

A SYSTEM ARCHITECTURE FOR MODULES SUPPORTING VARIATIONAL DESIGN IN ELECTRICAL ENGINEERING

Mr B Dettlaff, TCS GmbH
Professor Dr D Roller, Mr D Schäfer
University of Stuttgart, Germany

99ME003

0 Abstract

In today's first and second generation CAD systems for electrical engineering, variational design is supported only in a manual manner, i.e. by copying, modifying and deleting special parts of a design. Next generation E-CAD systems shall support variational design in a fully automatic manner. This can be achieved by using parametric and constrained-based modeling together with configuration and version management tools.

This paper presents the overview of an architecture for E-CAD variant modules to be integrated and employed in next generation E-CAD systems. We discuss the parts needed to build up such variant modules as well as the relationships between one part and another. Furthermore, we take into account some aspects of communications management, active knowledge handling and last but not least the graphical user interface. Finally we close with some remarks concerning CORBA, the CAD reference model, and software development in general.

1 Introduction

In these days the world wide CAD/CAM community is trying to define their challenges for the next millennium. Relating to computer aided design for the domain of electrical engineering the term „ECAD System“ is more and more replaced by the term „Electrical Engineering Solution“ (EES) [1]. Without spending any time on defining that term, everyone being familiar with CAD/CAM will grasp the right meaning, of course. A rather special part of such an EES will be a module for supporting variational design in order to reduce design time and costs, simultaneously with an increase of design quality. In other words, variational design seems to be a suitable solution for reusing existing constructions in an efficient manner [2, 3]. As mentioned before, variational design is based on parametric and constrained based modeling - technologies, that are well known and highly developed in the domain of mechanical engineering, today [8]. However, variational design for electrical engineering has to be distinguished carefully from that employed in mechanical engineering, because both domains are very different in their basic understanding [5]. In electrical engineering the essential kind of information is that of so-called logical information which are laying behind the graphics, whereas variational design in MCAD in the main is oriented at geometries and dimensions.

