INTERNATIONAL CONFERENCE ON ENGINEERING DESIGN
ICED 03 STOCKHOLM, AUGUST 19-21, 2003.

EXTENDED CAD MODEL

Nenad Bojcetic, Mario Storga

Abstract

The main objective of this work is to develop a program system to support the process of creating a computer model of the product that will be able to bound Feature Based Design (FBD) CAD model of the product with the knowledge used in its creation. The proposed solution is the Extended CAD model of the product containing the geometric representation of the product model and the knowledge required for its creation.

Keywords: design knowledge, FBD, engineering design, computer-based design

1. Introduction

Most CAD systems are primarily used for creating a computer representation of a real model of the product as well as for creating technical documentation. Unfortunately, CAD systems are focused especially on the problems of handling the graphic representation of the product model with a limited support for the interaction with the knowledge bases.

During the process of development and improvement of CAD systems, the emphasis is mainly put on techniques and technologies for the improvement of representation and manipulation of graphic models. In the most engineering domains, especially in the field of mechanical design, geometric representation is not the only form of information about the product that concerns a designer. The designer has to consider the information about the used materials, constraints of the available technologies, transport limitations, exploitation of the product and etc.

In addition, we are witnesses of the increased complexity of products and more rapid product development resulting in an enormous increase in the amount of knowledge one must deal with. In order to overcome these difficulties some firms are focused on more efficient usage of internal and external knowledge, reusability of existing components and design solutions, collaborative design and usage of artificial intelligence methods [1].

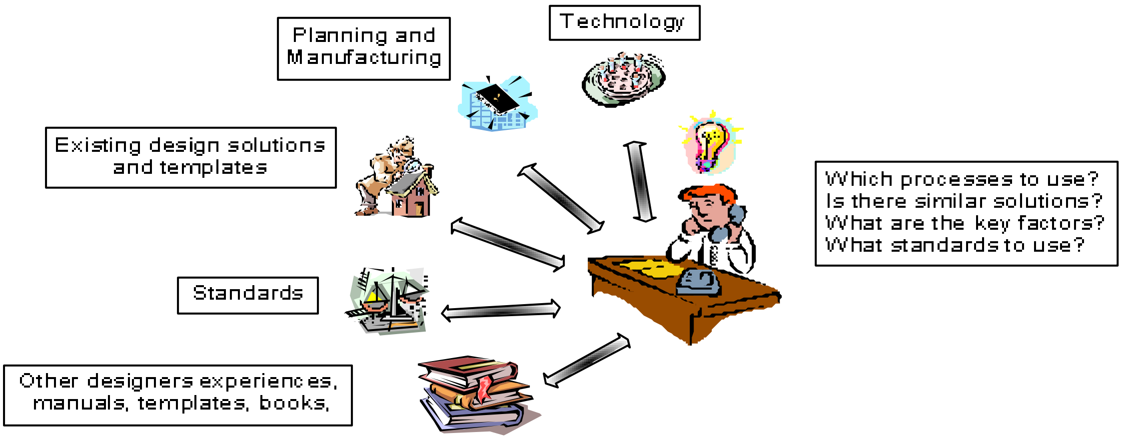
One way to help designers to overcome problems emerging during the design process is to bind the knowledge used and created during design and the geometric representation of the product. The issues pointed out above are the basis in the development of the Extended CAD model (XCAD).

The main goal which has to be fulfilled by the Extended CAD model is to integrate the knowledge used and generated during the product design with the feature-based CAD model of the product.

Design knowledge, system structure and system objects will also be briefly explained. For additional information, please contact the author.

2. Design knowledge

The nature of the design process and the complexity and variety of the knowledge, used during process of product creation (Figure 1), requires a flexible and robust model for the representation and handling of the knowledge [2]. One of important factors in the design process is the process of knowledge gathering and processing. The decisions a designer makes during the design process and the used design solutions depend on the knowledge available to the designer at the time.



