Object Oriented Storage of Material Data for Coupled Problems

Johan Driesen, Johan Fransen, Herbert De Gersem, Ronnie Belmans, Kay Hameyer
KATHOLIEKE UNIVERSITEIT LEUVEN, DEP. EE (ESAT), DIV. ELEN
Kardinaal Mercierlaan 94
B-3001 Heverlee, BELGIUM

Abstract — **The calculation of coupled field problems in order to obtain realistic numerical models becomes more and more important. Coupling strategies require a flexible data structure to access the required material data. An object oriented data structure offers several features proving to be useful to store these material characteristics. This includes different tensor representations of the characteristic and different mathematical descriptions as tables and formulae. Hence, an easy and robust environment for the CAD-programmer becomes available.**
The use of these data structures is demonstrated by two typical coupled magneto-thermal and a electro-thermal computations.

Index terms — **Finite element methods, object oriented programming, motor drives, losses, eddy currents, dielectric losses**

I. INTRODUCTION

The fast availability of the correct material characteristics in the appropriate shape speeds up the time for matrix assembly necessary in finite element method approximations or for post-processing. In coupled problems, a lot of parameters related to several physical phenomena have to be known. These parameters have different physical representations ranging from just a real number to a large symmetrical complex tensor. The mathematical descriptions of the numerical values take different forms, for instance an analytical formula or a multidimensional table.

Most characteristics depend on several physical variables making non-linear couplings a necessity. A data structure suited for handling the interdependencies and all the possible physical and mathematical representations can simplify the solution process of coupled problems.

II. COUPLING STRATEGIES

Coupling can be accomplished in two different ways: full coupling (numerical strong) and cascade coupling (numerical weak). Both strategies use different approaches in linking the systems of coupled partial differential equations describing the different physical effects of the problem.

Full or Numerical Strong Coupling

A very obvious way is to directly treat the coupled system by solving the complete set of non-linear equations. After discretization, this may lead to an extensive system of algebraic equations describing phenomena with different time constants [1]. To linearize this coupled system of equations a database containing sufficient information to determine the interdependencies (partial derivatives) of the material characteristics on the solution variables has to be present.

Cascade or Numerical Weak Coupling

An alternative strategy consists in the decomposition of the problem into several subproblems, each with the appropriate pre- and post-processing, solved sequentially in an iteration loop until convergence is reached. Intermediately, the material parameters are adjusted based on the solutions of the previous steps [2],[3].

Within this weak coupling strategy, two types of parameters on which the material data are depending can be identified. First, the external parameters are derived from the previous calculations. These parameters do not change during the following calculation step. On the other hand, the internal parameters are the solution variables. They may change during the calculation and have to be treated properly.

III. OBJECT ORIENTED MATERIAL DATA STORAGE

Unlike the classical structured programming method linking functions and the appropriate variables, object oriented programming languages, like C++ [4], place the data into the center of the programme design [5]. This offers the following advantages:

- *Data Hiding:* Data structures are encapsulated and can only be accessed through "member functions" or "overloaded operators". This offers a robust and rigid structure to protect the data from unwanted changes. Advanced compilers may produce a fast inline coding of the functions accessing the data.
- *Polymorphism:* In classical programming a lot of explicit "if-then-else"-like clauses would have to be checked to determine the kind of mathematical description of the material and the related evaluation function or operator. In an object oriented structure, the appropriate functions are immediately and implicitly determined by means of a virtual function table. All the programmer has to know is how to call a kind of *"give_value()"* function becoming subsequently overloaded with the correct function. This enables a programmer to implement abstract and general formulae.
- *Inheritance:* The object oriented library can very quickly be widened by code due to the inheritance property. The existing objects and their properties are "inherited" and modified, to implement new kinds of material characteristics. The software developing programmer is forced to implement a minimum set of necessary functions

Manuscript received November 3, 1997.
J.Driesen, +32/16/32.10.20, fax +32/32/16/32.19.85, johan.driesen@esat.kuleuven.ac.be, http://www.esat.kuleuven.ac.be/elen/elen.html.
J.Driesen holds a research assistantship of the Belgian "Fonds voor Wetenschappelijk Onderzoek - Vlaanderen"

because of the virtual function system and the abstract classes.

