Virtual product development for electrical motors

M. Schöning and K. Hameyer
Institute of Electrical Machines, RWTH Aachen University
Schinkelstr. 4, 52056 Aachen, Germany
marc.schoening@iem.rwth-aachen.de

***Abstract*—In this paper, a software for virtual product development of electrical motors is introduced. Nowadays, consideration of electromagnetic, thermal and acoustic design aspects of electrical machines can be achieved by either using specialized or universal software packages. The specialized packages are predominantly developed for only one physical domains whereas universal software packages need to be adapted for the individual application. In both cases, the design engineer spends plenty of time, adapting or coupling software instead of developing an electrical machine. The software presented in this paper is specialized for the development of electrical machines and combines simulation tools for all mentioned physical domains. Furthermore, numerical and analytical methods are coupled to reduce computation time of typical numerical simulations and to increase the accuracy of analytical approaches. For the implementation, open-source components are used to reduce costs and to use well documented file-formats. Furthermore, the visualization component includes virtual reality capabilities. This specialized approach allows for a timesaving virtual design process of electrical machines.**

***Index Terms*—Electrical machines, Electrical drives, Virtual Reality**

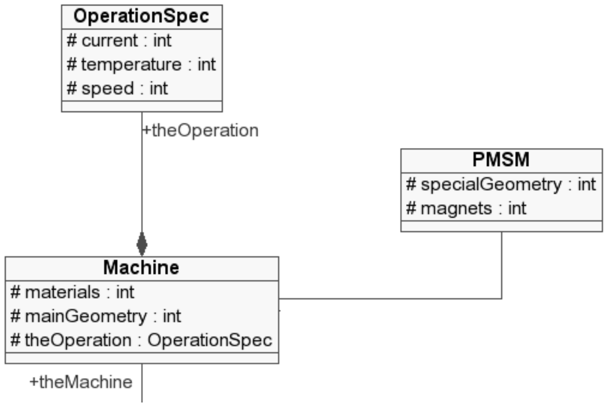

Figure 1. Implementation of the Machine.

I. Introduction

NOWADAYS, the development of electrical machines is more and more virtually accomplished. This means, that all necessary machine characteristics are determined by using various software tools. The first prototypes are built up in a late development state when the machine design is already determined. Their importance for validating simulation results decreases due to better software methodology. The virtual product development allows shorter time-to-markets and requires less prototypes compared to classical development methods. This leads to new demands for the product developer.

There are two ways to fulfill the requirements of a virtual product development process. On the one hand, specialized software packages e.g. [1], [2], [3], [4] can be used for the specific design aspects of electrical machines. Restrictions are, that most tools are developed only for one physical domain and they work only with either numerical or analytical methods. In this case, the design engineer has to implement the coupling of various software tools from different software-producers on his own, when regarding more then only one design aspect. This is difficult due to different file and data formats sometimes only readable by the specific software itself. On the other hand, there are universal software tools [5] available covering all physical domains. Because of their general design, they have to be adapted for the specific needs e.g. the design of electrical machines. With both methods, the design engineer spends plenty of time adapting or coupling software packages instead of designing a machine. Furthermore, high investment costs have to be payed for the software tools.

The software presented in this paper is specialized for the virtual product development of electrical machines. It regards all aspects of machine development including electromagnetic, thermal, structure-dynamic and acoustic characteristics. Furthermore, these design aspects can be computed analytically and numerically. Both methods are coupled to increase the accuracy and to reduce computational effort. This software allows for a fast design of electrical machines. The numerical core of the program is iMOOSE [6], all other computations are done by python [7] and the scientific python extension [8]. For the visualization of Finite-Element Solutions, VTK (Visualization Toolkit) [9] and VISTA [10] are used, the mesh generation is accomplished by Gmsh [11].