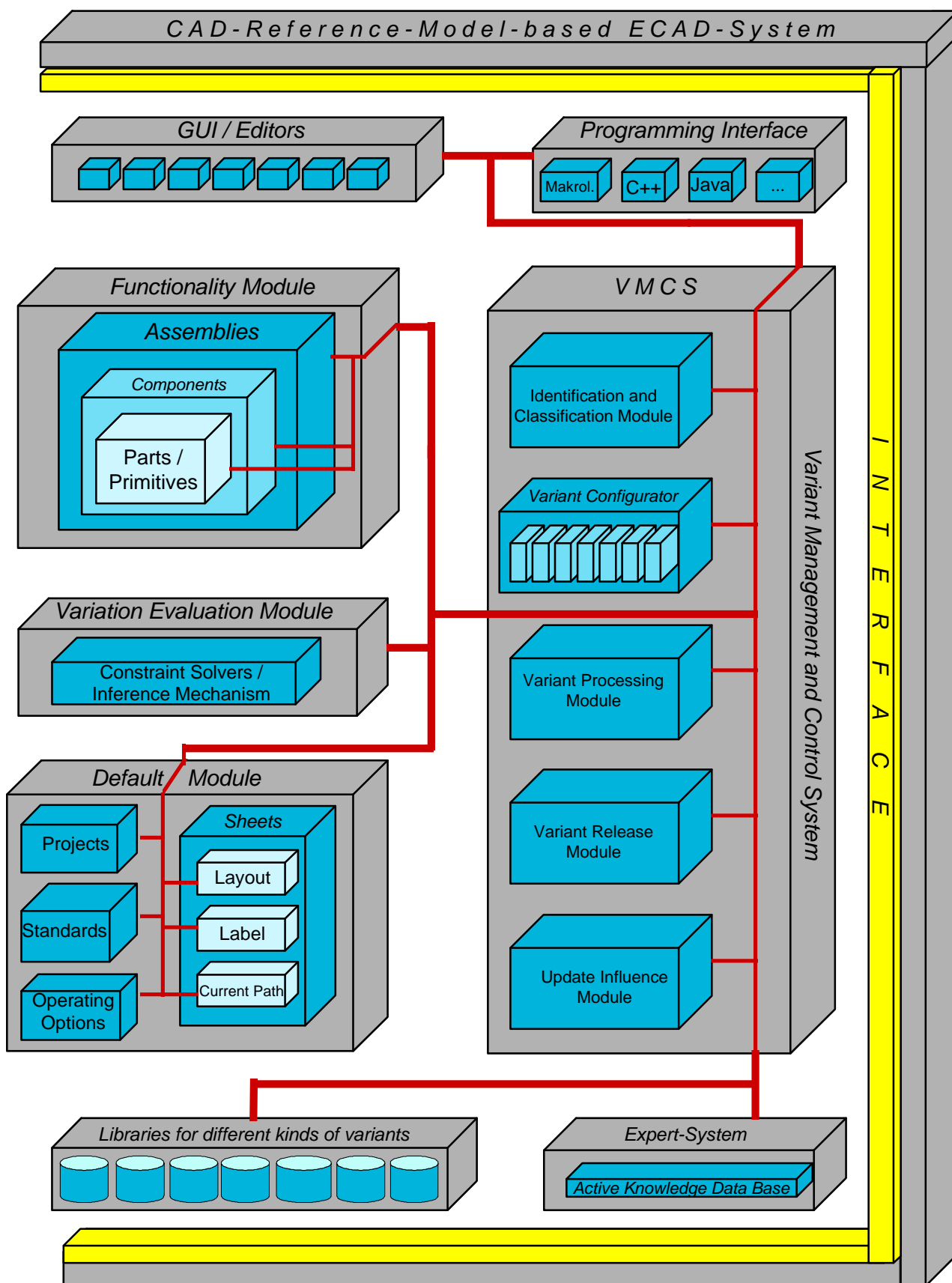


Figure 1: Overview of an Architecture for ECAD Variant Modules

Hence, in electrical engineering, variational design relates more to modifications of logical data or technical operating options that have to be carried out automatically than it has to deal with modifications of geometry.

In order to extend contemporary variational design techniques to EES specific variant modules, we have drawn up a comprehensive requirement analysis over a period of about one year. Based on the results of that analysis, we suggest a system architecture for future ECAD variant modules, that is briefly summarized in the following chapters. We have split the whole architecture into several modules. Nine top level modules, each of them internally consisting of submodules, make up the complete variant module. An overview of this architecture can be found in figure 1.

2 GUI and API

Obviously, a Graphical User Interface (GUI) for interactive variant specific operations is to be foreseen in our architecture. This GUI should be integrated into the ECAD systems GUI and not be a separate one, of course. The variant specific GUI mainly consists of several variant editors, one for each kind of variant (the classification of variants is not subject of this paper, cf. [4]) to be supported by the variant module. Within these specific editors, all the data relevant to the respective type of variant can be displayed and edited. The editors can be activated either by selecting the desired one from a menu bar (edit mode) or in context sensitive manner by double-clicking a specific part of the design (design mode). - The different modes mentioned before are considered later.

Usually, these editors consist of two or three different views of the relating data. All the editors usually have a so-called tree view that can be considered as a kind of file manager for the related type of variant or the corresponding variant library, respectively. Within these tree views, one can browse the variant libraries to select a desired variant category or instance. A second kind of view that also appears in each variant editor is a so-called attribute edit view. Herein, the attribute values of the design patterns parameters being currently activated can be displayed and modified. Some kinds of variants, e.g. part variants, also contain a graphical view. This graphics view can be used to display or modify symbols, or even to create new symbols for design parts.

