Nenad Bojčetić
Dragan Žeželj
Mario Štorga

ISSN 1333 - 1124

A TOOL FOR SUPPORTING THE PROCESS OF PROPERTY MANAGEMENT AND THE CREATION OF TECHNICAL DRAWINGS

UDK 744:004.4

Summary

CAD applications play a major role in everyday work of most designers. Whether designer creates a 3D computer representation of the product, technical drawings or a CNC code, CAD applications like CATIA® or ProENGINEER® are most likely to be used. Due to the fact that most CAD applications lend a limited support for the process of property management and the creation of technical drawings, there is a need for a tool to better support both processes. The APM (Application for Property Management) introduces an innovative and new approach based on the XML description of the model property and drawing information required for the creation and manipulation of CATIA Product and Part model properties and the automatic creation of the bill of material in technical drawing. Also, the APM connects a model and its drawing (creation and change of drawing head and drawing part list table) in such a way that data synchronization is available in both directions.

Key words: *technical drawings, CAD tools, CATIA, XML, design knowledge*

1. Introduction

In the daily work designers spend a significant period of time in the process of CAD model preparation (property management) and creation of the bill of materials in technical drawings. Whether the designer creates a 3D computer representation of the product, technical drawings or a CNC code, CAD applications such as CATIA® or ProENGINEER® are most likely to be used. Due to the fact that most CAD applications offer a limited support for these activities, there is a need for a tool to better support both processes. It is a well known fact that the usage of CAD applications can reduce the product time to market and have other benefits, such as error reduction, model reusability, etc. Therefore, usage of CAD applications offers endless possibilities to a designer. The designer can easily change design by testing different variants, can perform a finite element analysis, simulate various working environments, create assembling instructions, and so on.

But the usage of CAD applications does have some disadvantages [1]. As the amount of required information increases significantly [2], the designer has to spend considerable time on administrative tasks such as adding attributes to models in order to describe the customer's or company's specific information or to create a Bill of Material (BOM). A possible approach to these difficulties is to use Product Data Management (PDM) or Product Lifecycle

Management (PLM) applications or solutions [3]. The usage of these applications can create order in piles of documents and a huge amount of data related to the product. PDM or PLM applications or solutions are something that eventually has to be considered even in a small business. But the PDM implementation entails another set of issues. The implementation of a PDM solution is a time-consuming effort that involves every aspect of the company business, which may include adaptation of design processes, education of the users, installation of the computer hardware and software and dealing with legacy data.

Properties of a CAD model can be seen as a part of the design knowledge [4]. Properties hold vital information about the model and can be created either by the user or by the CAD system. Seen as a part of the design knowledge, properties data and their handing should be given special consideration. This is due to the fact that a designer must rely on the information stored in properties to make an informed decision about the model. Some pieces of information that are significant to the designer are, for example: Who approved the design or the model? Who drew it? What material is used? Are there any special considerations for the product handling or manufacturing?, etc.

The most demanding tasks for a designer, not in their complexity but in the amount of time required to perform them, are the processes of creating technical drawings and the creating BOM. Both processes are more or less supported by any CAD application, but every organization has its own practices and rules how to do them. In CATIA®, the process of the creating BOM and the drawing head may be a problem since the designer has to draw them line by line and to add each text object and write data from the model itself. The associative relationship between model attributes and BOM fields or head text objects does not exist; therefore, every change in the model also requires the updating of the drawing.

In this paper, we present the Application for Property Management (APM) that addresses the problems mentioned before and that can be located somewhere between the CAD application and the PDM. The proposed solution can support the process of technical drawing creation in such a way that BOM and drawing head data can be created automatically, and at the same time maintaining the associative relations with the model data. The APM can be seen as an interim solution towards the full PDM implementation. The APM is built on the basis of the eXtensible Markup Language (XML) [5] description of the CAD model properties and the elements of the technical drawing. A prototype is created for CATIA® but the APM can also be implemented in any CAD application that supports the Application Programming Interfaces (API).

2. Data model

The proposed application is planned to be used of the in various CAD applications, regardless of the application principle, native operating system or implementation. That is why the application data model is described in the XML, a widely used general purpose specification for data description. It provides a basic syntax that can be used to share information between different applications without the need to pass through many layers of conversion.