IV. Implementation

The object oriented library containing the material characteristics is programmed by means of C++-classes. It consists of two object structures: the *tensor*-objects, providing the physical structure, and the *charac*-objects, containing the structure to implement different mathematical material representations.

Tensor-Objects

Material data is retained in a class hierarchy (Fig. 1). The base class *tensorall* is a pure abstract class. The other classes are inherited from it. It contains virtual functions meant to return a numerical value or to copy the object. The derived classes enable the implementation of properties represented by a single value (*tensorconst*), a diagonal tensor (*tensordiag*) containing two values in case of two-dimensional simulations or six for 3D, or a symmetrical tensor (*tensorfull*) containing three or six elements. These "values" are stored in the structure as charac-objects.

The *tensorlist* is a special structure used in FEM-computations. It offers a dynamic structure to store material data per mesh element, independent of the kind of representation of the material characteristic. This structure is not only used in certain coupled FEM-computations, but also in non-coupled calculations, for instance to "freeze" the saturation of an electromagnetic device in a linearized operation point to investigate saturated harmonic fields.

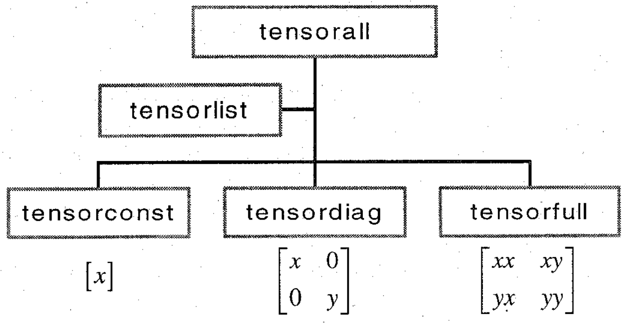
tensorall
tensorlist
tensorconst
tensordiag
tensorfull
$[x]$
$\begin{bmatrix} x & 0 \\ 0 & y \end{bmatrix}$
$\begin{bmatrix} xx & xy \\ yx & yy \end{bmatrix}$

Fig. 1: Structure of the Tensor-Object.

Charac-Objects

The abstract class *characall* (Fig. 2) contains virtual functions representing the basic operations. The inherited classes enables the storage of constant values, expressions (formulae) or tables with the appropriate interpolation functions. Most of the time, bicubic splines are implemented for this purpose. Once the spline coefficients are calculated in the object constructor, it is easy to obtain interpolates, derivatives or integrated values by calling a simple function. The dispatch of the different ways of material representations is done by the function table of *characall.*

The *characproc* object is a special structure to be used for material descriptions depending on the interval in which a parameter is situated, e.g. different formulations below or beyond the Curie temperature.

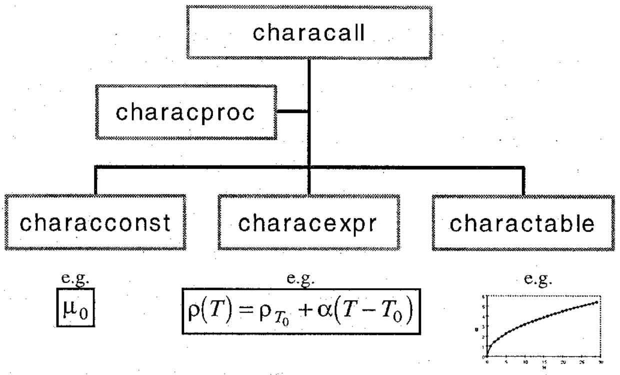

Fig. 2: Structure of the Charac-Object.