II. Implementation

The presented software is implemented, using modern object-oriented design aspects [12], [13]. The machine is introduced by three classes (see Fig. 1). The main class *Machine* stores the basic layout data of rotating machines. This includes the main geometry parameters as well as the stator layout, the winding dis-

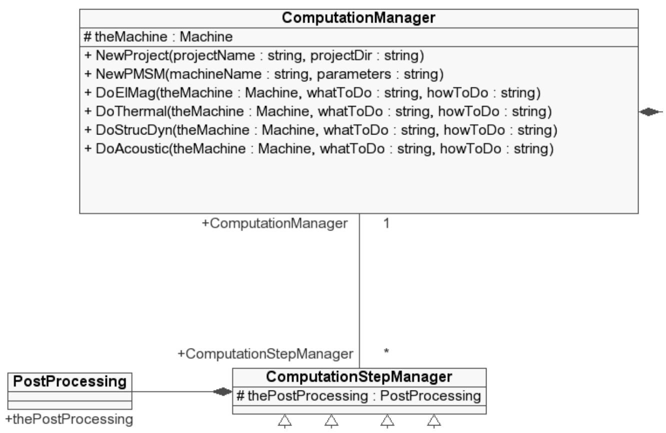

Figure 2. Main Manager Class.

tribution, the shape of the chassis and the material definitions. The specialized class *PMSM* extends the machine specifications for the needs of a permanent magnet synchronous machine. All custom parameters such as the magnet definition and the rotor shape are implemented in this class. Finally, the class machine is connected to the class *OperationSpec*, in which all definitions about the operating point are stored. These three classes define the machine completely.

In the next step, a main class controls all available computations and administrates computation results (see Fig. 2). It also forms the interface between the software and the graphical user interface. The *ComputationStepManager* class governs the physical domain, in which specialized inherited managers for each domain control the numerical and analytical methods. Furthermore, the *PostProcessing* class is linked to the managers to visualize the results. Due to the fact, that numerical methods are available for all physical domains, a base class *NumericalSimulation* is implemented and specialized for each domain later. These classes are connected to a *Mesh* and a *ProbDef* class, in which the Finite-Element or Boundary-Element Mesh is build and an adequate problem definition is generated. At the present development state, the software packages Gmsh [11] and ANSYS [5] are supported for mesh generation.

III. Coupling

The advantage of analytical methods is their low computational effort whereas numerical methods achieve a high accuracy. For studies of geometry variations, both methods can be coupled by parameterizing the analytical computation with a numerical simulation. This approach is shown for the electromagnetic computation exemplarily. As illustrated in Fig. 3, the machine parameters are extracted from the instance of the class *Machine*. After transferring the input data, the electromagnetic characteristics are computed analytically. When the computation process is finished, the Finite-Element model and the problem definition is generated automatically and the numerical simulation is started. Afterwards, the numerical calculated torque and induced voltage are compared to the previously determined analytical results. By means of an adjusting parameter, the accuracy of the analytical model is increased [14].

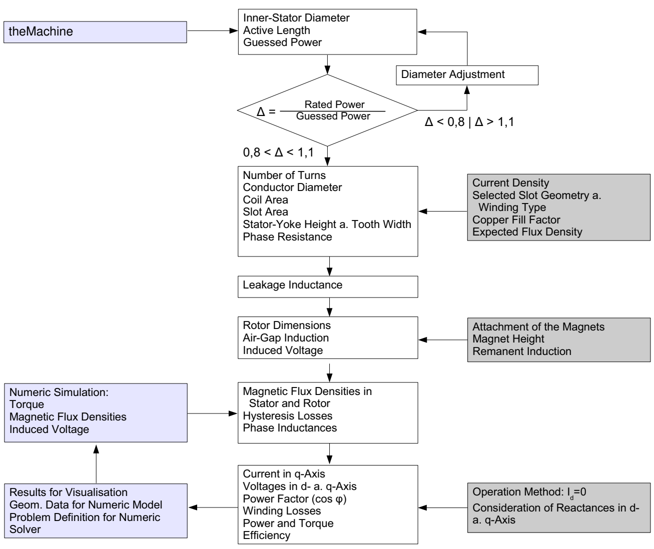

Figure 3. Electromagnetic Coupling.

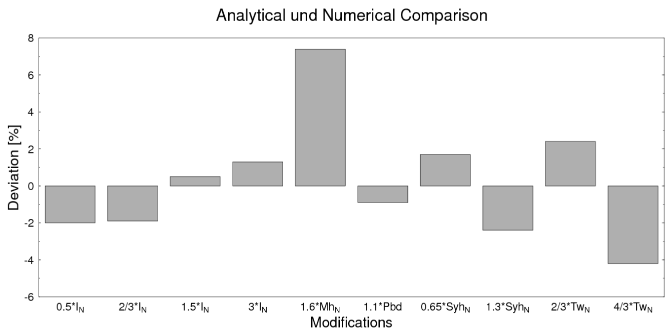

Figure 4. Comparing analytical and numerical Results.