Besides interactive variational design operations and in addition to automatic variant evaluation to be considered later, it should also be possible to describe variants in the underlying ECAD systems macro language. Furthermore, programming interfaces for higher languages like C/C++, JAVA and so on should be accessible to allow the integration of existing programs into system specific macros as well as the employment of internet technologies for using the WWW as a virtual design environment for practicing concurrent engineering in the sense of an interdisciplinary global engineering process (IGEP).

3 Functionality Module

The functionality module and its submodules contains all the functionalities and responsibilities which are necessary to realize the technological design aspects of variational design. This mainly concerns the generation, modification and management of parts, components and assemblies as well as variants of those. A rather complicated thing to take care of within the functionality module is to manage the interplay and the mutual dependencies of the parts, components and assemblies. Concretely, the functionality module consists of three submodules: part module, component module and assembly module. These three submodules are nested one into another. From an abstract point of view, each part variant is a trivial component variant and each component variant is a trivial assembly variant. Strictly speaking, one could consider even a single part as a most trivial assembly.

From a communicational point of view, all the functionality modules are responded by a variant management and control system, that from now on will be abbreviated as VMCS. As a basic principle, there is absolutely no direct communication between two different modules of the system.

Any communication is handled by the VMCS. Now let's have a closer look to the functionality modules three submodules:

3.1 Part Module

Within the functionality module the part module is of rather special interest. It is used to create part variants, to modify them and to do the corresponding data and file management. For this purpose, a special part manager has to be included within the VMCS. Also, a corresponding part editor should be available from the GUI. In principle, the part module consists of the actual data model, the part editor, and the corresponding part manager. All the processes concerning this, are activated and controlled by the VMCS and its submodules. Hereby, especially the relation to the variant processing module, the variant release module and the update influence module (we will discuss them in chapter 4) is to be emphasized.

In the edit mode, part or part variants, respectively, can be loaded, modified, deleted and stored without influencing the actual design (only the part library level is concerned). In design mode however, the above mentioned activities directly relate to the construction currently being under revision.

In principal, the system offers a certain number of standardized and predefined parts being available to the user within a special part library. These parts represent design patterns, that can be selected and used from the user via the part manager at any time. For this purpose, we should have a special part editor consisting of the following three components: a kind of file manager for part variants, a graphical preview of the symbol, as well as a table showing all of the parts attributes and the corresponding actual parameter values. The part specific file manager allows to scroll, search, load, delete, and store parts from/to the part library. The graphical preview shows the graphical symbol that represents the part within the ladder diagram. Modifications of these symbols are generally not allowed at this stage because there might occur inconsistencies to existing constructions and projects. However, if such modifications are desired, the user needs to have a special authorization to carry them out. The table mentioned before is used to display all the attribute values of the part variants as well as to adjust them according to the designer intend. At this point, one has to take into account that parts can have a lot of referenced attributes which parameter values may arise from attributes of various other parts. The system also has to monitor in which components and assemblies the parts are embedded. Imagine the following situation: A special part has been modified somewhere in the design and this modification is desired to influence the entire construction or project. In this case, all the other instances of this part and all the components and assemblies using this part, have either to be adjusted automatically, or at least shown to the user to allow manual, dialog-based acceptance.

3.2 Component and Assembly Module

The general functionalities of the component module mainly come up to those of the part module. A component specific editor and manager are to be foreseen as before. A rather special demand of the component module is to provide significant selection methods for defining new components within a ladder diagram, that explicitly can be stored as independent components in the component library. To realize this, support of the VMCS is needed again. Another responsibility of the component module appears in relation to the default modules operating options module to be considered later. In cooperation with these both modules it is possible to activate or deactivate components to separate them from the construction in early design phases or testing phases. Especially the management of dependencies to other modules is a quite complicated thing to be considered within the context of component variants. For this purpose, the related variant managers have to interact with each other intensively. Imagine the following situation: A user wants to modify a special part that belongs to a

component he has selected before. Now, the part editor has to be launched from within the component editor to allow the desired modification.