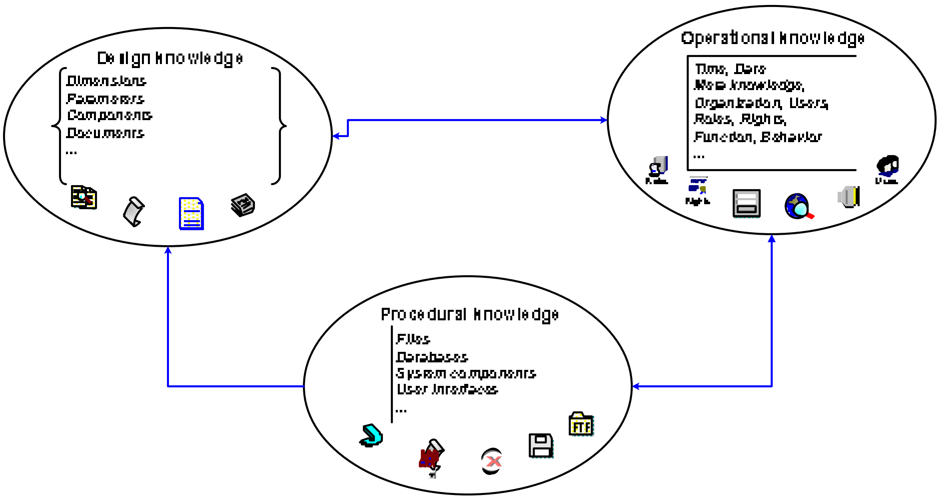


Figure 2. The design knowledge structure

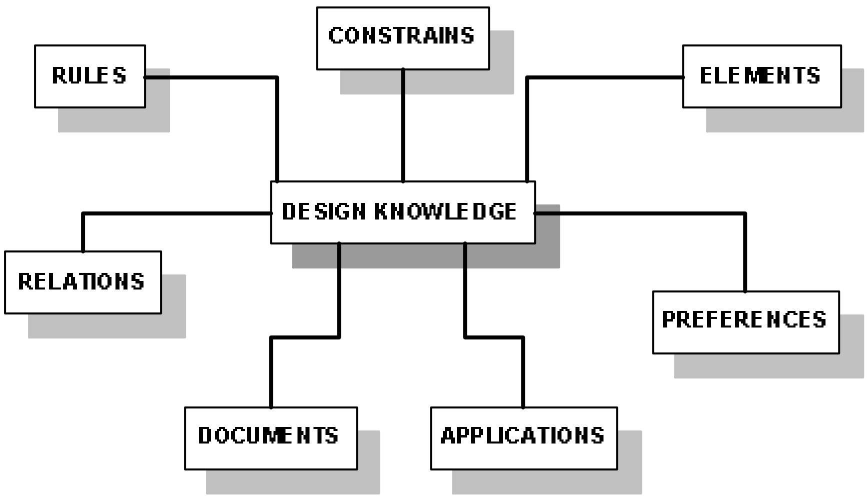
Design knowledge represents a knowledge generated or used during design of the product. This type of knowledge is bound to the subject of design and it contains the information relevant to the process of design. Information can be divided, depending on its structure and format, as into: geometric information (geometric representation of the model of the product in most cases the CAD model), information about the documents used during the design process (standards, manuals, and recommendations), information about inference rules and external program applications (calculations, simulations …).

Operational knowledge is an extension of the design knowledge and it contains the information about users, security, time stamps, etc. Maybe the most important feature of the operational knowledge is that it describes the knowledge about the knowledge contained in the system, or, in other words it stores the meta knowledge information. The meta knowledge describes the design knowledge contained in the system, and gives the information on how to use this knowledge, where it is stored, in what format, what is its structure, who used it last, when, for what purpose, etc.

Procedural knowledge relates to the process of management and manipulation of the system data and also to the information about external data sources.

Design knowledge is further divided into knowledge items (Figure 3) that are considered to be the carriers of certain types of knowledge. Knowledge items are as follows:

- Elements (defines the version of the product or product part),
- Relations (defines the sequence of the applied constraints in a particular version),
- Documents (defines the documents relevant for the product),
- Applications (contains the information about required external applications),
- Preferences (defines the material properties, units, and parameter boundaries),
- Constraints (defines the conditions that must be met in order to have a valid model),
- Rules (defines an instance of the product).