To verify this procedure the analytical and numerical results are compared to the measurements of a known machine. For the machine in this study, the deviation between the numerical computed torque and the measured torque is less than 1%. Because the torque is an initial quantity for the analytical model, only the induced voltage and the geometry differs from the known machine. These deviations can be reduced to less than 1% by parameterizing through the numerical simulation. To reduce the overall computation time, the numerical simulation is only performed when the accuracy of the analytical model degrades significantly.

To evaluate the parameters, that affect the accuracy, the analytical model is parameterized once by the numerical simulation. Afterwards, various modifications are applied to the geometry, the magnets and the current. These modifications are calculated analytically as well as numerically and the results are compared. In Fig. 4, the deviation of the computed torque is plotted. Negative values indicate, that the analytical result is smaller compared to the numerically obtained result. The first modification halves the current in the stator coils. The analytical model calculates a 2% smaller torque then the numerical simulation. When adjusting the current to $\frac{2}{3}$ of the nominal value, the deviation is -1.8%. The difference increases up to +1.6% when the current is increased to three times of its nominal value. The maximum deviation is obtained when the magnet height is varied. Altering the magnet height Mh_N by 1.6 results in a difference of +7.65%. Varying the pole-pitch Pbd, the stator-yoke height Syh_N or the tooth width Tw_N yield a small deviation of less then 2.2%. Only changing the magnet height, which in fact varies the air gap too, leads to less accurate results and a new adjustment by a numerical simulation is recommended.

This study demonstrates that the once parameterized analytical model produces very accurate results, when calculating variations of machine designs. When combined with optimization algorithms, parameter studies can be performed very quickly compared to purely numerical methods, though achieving a high accuracy due to the initial parameterization.

IV. Data Structure

A continuous data structure is one of the most important parts of the presented software. Simplicity has to be combined with

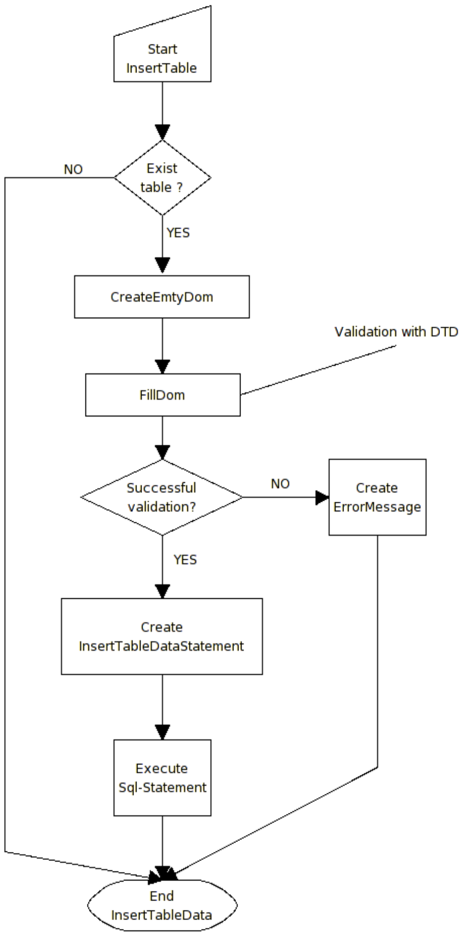

Figure 5. Validation.

robustness and integrity. Furthermore, a history function needs to be implemented, where all changes to the data are recorded, traceable and reversible. These major requirements need to be combined with a simple backup component, which exclude typical databases, where a system administrator must provide access and is responsible for backups. Beside this, a validation as well as dependency checks are required to prevent wrong data in- and outputs.

