describing constraints on the possible values and combinations of values.



Object-relational model: Similar to a relational database model, but objects, classes and inheritance are directly supported in database schemas and in the query language.



4.0 Self-Assessment Exercise(s)

Answer the following questions:

1. Draw the FEM physical simulation process
2. Briefly discuss each of the data associated with FE used in programs for elementary calculations.
3. With the aid of diagrams differentiate between the common data model
4. Briefly discuss the basic rules for the treatment of boundary conditions
5. What is FEM and how/where can it be applied?
6. How does the ANSI three perspectives to data model allows for relatively independent of each.
7. What is the purpose of FEM discretization



5.0 Conclusion

The development of systems and interfaces often cost more than they should, to build, operate, and maintain. A major cause is that the quality of the data models implemented in systems and interfaces is poor. This usually is as a result of:

- Violation of business rules, as a result small changes in the way business is conducted lead to large changes in computer systems and interfaces.
- Unidentified or incorrect identification of entity which can lead to replication of data, data structure, and functionality, and increased costs of development and maintenance

Consequently data cannot be shared electronically with customers and suppliers due to unstructured and lack of standard data that can meet business needs.



6.0 Summary

In this unit, we have discussed elaborately,

- the Physics-based Finite Element Method (FEM) which is defined as a numerical technique for finding approximate solutions of partial differential equations (PDE) as well as of integral equations.
 - Here we stated the uses, traced its origin and discussed its applications
 - Carried out the overview of the FEM basics including:
 - Idealization
 - Discretization; purpose and error sources
 - Finite and Boundary element methods
 - Interpretations of FEM and
 - Elementary attributes used in FEM
- Stated the three perspectives of data models; Conceptual, logical and Physical schemas.
- The different types of database models; Flat, hierarchical, network, Relational and Object oriented models.



7.0 Further Readings

- Gordon, S. I., & Guilfoos, B. (2017). *Introduction to Modeling and Simulation with MATLAB® and Python*. Milton: CRC Press.
- Zeigler, B. P., Muzy, A., & Kofman, E. (2019). *Theory of modeling and simulation: Discrete event and iterative system computational foundations*. San Diego (Calif.): Academic Press.
- Kluever, C. A. (2020). *Dynamic systems modeling, simulation, and control*. Hoboken, N.J: John Wiley & Sons.

- Law, A. M. (2015). *Simulation modeling and analysis*. New York: McGraw-Hill.
- Verschuuren, G. M., & Travise, S. (2016). *100 Excel Simulations: Using Excel to Model Risk, Investments, Genetics, Growth, Gambling and Monte Carlo Analysis*. Holy Macro! Books.
- Grigoryev, I. (2015). *AnyLogic 6 in three days: A quick course in simulation modeling*. Hampton, NJ: AnyLogic North America.
- Dimotikalis, I., Skiadas, C., & Skiadas, C. H. (2011). *Chaos theory: Modeling, simulation and applications: Selected papers from the 3rd Cghaotic Modeling and Simulation International Conference (CHAOS2010), Chania, Crete, Greece, 1-4 June, 2010*. Singapore: World Scientific.
- Velten, K. (2010). *Mathematical modeling and simulation: Introduction for scientists and engineers*. Weinheim: Wiley-VCH.