User Interface

A programmer using these data structures does not have to know about the implementation behind it. Even the fact whether the data originally was provided in table or formula format is of secondary importance. Therefore, the only thing that has to be coded by the programmer into the code is the call to the constructing functions (in the C++ implementation, constructors searching for the appropriate database file containing the characteristic representations, hereafter the structures are set up). Then, only simple calls masked behind functions or operators such as round or squared brackets on the right place in the code (e.g. *region[k].cond()*) are necessary to obtain the desired numerical value. Thus, a natural looking notation of the formulae is used, even clear for non-programming specialists.

In this way, the described data structures are used in different FEM-solvers, post-processing, visualization code and the coupling programmes that form the skeleton of a cascade coupled algorithm as discussed before. Hence, it is ensured that the same data values are used everywhere during the execution of the programs

Code Organization and Implementation

Most coupled simulations involve solving Laplace-like equations. It is very profitable to use a common general base FEM-code for Laplace-like equation solving. The flexibility of the solver lies in the material handling, providing the right coefficients to solve the equations at stake describing the thermal, electric or magnetic problem. By using an object oriented material implementation combined with the mathematical code, a relatively small and efficient flexible coupled solver is obtained.

By reusing the same material library for the solvers, coupling programs and post-processors, the consistency of the material properties in the code is guaranteed.

The cost of this implementation is a small overhead of allocated memory, originating from the construction of the virtual function tables. In general, the material diversity within the models is much smaller than the number of elements, so usually this is no problem. The run-time determination of the right member-function causes some overhead in computation time. A comparable flexible code in a non-object oriented language would need the same overhead in computation time.

IV. Application Examples

Magneto-Thermal Coupled Computation Problem

The materials used in magnetic devices are often temperature sensitive. The flux from permanent magnets, in the rare-earth material, depends on the temperature. All magnetic materials lose their properties beyond a certain transition temperature, like the Curie temperature.

On the other hand, these materials are often a source of heat, since they show ferromagnetic losses. To represent these, tables or formulae with matched coefficients may be used [6]. These heat sources and joule losses from eddy currents are usually unwanted in machines based on a driving electro-magnetic field. However, sometimes these effects can be used positively in induction heating applications.

Ex. 1: Induction Machine

The total behavior of induction machines is governed by the following 2D-differential equations:

$$\nabla\left(\upsilon(A,T)\nabla A\right) - j\omega\sigma(T)A + \frac{\sigma(T)}{\nu_0}\nabla(V) = 0$$
$$\nabla\left(k(T)\nabla T\right) = -\frac{J_s^{\ 2}}{\sigma(T)} - p_{Fe} + \sigma(T)\omega^2 A^2 \qquad (1)$$

with A, the magnetic vectorpotential ($\nabla \times \vec{A} = \vec{B}$) and V, the voltage applied across conducting regions.

The first equation leads to the force-generating magnetic field. The second equation describes the thermal field and its sources. For this type of machines, the representation of the reluctivity tensor is of large importance, since the saturation forms a highly non-linear aspect of the machine. This makes the simulation a highly non-linear calculation. The convergence of these problems is influenced by the derivatives of the non-linear parameters, particularly $\nu(B)$, so a correct interpretation of this characteristic is recommended.

The temperature dependence of the conductivity σ influences both equations and makes the resulting system of equations strongly coupled. It also influences the algebraic circuit equations necessary to represent the end-effects [7]. This characteristic has a large influence on the total convergence as well.

The reluctivity ν can be an anisotropic tensor described by spline-interpolated 2D-tables with inputs *B(A)* and *T*. The electrical conductivity σ is generally described by a rational function of *T*, whereas the thermal conductivity *k* is described by a procedure approximated by a constant below a certain transition temperature and depending on *T* above this temperature. However in almost all machines, this temperature is never reached (unlike induction heating applications).