Main objects in any CAD Feature Based Design (FBD) [6] application are an assembly (or a product), a part and a drawing. All of these have associated different sets of attributes or properties. One set of attributes is defined and handled by the CAD application while an user can have only limited capability to alter its values, such as mass, instance name, density, volume, part number, etc., or cannot make any direct changes at all. The other set of attributes is defined by the user, in which case the user has complete control over them. These attributes vary from those related to the design and design process to those related to administrative purposes, such as the designer's name, data about the customer, version, designer's remarks, etc. Managing all these attributes is a tedious job, but an important one. Some external

applications rely on attributes and their correct values in order to run without problems; therefore, the user must pay special attention when dealing with attributes. The designer has to have the capability to define a different set of attributes for a product, a part or a drawing because each of them can hold or require different data. Some attributes are preferably selected from a predefined set of values e.g. division name, standard, surface finish, treatment, etc. In such case the designer does not need to waste time in searching for an allowable value. This approach also reduces typing errors. Unfortunately, in CAD applications, attributes are often defined as single value variables and there is no possibility of choosing values from a previously defined set. Based on analyses of attribute creation and manipulation in different CAD applications, a data model for describing attributes is defined and depicted in Figure 1. An short example is shown in Listing 1.

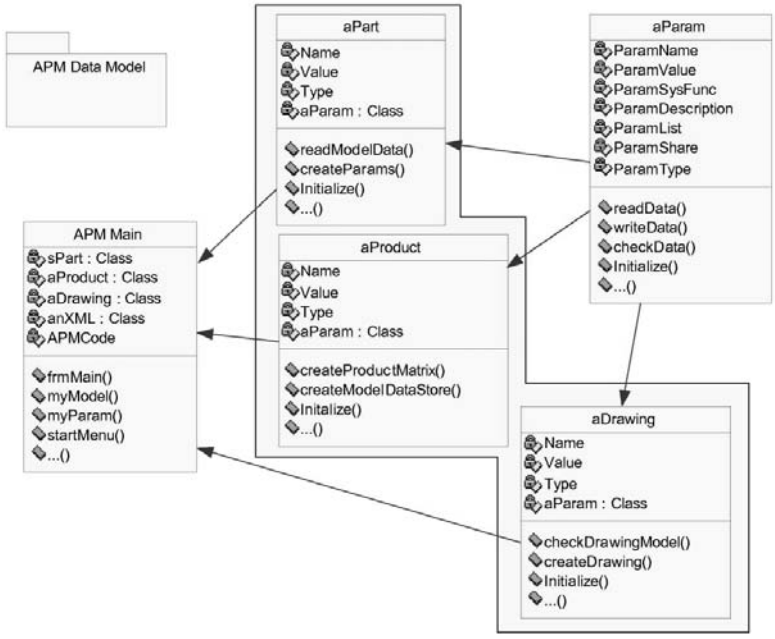

Fig. 1 APM Data Model

```
<Part>
     <Param Name="Bounding box height">
        <Description>Height of bouding box</Description>
        <Value>100.0</Value>
        <SysFunc>SysCode</SysFunc>
        <Type>DOUBLE</Type>
        <DrwName>drwBBH</DrwName>
     </Param>
</Part>
<Product>
     <Param Name="Designer name">
        <Description>Designer name</Description>
        <Value>Some Name</Value>
        <Type>STRING</Type>
        <DrwName>drwDesignerName</DrwName>
        <List>ListUser</List>
        <Share>Yes</Share>
     </Param>
</Product>
<Drawing>
     <Param Name="Scale">
        <Description>Parameter description</Description>
        <Value>1:1</Value>
        <SysFunc>SysScaleISO</SysFunc>
        <Type>STRING</Type>
        <DrwName>drwScale</DrwName>
     </Param>
</Drawing>
```

Listing 1 Example of XML description of part and product properties

An attribute is defined by its name and the value, but extra information is required for the CAD model description and for a proper creation of a technical drawing. Examples of additional information are links to the drawing entity, a predefined value, the information on whether a value is entered by the user or whether it depends on the application and model status, the attribute propagation, etc.

Looking from the point of view of a technical drawing, main elements that are required for a proper design description are a drawing head, which usually contains data about the model used as a basis for the drawing creation, and the BOM table if the model is a product or assembly. Of course, these are not all the elements used in the creation of a technical drawing, but these are the ones that are in the scope of this work. A data model for the description of elements of a technical drawing is shown in Figure 2.