The elements in above described item and subitem definition represents: <id> - identification number, <si> - symbolic name, <nk> - component name, <nr> - relation type, <no> - operation name, <np> - preference symbolic name, <td> - document type, <sz> - component status, <iok> - component identification mark, <nvpa> - external application name, <pv> - default value, <sqlcs> - SQL connection string, <vk> - component connection, <od> - document format, <docid> - document id, <ndp> - parameter name, <to> - operation type, <ioto> - operation type id, <path> - external document path, <sqlu> - SQL query string, <minv> - minimal value, <maxv> - maximal value, <vn> - discrete value set, <sj> - unit system, <zkus> - component feature in the assembly, <zkup> - component feature in the part. Most detailed description and explanation of the item and subitem elements can be found in [5]. Examples of knowledge items are shown in Table 1.

Table 1. Example of knowledge items

item	<id>	<si> | <nk>	<nr> | <no> | …	<iok>	<subitem>	<vk> | …	<docid>
Element	1	Hole	**active nonactive**	7	=;;cut id 7	**unbound, bound**	2
Relation	1		**map**	2	d0 = part3;;d48		3
					d0 = part3 ; ; d48		
					d0 = **execute** ; 1 ; 8, 30, 40		
					d0 = **minmax** ; 2 ; 20		
					d0 = **discrete** ; 3 ; 5		
					d0 = ; ; 34		
					d0 = ; ; ?		
			intrel		d28 = d13 / 2		
Application	1	Excel	**cp, sql, run**	excel.exe	=;;D:\home\one.xls		4
Document	1	Standard A	**html, doc, txt**	ie.exe	=;;D:\home\two.html	**digital, physical**	
Preference	1	Boundaries A	**minmax**	10	=;;(20, 30)		6
		Value B	**discrete**	5	=;;(2, 5, 8, 10)		
		C.0361	**material**		=;;(ro, E, ni, G)		
		ANSI	**units**		=;;(ANSI)		
Constraint	1	part11	**mate, align, insert, orient, exist**		=;;(DTM1, DTM3)		10

Rules are defined and structured similar to traditional imperative programming language. During the process of instance definition user writes (through graphic user interface) the rules (see Figure 4) using the following language elements:

- Command blocks:
  - **GLOBAL, END GLOBAL** – defines the set of commands that will be executed for every selected instance,
  - **FOR version, END FOR** – defines set of command that are specific for particular instance.
- Conditions:
  - **IF condition, THEN, ELSE, END IF** – defines set of command that will be executed if certain conditions are met,
  - **CONSTRAIN** – defines constrains for particular instance.
- Commands:
  - **MATERIJAL** – defines material used in particular instance,
  - **DOKUMENT** – defines documents that will be shown to the user during process of instance creation,
  - **UNITS** – defines active unit system,

```
GLOBAL
  UNITS 1
  DOCUMENT 3
END GLOBAL
FOR version 1
  CHECK 1
  ELEMENT 2
  IF CONSTRAINT 3 THEN
      RELATION 1
      DOCUMENT 1
      DOCUMENT 2
      MATERIAL 1
  ELSE
      RELATION 2
      DOCUMENT 1
  END IF
END FOR

FOR version 2
  CONSTRAINT 2
  ELEMENT 3
  IF d20 > 9.0 THEN
      RELATION 2
      DOCUMENT 1
  END IF
  IF CONSTRAINT 9 THEN
      IF part18:d34 == 35 THEN
          RELATION 1
          MATERIAL 2
      ELSE
          RELATION 2
      END IF
  END IF
END FOR
```

Figure 4. Example of the Rules code

- **RELATION** – defines relation that will be executed for particular instance,
- **ELEMENT** – defines components and features which constitutes an instance.

The structure of the knowledge of the Extended CAD model complies with STEP ISO 10303 product structure [6]. Actually, the knowledge is defined and structured in the same way as the product (Knowledge as the Product) (Figure 5) and can be attached to the product definition in the same way as the definition of the product document (Document as the Product).