The mentioned requirements leads to the file based database SQLite [15]. It is a C library that implements a self-contained, embeddable, zero-configuration SQL database engine with a simple API, that can be saved like a standard file. SQlite can be combined with a powerful XML [16] data structure, where XML covers all aspects of data validation and dependency checks. This is done by using the standardized DOM (Document Object Models) and DTD (Document Type Definition) 5. The DOM is the interface to the XML data, allowing for an ordered reading and writing of data in the structure. The DTD regulates the validation of the data, by comparing the given value for a parameter with its properties, that must be defined once for each. The presented data structure achieves all necessary requirements in a simple and concise way.

V. Graphical User Interface

One main design aspect of the presented software is usability. The main features should be controllable by an easy to use graphical user interface. Various features have to be included when implementing the structure of the GUI where simplicity is the most important one. Another fundamental feature is the multi user mode, which allows multiple designers to work on one machine. This leads to the necessity of a history function in the data structure, already discussed before.

Fig. 6 illustrates the first dialog, where the user chooses his name and an appropriate workspace identifier. If the workspace is not available, a new one is created. With the given information, the graphical user interface is initialized (Fig. 7). On the left-hand side, a List Control is used to present the physical domains in a clearly arranged way. When switching between the physical domains, the already mentioned dependency check is used, to give information about possibly needed intermediate results, required to accomplish a specific computation. For example, when the force density on the stator teeth is requested, the flux density in the stator and the surrounding air must be known. If these intermediate results are not already calculated, the user is asked to perform a numerical electromagnetic simulation.

The right-hand side of the graphical user interface is used to display results. In Fig. 7, the cogging torque of an example machine is shown. The presented data on the right-hand side can be explored interactively, which means, that functions like

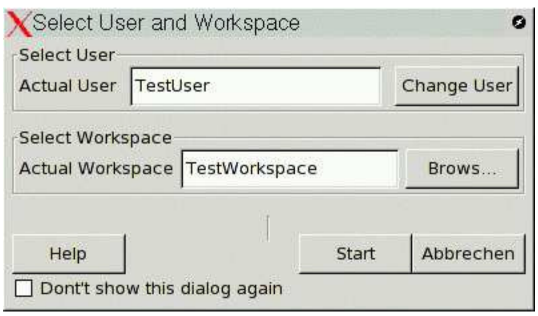

Figure 6. Start Dialog.

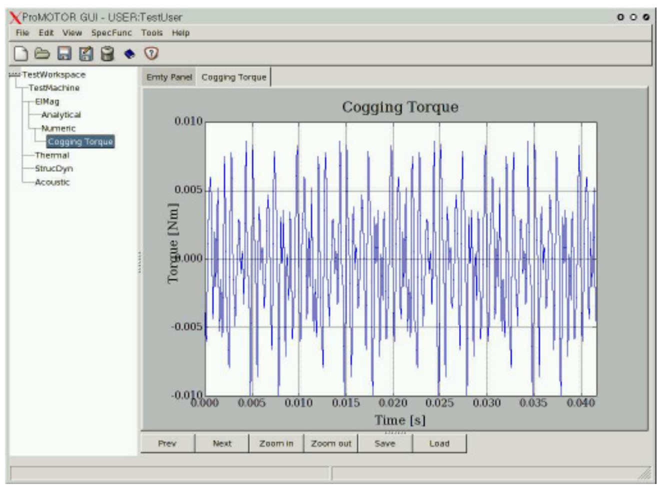

Figure 7. Main GUI Window.

Figure 8. Virtual Reality Example.

zooming and switching between plots are available. Pictures from the shown data can be created instantaneously.

VI. Virtual Reality

The desired simplicity of the software does not only affect the design of the graphical user interface but also the visualization of the computed results itself. By virtual reality techniques, large amount of computed data, normally produced when using the Finite-Element Method, can be explored efficiently and in detail. It allows making a correct interpretation of the results and taking the right decisions very quickly. Also it supports the idea of virtual product development, because the user can dive into a virtual representation of the machine concept in pre-prototype state.