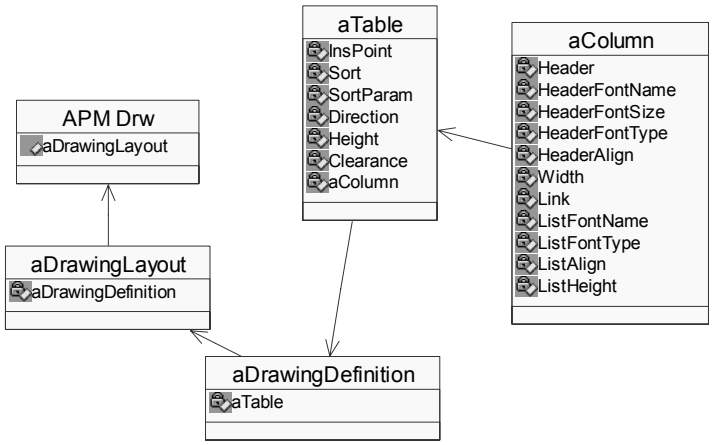

Fig. 2 Technical drawing Data Model

The XML description of a part of the data model is shown in Listing 1. A part of the XML file that is shown in Listing 2 is related to the data required for the creation of the BOM table. In the *DrawingLayout* section, the first part describes available BOM configurations. These configurations are then described in more detail, concluding with the actual table description. The user can define the sorting of table elements, the position and orientation of the table and, for each column, the data for a visual representation of the column and the link to the attribute whose value will be show in the table.

```
<DrawingLayout>
    <DrawingTypes>
       <Type>Part drawing</Type>
       <Type>Part list</Type>
       <Type>First level</Type>
       <Type>Standard parts</Type>
    </DrawingTypes>
    <DrawingDefinition Type="Part drawing ">
       <Table>None</Table>
    </DrawingDefinition>
    <DrawingDefinition Type=" Part list ">
       <Table>PartList,PartList</Table>
    </DrawingDefinition>
    <PartList>
       <InsPoint>InsPointT2.point</InsPoint>
       <Sort>asc</Sort>
       <SortParam>Position</SortParam>
       <Direction>Up</Direction>
       <Height>6</Height>
       <Clearence>30</Clearence>
       <Column id="1">
              <Header>Poz.</Header>
              <HeaderFontName>Swiss</HeaderFontName>
              <HeaderFontSize>2,4</HeaderFontSize>
              <HeaderFontType>Bold</HeaderFontType>
```

```
                <HeaderAlign>Center</HeaderAlign>
                <HeaderHeight>5,0</HeaderHeight>
                <Width>12,0</Width>
                <Link>Pozicija</Link>
                <ListFontName> Swiss </ListFontName>
                <ListFontSize>2,4</ListFontSize>
                <ListFontType>Normal</ListFontType>
                <ListAlign>Center</ListAlign>
                <ListHeight>5,5</ListHeight>
        </Column>
    </PartList>
</DrawingLayout>
```

Listing 2 Example of XML description of elements of BOM table

The XML file has no other restrictions than for the data format, so the user can tailor the BOM table and attributes according to the needs. Some examples of different configurations are as follows: a description of the first level elements of a product, a description of part specifications in which all parts regardless of their position in the model tree are shown, a description of standard parts, a description of parts that are made of a particular material or that have a certain attribute value, and so on.

As an addition to the XML structure, another section is added at the beginning of the file to describe departments in the organization. In that way, one XML file can hold the attribute and drawing description for different departments or purposes and can give the user more flexibility in creating an XML description that would satisfy particular needs.

3. APM architecture

Based on the described data model, the APM application architecture is defined. Application classes are shown in Figure 3.