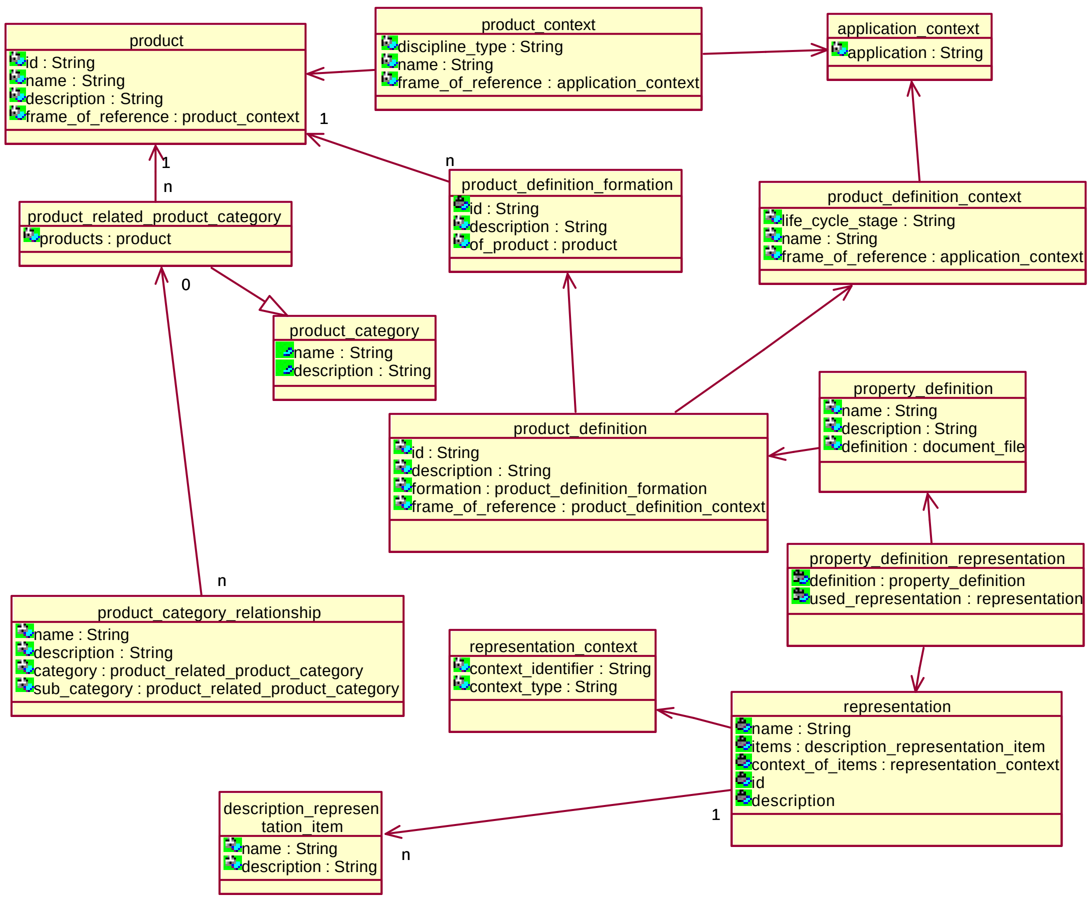


Figure 5. STEP like design knowledge structure

Implementation of the Extended CAD model is carried out by using the developed program system presented in Figure 6.

The developed system is consists of the following modules:

- Core (system management and point of control),
- Database interface module (establishing and maintenance of connection between the system and the database),
- User interface module (manages user actions, and creates and initializes graphic user interfaces),
- Knowledge management module (manages the creation, usage, change and disposal of the knowledge (design and operational) handled through the system),
- Auxiliary modules (set of modules for interaction with the operating system, network, etc.).