The assembly module allows to edit and manage assemblies very similar to the way stated above for components, just one abstraction level higher. The assembly module manager can be considered as an intelligent configuration tool. It can be used to select assemblies from the assembly library or from other constructions to integrate them into other projects or even to create a new draft project in this way.

4 Default Module

The default module is responsible for all properties and functionalities concerning the outward appearance of a construction, including its whole documentation. It is divided into four submodules: the project module, the standards module, the operating options module and the sheets module.

4.1 Project Module

The project module is needed to adjust and modify the global properties and parameter values which are valid within an entire construction or project as well as to do the corresponding data management. For example, default values for the documentation language, the underlying norms, standard variants for sheet layout, labeling, denotation of parts and current path, cross references, etc. can be set in this way. All these settings can be done using a special project editor that interacts via the corresponding project manager with the related libraries. So to speak, a project variant reflects some kind of default settings of all the other kinds of variants belonging to the default module. Whenever project parameters have been modified, the variation evaluation process has to be started to retrieve the project according to the current settings.

4.2 Standards Module

The standards module allows to adjust a construction or project in respect to parameters concerning technological and documentational properties with project wide validity. Such properties usually refer to national norms and standards or enterprise wide arrangements. Typically, a construction is based on some kinds of norms and standards. Under scope are mainly national standards, e.g. DIN or JIC, as well as enterprise wide arrangements, e.g. graphical symbols for design parts. These kinds of information are brought into the system by so-called standard variants. Predefined variants or patterns of such variants are stored in a special library. Whenever such a predefined standard variant has been modified, it can be stored as a new variant using a new, unique ID. New kinds of standard variants are usually required whenever authorities have changed special norms or standards or when they have been extended to new aspects. Equivalently to the other modules, the standards module in detail consists of the underlying data model, a special manager, editor and library. As mentioned earlier, all the communication between the different modules is controlled by the VMCS. As we will see later on, the standard variants are close cooperating with the project variants.

4.3 Operating Options Module

The operating options module is responsible for controlling the technological operating options of an installation to be designed or its subsystems, respectively. The module consists of the operating option variants data model, a specific editor for modification operations and visualization as well as a corresponding manager to enable the general management functionalities. Now let's have a closer look at the operating options modules sphere of activity:

The operating options module directly influences the assembly module, the component module and the part module. Furthermore, the operating options settings can be transferred from the assembly module to the component module and from the component module to the part module or the corresponding variants, respectively. It is also possible to influence the functionality modules submodules via the project module in regard to operating option variants.

Let us close for the moment with some remarks about the dependencies and connections within the default module as well as the communication flow:

The standards module is the most important submodule of the default module. It influences the sheet module, the operating options module and the project module. The intervention here takes place for example like this: operating options, layouts etc. can be derived from national standards and transferred as default values to a project variant. Since all settings can be achieved in a manual manner within the project module, the project module is also directly influenced by the operating options and sheet module. The sheet module consists of three submodules to be discussed in the following section.

4.4 Sheet Module

To adjust the project wide documentation layout, that means the layout of each sheet, a special sheet module has to be realized. The responsibilities of this module can be transferred to three specific submodules concerning the basic layout, the labeling and the representation of current paths. The corresponding variants can be created, modified and represented using the features of those modules. The different kinds of variants being available within the sheet module can either be propagated as global valid parameters (default values) to a project variant or adjusted and modified in a local manner (for a single sheet). The structure of the sheet module equals that of the other modules (data model, manager, editor, library).

5 Variant Management and Control System (VMCS)

The variant management and control system (VMCS) is responsible for managing and controlling all kinds of data flow within the entire ECAD variant module. This especially refers to the communication among the different modules and submodules as well as to the administration of the various kinds of dependencies that can exist between one module and another (constraint management). The VMCS controls the communication with the user, accesses many libraries, and initiates the variant evaluation process whenever valid modifications of parameter values have been carried out. Another important task for the VMCS is to detect automatically the different types of variants, when the user selects one somewhere in the design editor. Furthermore, if the type of a special variant has been detected, the corresponding variant manager has to be launched.