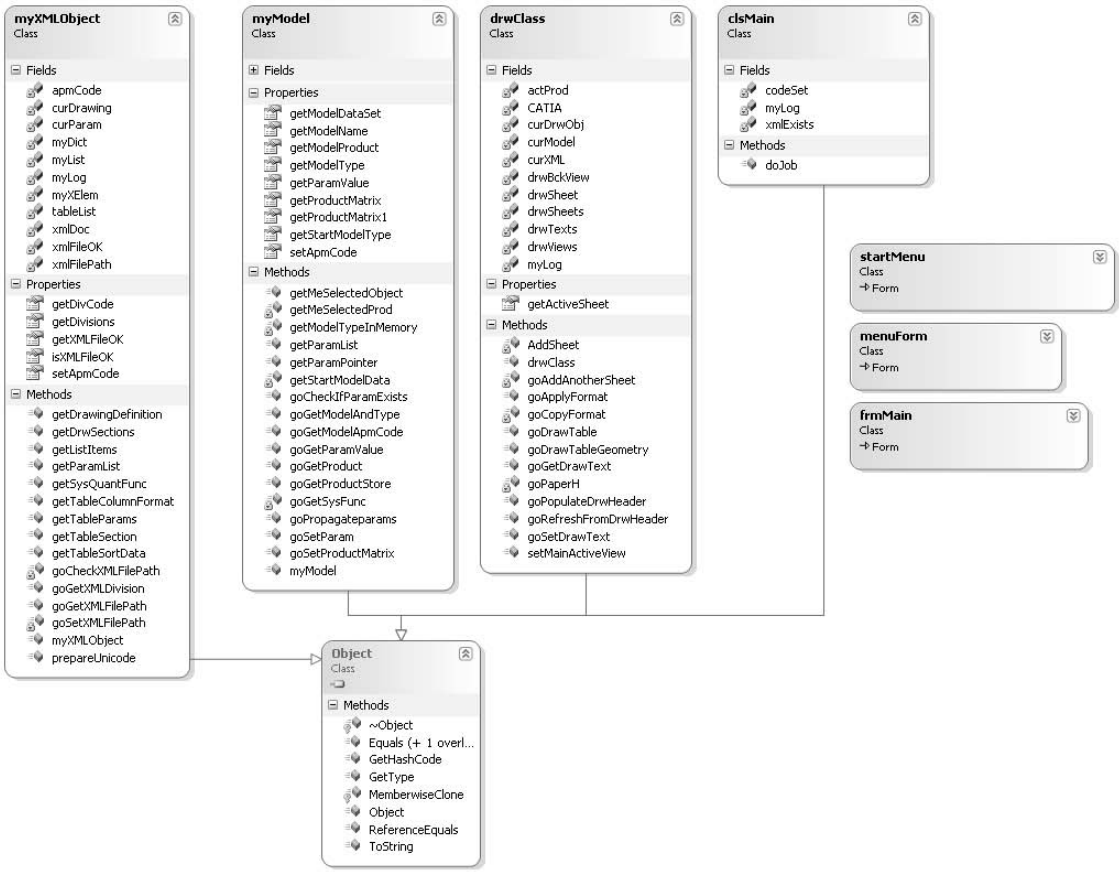

Fig. 3 Part of application class diagram

The APM application is created as a Dynamic Link Library (DLL) in Microsoft Visual Studio® using C# as a programming language. CATIA® enables interaction with the CAD engine, model and drawing elements through the API. Since the CATIA® API® is COM compliant, it can be used in almost any programming language under Microsoft Windows® (Visual Basic 6®, Visual Basic.Net®, C# Net, C++ Net, Java, etc.). The application can be started from the CATIA® using a macro script shown in Listing 3.

A special consideration is given to the code optimization for speed. Also, an extensive testing was conducted before the DLL approach was selected. As noticed during testing, the stand alone application created in any language is slower in execution because of the fact that it need to connect to or create a CATIA® session. The CATIA® own script language (VBAutomation a Visual Basic for Applications compliant) was rejected because of modest implementation of data containers and a limited set of elements for the creation of a Graphical User Interface (GUI). One of the problems during the realization of the application was very poor documentation and a lack of good examples on how to accomplish API method calls using C#. The export of model attributes is achieved by using the Microsoft Excel® interface. The format and the description of data used for exporting are also defined by using the XML. The user has the ability to describe what attributes he/she wants to export. An example of the part of the XML file used for the description of data for Excel is shown in Listing 3.

```
<Sheet Name="Part list">
        <StartRow>7</StartRow>
        <StartColumn>1</StartColumn>
        <ReportType>PartList</ReportType>
        <Param>
                <ParamName>Product Name</ParamName>
                <ParamRow>1</ParamRow>
                <ParamColumn>3</ParamColumn>
        </Param>
</Sheet>
```

Listing 3 Example of XML file for Excel data export

4. Test example

The application was tested on several models, both single parts and complex products or assemblies. After the script activation from CATIA®, the main window shown in Figure 4 opens and the user can select a command to accomplish the desired action. The first action is to define attributes or properties for the model. Because of the fact that the prototype application is created in Croatia, the Croatian language is used in pictures, but the localization to any language is possible without the loss of functionality.