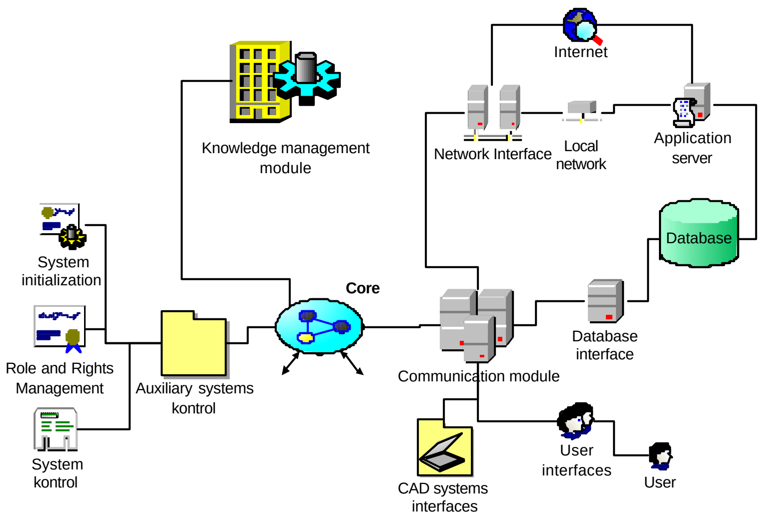


Figure 6. Extended CAD System structure

The system is created on the basis of the defined class structure (Figure 7) of the Extended CAD model objects.

Extended CAD model has been implemented in the Java programming language and tested in the Pro/Engineer environment. The system was linked to the CAD modeler using J.Link libraries.

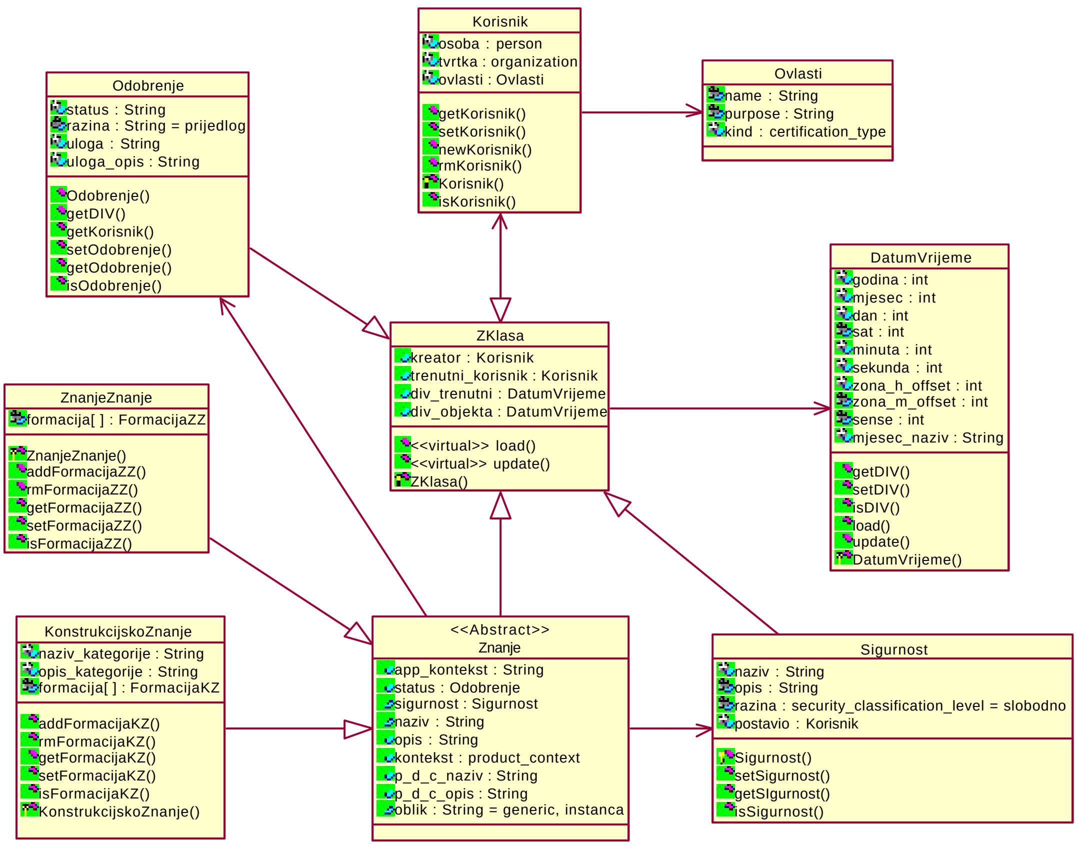


Figure 7. UML representation of the system objects structure for the Extended CAD model

3. Test example

For experimental verification, a simple CAD model was built. It consists of four components with a possibility of creating of about six different versions depending on the activated features (Figure 8).