As mentioned earlier, there are two different processing modes within the ECAD variant module. A variant can be edited either on a database level or on a design level. The different variant processing modes to be considered in detail later on have to be taken into account whenever a construction is to be updated after parameter values have been changed. The entire VMCS consists of five single modules together with the actual control system. The structure and the responsibilities of these modules are subject of the following sections.

5.1 Identification and Classification Module

At the beginning of this section, let's have a look at a short extraction of a simplified scenario: A user wants to change some parameter values of a part within the design mode. To do this, he might position the mouse pointer somewhere over the related part and double click a mouse button. At that

moment, the system has to detect automatically that a part variant has been selected. After that, the part manager automatically has to launch the part editor. For all the other kinds of variants, this process has to take place analogously. To realize this, a special identification and classification module for variants is to be foreseen. This module processes a users action like this: At first, the module identifies the selected type of variant. Then the responsibility for the further processing is transferred to the corresponding variant manager.

5.2 Configuration Module

After the type of a selected variant has been identified as described above, the further responsibility is transferred to the configuration module. This module contains a rather special variant manager for each kind of variant. These variant managers usually have the following responsibilities:

- General management operations
Load, Store, Delete, Edit, Search, Copy, Cut, Paste, etc. as far as these operations are relevant for the according type of variant.
- Access to libraries / data transfer
The variant manager gets access to the relevant libraries, controls the data flow between the underlying data model, the libraries and the corresponding editors. If, for instance, a special part would be changed within a component variant, the variant managers of both, part variants and component variants, would have to be activated.
- Updating the data model
After the variant manager has read the relevant data from the library, these data are transferred to the actual data model of the related type of variant.

Updating the editors

After the data model has been updated, the same has to be done with the corresponding variant editor. Within the editor, the different views are filled according to the data stored in the data model.

5.3 Variant Processing Module

The variant processing can usually be carried out in two different modes. Let us first consider the so-called edit mode, that exclusively effects the database level. Predefined variants can be loaded from the database, modified, and saved using a new file name. Hereby, the modifications are mainly related to adjusting parameter values, adding new parameters, deleting parameters, and modifications of graphical symbols. Before a new variant that has been created in edit mode can be stored to the library, its correct functionality has to be checked by a design engineer and finally released with an electronic signature. For this purpose, also a special module has to be implemented within the VMCS.

To avoid inconsistencies, it is not allowed to modify existing predefined variants and to save them using the original name or ID, respectively. Modifications that have been carried out in edit mode are of maximum global influence. That means, they are valid for all projects in which the concerned parts are used.

The second processing mode is the so-called design mode. This mode is active whenever modifications or other operations are carried out within the graphical design area, that means, whenever a construction actually is under revision. Operations carried out in design mode directly influence the construction being under revision as well as the corresponding documentation. Usually, these local modifications must not influence the related libraries, because for global modifications the edit mode would be the right choice.

5.4 Variant Release Module

As mentioned in a previously, some security checks have to be met before a new variant can be added to the corresponding library. If, for example, a new component variant has been designed, the first thing to do is to check its correct functionality and interface. For this reason, a new variant to be added to a library has to be checked by a responsible design engineer who finally can release the variant by some kind of electronic signature. For this purpose a special variant release module that controls the checking and release mechanisms has to be realized within the VMCS. Possibly, a special functionality for managing and monitoring different levels of user rights should be integrated into this module, too.

5.5 Update Influence Module

Modifications that are carried out in design mode can have a different scope of action. If a part that can be found several times in a construction is modified, there are three possible ways for updating that construction:

The first way is to modify only that part, that has been explicitly selected. In this case we are talking about a so-called local sphere of action. In addition to a local sphere of action, also a global sphere of action can be useful. In this case, one has to distinguish two different ways:

If all instances of the selected and modified part that can be found in the whole construction are modified automatically, one speaks of an automatic global sphere of action. If, however, the user is prompted to accept or reject the modification for all the other parts, this is called a manual global sphere of action.