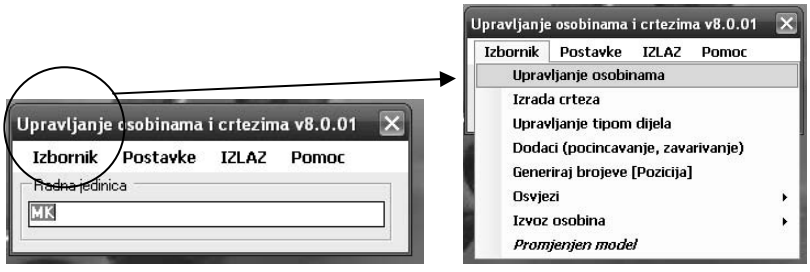

Fig. 4 Application main window and main menu list

A window for setting attribute values is shown in Figure 5. As it could be seen, some attributes can be chosen directly from the list of values. Additional values can be input to the list and if required the list of elements can be edited.

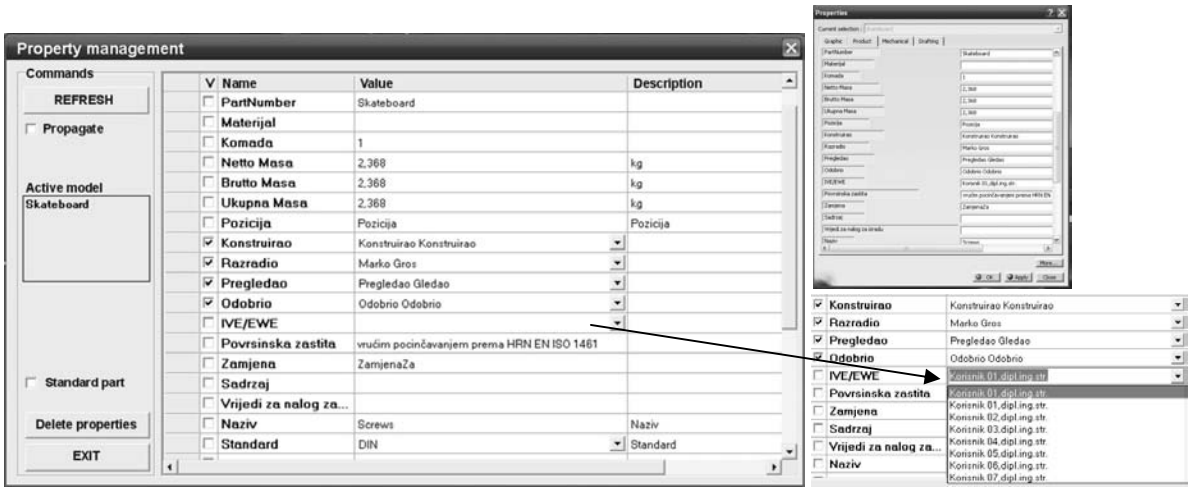

Fig. 5 Property managing window

The property window from the CATIA® is shown in the upper right corner of Figure 5,. This is the result of the action *Refresh* in the property management window. The designer may select option *Propagate* that will automatically enter values for selected properties for all the parts and products down the tree of active product, as seen in column V in Figure 5. For example, a senior designer has to approve all the models of the product and the product itself. Instead of entering his name under the *Approved* property for every part and product, he can achieve the same by entering the name once and by selecting this property to be propagated down the tree.

If the designer wants to create a technical drawing, a dialog as the one presented in Figure 6 will be shown, enabling the designer to select a desired table or set of tables. Elements in the list are templates from the specified folder, but the designer can select a different one by selecting *Select additional template*.