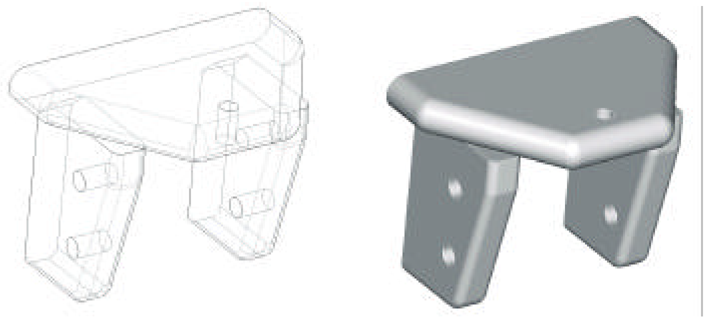


Figure 8. Test model example

Through system interfaces, the relevant parameters, dimensions and features (that define a particular component version) are defined and initialized. Using a defined set of programming instructions, we defined several different versions of the product. Sequence of screenshots, shown in Figure 9, depicts the process of creation of XCAD part.

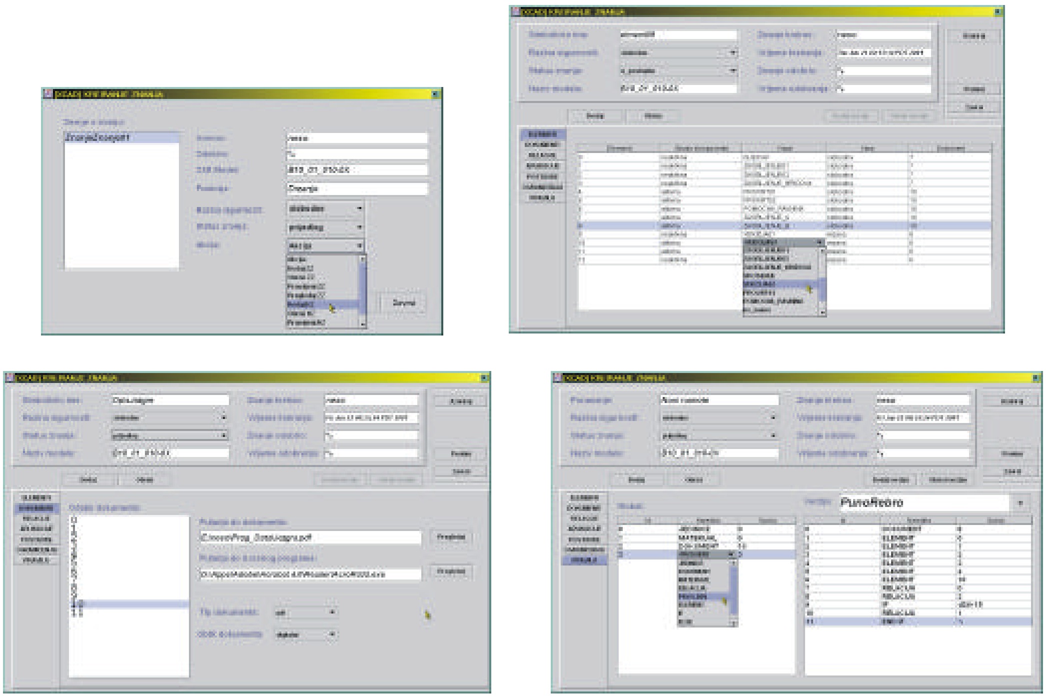


Figure 9. Screenshots during the process of creation of the Extended CAD Model

Four assemblies were created in Pro/Engineer using various versions of Extended CAD models. During the creation of the product model, the user was able to choose between different versions of the Extended CAD model.