Fig. 3 shows such a calculation. The machine under study is a 4-pole 400 kW traction induction motor [8]. The hot spots in the machine can be determined and a more accurate knowledge of the machine behavior is obtained by considering the thermal effects.

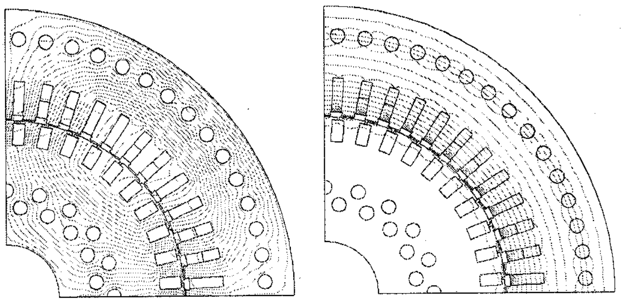

Fig. 3: 4 pole induction motor (a) plot of the magnetic field and (b) the isothermals obtained by the thermal computation

Ex. 2: Permanent Magnet Machine

The behavior of permanent magnet machines is described by equations (2):

$$\nabla\left(\nu(A,T)\nabla A\right) = -\frac{J_s}{\nu_0} - \nabla\left(\nu_{rev} M\right)$$
$$\nabla\left(k(T)\nabla T\right) = -\frac{J_s^{\ 2}}{\sigma(T)} - p_{Fe} \qquad (2)$$

with A the magnetic vectorpotential ($\nabla \times \vec{A} = \vec{B}$).

This calculation shows a lot of similarities with ex.1. Here, the permanent magnet term introduces a strongly temperature dependent tensor. This temperature dependence is typical for certain types of materials. Considering the ferromagnetic properties, ferrites usually have positive temperature coefficients, whereas rare-earth materials may have negative temperature coefficients. This is described in tables or simplified in formulae. In either way, it has to be checked (in a procedure) whether the magnet still produces a field for a given temperature and whether it is not demagnetized irreversibly yet.

The example in Fig. 4 can only be calculated correctly in a coupled way. It represents a small motor for automotive applications. Therefore its behavior has to be simulated at different environment temperatures.

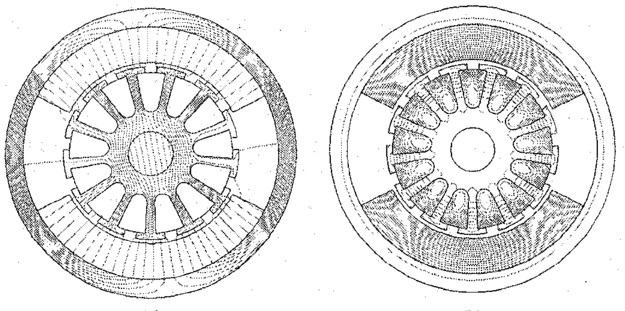
(a) (b)

Fig. 4: (a) Plot of the magnetic flux lines of a small permanent magnet machine, (b) isothermal lines

Electro-Thermal Coupled Calculation

Another application of extensive coupled simulations are the precise modeling of electric fields and their side-effects, the dielectric losses. These cause a change in material temperature, which causes material parameters, like the dielectric loss factor, to change as well. In this case, the heating effect is desired (capacitive heaters).

Ex. 3: Capacitive Polymer Welding

Capacitive heating is an economic way to heat up materials with certain dielectric properties. This principle of heating can be applied locally with special electrodes and the heat production is controlled by changing the applied voltage and/or frequency. The simulation is based on the coupling of equations (3) and their boundary conditions:

$$\begin{aligned} &\nabla\left(\varepsilon_r(f)\nabla V\right)=0 \\ &\nabla\left(k(T)\nabla T\right)=-\omega\left(\varepsilon_r \tan\delta\right)(T,f)E^2 \end{aligned} \qquad (3)$$

with $\vec{E}=-\nabla V$.