This virtual reality feature is implemented using the open-source visualisation toolkit VTK [9] combined with VISTA [10]. First of all, the Finite-Element solution is converted to the VTK data structure. Thereby, all VTK filter algorithms can be applied to the numerical solution, which offers the possibility to create scalar and vectorial plots [17] with various additional features like visualizing field lines in air volumes. The created visualisation needs to be coupled to virtual reality hardware. This is achieved by the software library VISTA. VISTA covers all aspects of virtual reality technology. It is scalable in order to integrate all kinds of display systems, ranging from high-end visualisation displays like CAVE-style systems down to standard desktop monitors, and it offers access to a variety of 3D input devices. Special features like force-feedback and three dimensional acoustics are also covered. VISTA is implemented as a library, so that virtual reality applications can be developed rapidly for specific purposes such as the presented software. By using additional hardware, like a holobench or a cave, combined with additional features like force feedback system and head position tracking system, the realistic effect of virtual reality environments can be increased significantly. An example of a user in a cave style system is illustrated in Fig. 8. The virtual reality representation of the shown claw pole alternator was created with visualisation module of the presented software.

VII. Results and Conclusions

In this paper, a software is presented specialized for the virtual development of electrical machines allowing for a timesaving design process. This is achieved by combining analytical and numerical simulation methods for electromagnetic, thermal, structure-dynamic and acoustic aspects into one continuous software tool. No more coupling or specialization of different software packages is needed anymore. The simplicity of the program is realized by the graphical user interface and the data structure, controlling the design process. Using XML and SQLite for the data processing allows to prevent wrong user input by standardized validation techniques. Furthermore, the implemented history feature, which journals changes and makes them traceable and reversible, provides a multi user function, where multiple designers can work on one machine concept. The visualisation component based on VTK and VISTA allows for displaying the numerical solutions also in virtual reality environments. This simplifies the understanding of results and provides an adequate instrument for product presentations in pre-prototype state.

References

[1] Cedrat & Cedrat Technologies, "Flux2D / Flux3D," http://www.cedrat.com/software/flux/flux.htm [online], 2006.

[2] The Scottish Power Electronics and Electric Drives Consortium, "Speed software for electric machine calculations," http://www.speedlab.co.uk/index.html, [online], 2006.

[3] G. Lacombe, A. Foggia, Y. Marechal, X. Brunotte, and P. Wendling, "From general finite element simulation software to engineering-focused software," in *Electromagnetic Field Computation, 2006 12th Biennial IEEE Conference on*, 2006, pp. 226–226.

[4] Ansoft Corporate, "Maxwell 2d/3d," http://www.ansoft.com, [online], 2006.

[5] ANSYS INC., "Ansys inc. - open and flexible simulation software solutions for every phase of product design," http://www.ansys.com, [online], 2006.

[6] D. van Riesen, C. Monzel, C. Kaehler, C. Schlensok, and G. Henneberger, "iMOOSE - an open-source environment for finite-element calculations," *IEEE Transactions on Magnetics*, vol. 40, no. 2, pp. 1390–1393, 2004.

[7] G. van Rossum, "The python programming language," http://www.python.org/, [online], 2006.

[8] T. Oliphant, "Scipy - scientific tools for python," http://www.scipy.org/SciPy, [online], 2006.

[9] Will Schroeder, Ken Martin, and Bill Lorensen, *The Visualisation Toolkit, 3rd Edition*, Kitware, 2004.

[10] T. van Reimersdahl, "ViSTA: a multimodal, platform-independent vr-toolkit based on WTK, VTK and MPI," 2000, Fourth International Immersive Projection Technology Workshop.

[11] C. Geuzaine and J. F. Remacle, "Gmsh: a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities," http://www.geuz.org/gmsh/, Jan. 2006.

[12] Craig Larman, *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development*, Prentice Hall PTR, 2004.

[13] Erich Gamma, Richard Helm, Ralph Johnson, and John Vissides, *Design patterns : elements of reusable object-oriented software*, Addison Wesley, 1995.

[14] M. Schoening and K. Hameyer, "Coupling of analytical and numerical methods for the electromagnetic simulation of permanent magnet synchronous machines," *7th International Symposium on Electric and Magnetic Fields, EMF2006 Proceedings*, pp. 213–214, 2006.

[15] Chris Newman, *SQLite (Developer's Library)*, Sams, 2004.

[16] Eric Van Der Vlist, *XML Schema*, O'Reilly Media, 2002.

[17] M. Schoening, K. Hameyer, and D. Van Riesen, "Visualising finite element solutions in virtual reality environments," *Sixth International Conference on Computational Electromagnetics, CEM 2006 Proceedings*, pp. 97–98, 2006.