The supporting computer system actively controls the regeneration of the model and its validity. The applied knowledge can be viewed or modified at any time (Figure 11), on-line by Pro/Engineer using the support system or off-line by MS Access.

The created Extended CAD Model part is inserted several times using different configuration in the assembly. The result is shown in Figure 10.

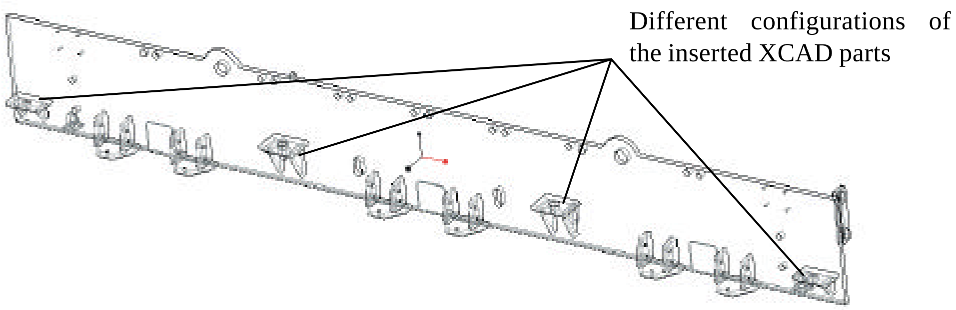


Figure 10. Assembly with inserted XCAD parts

During the process of insertion of the chosen XCAD part in the assembly the user can control the values and boundaries of the part parameters (Figure 11).

Figure 11. Screenshots of the system windows during the insertion of the XCAD part (upper two pictures) and the dialog boxes for the control of the parts parameters (lower five pictures)

4. Conclusion

Extended CAD model is an extension of the parametric based CAD model in the way that knowledge items are added to the parametric CAD model as features. The realized Extended CAD model exhibits a quasi intelligent behavior represented through the following properties:

- recognition of the existing status of the Extended CAD model,
- interaction with the user during the creation and usage of the Extended CAD model,
- maintenance of the validity of the CAD model,
- invocation of the use case scenario if the conditions of the model are fulfilled.

The design knowledge in the Extended CAD model is composed of the following components:

- Features that are of importance for particular component version.
- Relations on parameters that can be changed by the user in the boundaries imposed by used Preference knowledge item.
- Any Document that is relevant to the particular component version.
- Applications from which the values of parameters can be obtained.
- Available Material specifications for that could be used for the particular component creation.
- Rules that describe knowledge items used to make up the particular component version.

The realized Extended CAD model can be used as the basis for the creation of a more advanced knowledge supporting CAD systems. The next step will be to implement the learning capabilities into the computer system, so the Extended CAD model can learn from the user and expand the existing knowledge.

References

[1] M. Bangemann, Europe and Global Information Society – Recommendations for the European Council, Brussels, 36. May 1994.

[2] W. C. Regli, Digital Library Support for Engineering Design and Manufacturing, Proceedings of the 1999 ASME Design Engineering Technical Conferences, 12. – 15. September 1999., Las Vegas, Nevada

[3] M. Jovic, N. Bojcetic, Development of the Mechanical Assembly Description Model, Proceeding of th 5th International Design Conference Design 98, Dubrovnik, 1998.

[4] D. Marjanovic, N. Bojcetic, D. Dekovic, N. Pavkovic, D. Pavlic, M. Storga, D. Zezelj, Design Department Data Flow Integration, Integrated Product Development Proceedings 3rd International Workshop, Magdeburg, 2000.

[5] Moj doktorat

[6] J. Flowler, STEP for Data Management, Exchange & Sharing, Technology Appraisals, Twickenham, 1995.

Nenad Bojcetic, PhD, Faculty of Mechanical Engineering and Naval Architecture, Ivana Lucica 5, 10 000 Zagreb, Croatia, Tel: +38516168121, Fax: +38516156940, E-mail: nenad.bojcetic@fsb.hr, URL: www.cadlab.fsb.hr

Mario Storga, MSc, Tel: +38516168119, E-mail: mario.storga@fsb.hr