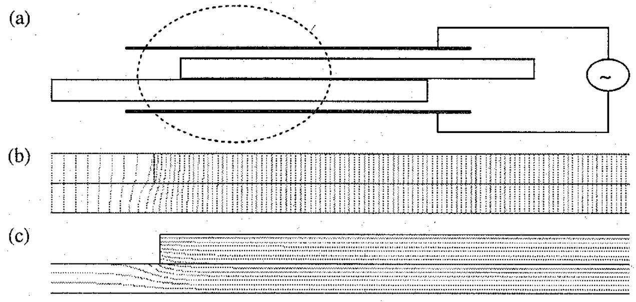

Fig. 5: (a) Outline of the heating installation, (b) electrical flux lines during welding polymers by capacitive heating, (c) isothermal lines

In this case, the coupling can be described in a one way direction, since their is no direct temperature feedback of the thermal equation to the electric potential equation. Hence, a faster convergence can be expected, but still a correct representation of the loss factor is necessary to obtain a fast convergence of the thermal equation.

In the example of Fig. 5, the welding of two thin polymer sheets is represented. In order to study the temperature distribution, which highly determines the quality of the resulting sheet material, a coupled computation is performed. In this way, the form, position and driving parameters of the electrodes can be optimized.

V. CONCLUSION

Object oriented programming techniques are well-suited for the implementation of material characteristics in different representations as required to perform a flexible coupled problems analysis. Features such as data encapsulation, virtual functions, overloaded operators and polymorphism can be used to enable the flexible and robust generation of programme code.

A data structure, which is implemented in C++ code is presented and its use is demonstrated by means of magneto-thermal and electro-thermal simulations.

ACKNOWLEDGMENT

The authors are grateful to the Belgian "Fonds voor Wetenschappelijk Onderzoek Vlaanderen" for its financial support of this work and the Belgian Ministry of Scientific Research for granting the IUAP No. P4/20 on Coupled Problems in Electromagnetic Systems. The research Council of the K.U.Leuven supports the basic numerical research.

REFERENCES

[1] P. Molfino, M. Repetto, "Comparison of Different Strategies for the Analysis of Non-linear Coupled Thermo-Magnetic Problems under Pulsed Conditions," *IEEE Trans. on Magnetics*, vol. 26, no. 2, pp. 559-562.

[2] K. Hameyer, U. Pahner, R. Belmans, H. Hedia, "Thermal computation of electrical machines,", *3rd international workshop on electric & Magnetic fields"*, Liège, Belgium, May 6-9, 1996, pp.61-66.

[3] C. Chaboudez, S. Clain, R. Glardon, D. Mari, J. Rappaz, M. Swierkosz, "Numerical Modeling in Induction Heating for Axisymmetric Geometries," *IEEE Trans. on Magnetics*, vol. 33, no. 1, pp. 739-745.

[4] Bj. Stroustrup, *The C++ Programming Language*, 2nd ed., Addison Wesley Publishing Company 1992.

[5] G. Booch, *Object Oriented Design*, Benjamin Cummings 1991.

[6] A. Arkkio, A. Niemenmaa, "Estimation of Losses in Cage Induction Motors using Finite Element Techniques," *Proc. of ICEM '92*, Manchester, UK, Sept. 15-17, 1992, pp.317-321.

[7] R.De Weerdt, K.Hameyer, R.Belmans, "End ring inductance of a squirrel-cage induction motor using 2D and 3D finite element methods," *30th IAS-IEEE Conf.*, Orlando, USA, Oct.8-12, 1995, Vol.I, pp.515-522.

[8] R.De Weerdt, K.Hameyer, R.Belmans, "Finite-element analysis of steady state behaviour of squirrel cage induction motors compared with measurements," *IEEE Conf. on Electromagnetic Field Computation (CEFC)*, Okayama, Japan, March, 18-20.1996, pp.220.