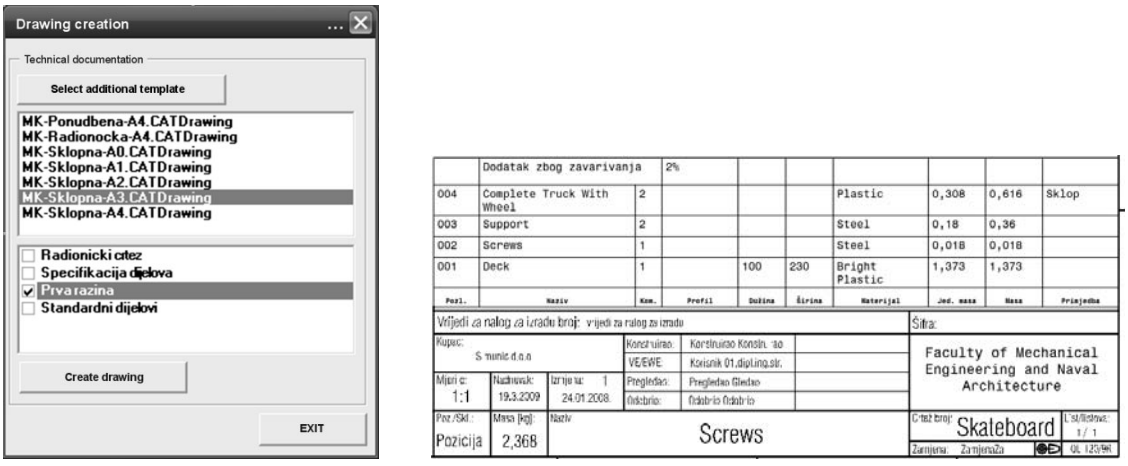

Fig. 6 Drawing table selection dialog and the final table in drawing

The application has additional functionalities or tools like these shown in Figure 7. The one on the left shows a dialog for setting the part or product source (standard or bought) and the one on the right can generate numbers for parts and/or products in various flavours.

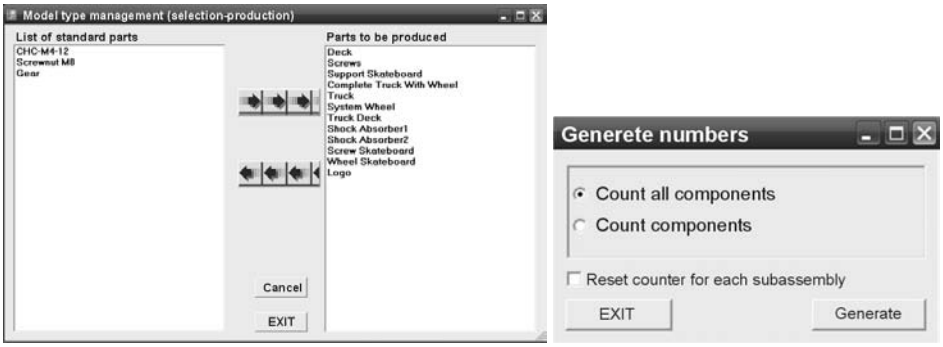

Fig. 7 Standard elements handling and generate numbering tool

Synchronization between the model and the drawing can be singled out as the most valuable application functionality. The designer can, at any time, activate synchronization from the model to the drawing or from the drawing to the model, so that the data is correct in both documents. The authors hope that Dassault Systems® will add action triggers, so that the user does not have to explicitly start a script to accomplish some actions.

5. Conclusion

The APM application is conceived and created to satisfy the needs articulated by the users of the CATIA® application and other designers that deal with technical drawing and CAD model properties on daily basis. During the testing of the application, potential users found the concept very appealing and easy to use. The most impressive result of the application testing is a general consent of the designers that the time required to create properties and a technical drawing was significantly reduced as well as the number of errors. In addition, technical drawings produced in this way have a more unified look.

Future developments would include linking the application to the database and branching to Pro/Engineer and AutoCAD. Additionally, extensive testing is under way in a real work environment. This will further improve the application to meet the designer's needs regarding work in creation of technical documentation.

REFERENCES

[1] James M. Morgan, Jeffrey K. Liker: The Toyota Product Development System, Productivity Press, New York, 2006, ISBN 1-56327-282-2

[2] Steven F. Bridge: Intelligent Representation of Geometric Knowledge for CAD, Intelligent CAD Systems II, Springer-Verlag, London, 1989, p.p. 275-291, ISBN 3-540-50914-3

[3] N. Bojcetic, Computer Model of Design Knowledge, Disertation, 2001, UDK: 658.512.2:681.3:621

[4] John Stark: Product Lifecycle Management, Springer-Verlag, London, 2005, ISBN 1-852-33810-5

[5] Elliotte R. Harold, The XML Bible, Wiley Publishing, New York, 1999, ISBN-13 978-0764532368

[6] Jami J. Shah, Martti Mantyla: Parametric and Feture-Based CAD/CAM, Wiley Publishing, New York, 1995, ISBN 0-471-00214-3

Submitted: 03.3.2009

Accepted: 05.6.2009

Nenad Bojčetić
nenad.bojcetic@fsb.hr
Dragan Žeželj
Mario Štorga
University of Zagreb
Faculty of Mechanical Engineering and
Naval Architecture