A similar problem occurs in the following situation: Supposed that a part variant or an operating option variant is modified in design mode, probably many other parts, components or assemblies can implicitly be effected by that modification. Hence, the system needs the information, whether that modification relates only to the selected part (local sphere of action) or if the entire construction has to be updated in global or manual sphere of action. This update influence information then has to be passed on to the variant evaluation module to carry out the modifications in the desired manner.

Now let's make some remarks concerning the most important part of the VMCS, its internal communication system. The whole internal communication within the VMCS as well as the external information exchange between all the other modules of the entire ECAD variant Module is handled by this communication system. Its functionality is based on the way that is described in the CAD reference model [6, 7].

Whenever parameter values have been changed somewhere in the parametric based data model, the construction being under revision - in other words the new design variant - has to be reevaluated. Depending on the kind of the changed parameters, the sphere of action for retrieval can be very different. For example, a new national standard variant requires the whole construction to be redesigned, whereas, for instance, a modification of a single part may influence only a very small sphere of action. To carry out the retrieval process, a special module with the required responsibilities must be implemented. This module mainly consists of a so-called constraint solver and sophisticated mechanisms for maintaining inference chains. In this context we have to mention, that it would be helpful to employ an expert system because of the quite complicated constraint modeling. In addition to that, we suggest to employ an active knowledge data base for representing active knowledge and for constraint propagation, together with several libraries, one for each type of variant.

6 Conclusions

In this paper we have presented an overview of a system architecture for modules supporting variational design in electrical engineering. Since variational design has become a CAD base technology in recent years, such modules will be employed in next generation ECAD systems. Our suggestion has been based upon a comprehensive requirement analysis, we did during the last year. For the time being a prototypic implementation of such an ECAD variant module is under way.

7 References

- [1] Roller, D., Achilles, A., Richert, U., Schmich, M., Verhaag, E.: Kosten- und Zeitreduzierung in Elektrokonstruktion und Anlagenprojektierung, expert-Verlag, 1998, ISBN 3-8169-1616-3
- [2] Roller, D.; Schäfer, D.; Richert, U.: Variantentechnologie für Elektrokonstruktion und Anlagenprojektierung, CAD-CAM REPORT, Dressler-Verlag, No. 9/98, pp. 90-97
- [3] Roller, D., Dettlaff, B., Richert, U.: Rationalization in the Electrical Engineering Process, in: D. Roller (ed): Mechatronics - Advanced Development Methods and Systems for Automotive Products, ISATA Proceedings, Automotive Automation Ltd., Croydon, England, 1996, pp. 279-284
- [4] Roller, D.; Schäfer, D.: Variational Design in Electrical Engineering - An Extension of Parametric Modeling, to appear 1999 in: Proceedings of Seminar on CAD Tools and Algorithms for Product Design, Wadern, Germany
- [5] Mink, U.; Roller, D.: New ECAD System Technology, in D. Roller; P. Brunet (eds), CAD System Development - Tools and Methods, Springer Verlag, 1996, pp. 41-54
- [6] Roller, D.; Dettlaff, B.: Realisierung einer modernen ECAD Systemarchitektur in Anlehnung an das CAD-Referenzmodell, in: Ruland, D. (ed) Tagungsband CAD 96, Verteilte und intelligente CAD Systeme, DFKI GmbH, Kaiserslautern, 1996, pp. 41-54
- [7] Abeln, O.: CAD-Referenz-Modell, Teubner-Verlag, 1995, ISBN 3-519-06356-5
- [8] Roller, D.: CAD - Effiziente Anpassungs- und Variantenkonstruktion, Springer-Verlag, 1995, ISBN 3-540-58779-9