10th International Research/Expert Conference
"Trends in the Development of Machinery and Associated Technology"
TMT 2006, Barcelona-Lloret de Mar, Spain, 11-15 September, 2006

APPLICATION OF UML FOR OBJECT ORIENTED MODELLING OF CUSTOMIZED PRODUCT FAMILIES

Prof. Dr. Ilia Boyadjiev, Ass. Prof. Dr. Dimitar Bojkov, Ass. Prof. Angel Bachvarov
Faculty for German Engineering and Management Education (FDIBA)
Technical University of Sofia
„Sveti Kliment Ochridski" Blvd. 8, 1000 Sofia
Bulgaria

ABSTRACT
The engineering design activity becomes extremely complex and is related to huge data volumes, intensive computer calculations and simulations. Hence adoption of new techniques and methods for model representation of the product, common for the systems and software engineers, which will provide a direct translation of the product model into a program code within early design stages, is an important issue. In this paper, the use of the object-oriented approach and Unified Modelling Language (UML) within product development process is examined. Some benefits related to the improving and reducing the design efforts and enhancing the product success in case of technical systems are outlined. An object oriented approach for development of product families of linear handling systems is presented.
Keywords: object-oriented design, unified modelling language (UML), product family

1. OBJECT-ORIENTED APPROACH FOR PRODUCT DESIGN
The software design methodology of object-oriented programming can be applied to engineering system modelling with the benefits of simplified model creation and maintenance [1,2]. In the object oriented modelling the fundamental construct is ***an object***, which combines data structure and behaviour in a single entity to represent the components of a system. The system is considered as a collection of interacting objects, working together to provide the expected solution [3].
Object-oriented design is discussed in details in [2,4]. It is defined as a method of design encompassing the process of object-oriented decomposition and a notion for depicting logical and physical as well as static and dynamic models of the system under design. All entities associated with design are regarded as *objects*. These include design artefacts, design processes, design algorithms and design data. Each design object can contain *methods* (procedures that act on design objects), *data* (a named attributes) and *interfaces* (the basic structure and interface of a design object). Design objects are related to each other by object dependencies. They can be used to develop a Design Process Model for a particular design problem and domain. Central element of design is the Design Model ***(D)***. ***D*** is representation of an artifact to be designed and consists of one or more Design Objects (O_i). Figure 1 presents how a particular instance of a design model, D_I^k is obtained from the Design Algorithm, Evaluation Schema, Requirements, Constraints and the Design Model Object. For formulation design a new design model object ***D*** is defined. A specific instance, D_I^k of this design model can then be created. For parametric design then the design model object ***D*** already exists and the design process involves only the determination of a specific instance, D_I^k [2].

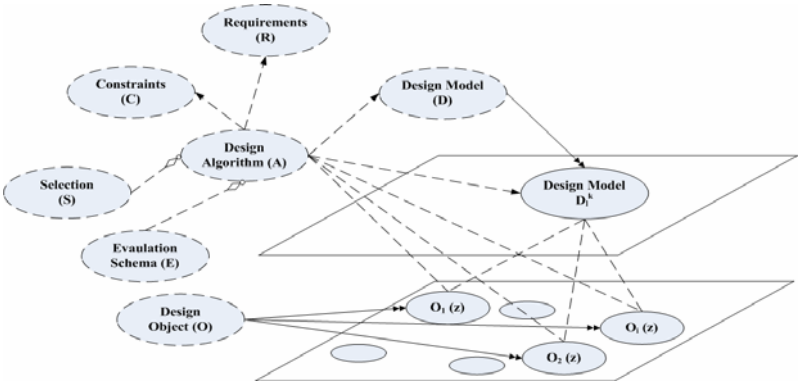

Figure 1. Overall structure of Object-oriented Design [2]

2. DESIGN PROBLEM AND DESIGN ALGORITHM

The design of highly customized modular systems for industrial automation is a complex process and requires many diverse considerations. An important task is to take the requirements and constraints (functional, physical, economical), together with a set of possible mechanical, electromechanical and electronics components (functional units), and then to select a subset of the components to satisfy these requirements and constraints, while minimizing or maximizing the objective function, where some of the participants in the process (customers) are remote. Hence the design problem for developing new handling system can be formulated as follows: *With a given set of candidate components {q_i,.(z)}, at design iteration k, to produce a design D_i^k composed of a subset of the candidate components and which satisfies both a set of functional requirements {$R_{i,p}$} and a set of constraints {C_i(z)} while minimizing (maximizing) an objective function V_l,* where $R_{i,p}$ is functional requirement i with attribute p.

According to [4] such kind of problem can be solved with following three-step design algorithm that underlies the object-oriented design process model:

Step 1: Translation the set of initial design specification into an equivalent set of functional requirements.

Step 2: Importing the required data from Requirements ***(R)*** and Constraints ***(C)*** into the Design Algorithm ***(A)***.

Step 3: Invoking a selection algorithm ***(S)*** which selects a functional requirement and a design object satisfying the requirement.

When all functional requirements have been satisfied the main design stage is completed. The resulting instance of the design model obtained will be one that does not violate the set of constraints ***{C_i(z)}***.

3. UML IN PRODUCT DESIGN PROCESS

Unified Modelling Language (UML) is a graphical language for specifying, constructing and documenting the artefacts of a software-intensive and non-software systems. It offers a standard way to write a system's blueprints, including business processes and system functions [5]. Diagrams perform the graphical representation of a system in UML. They are grouped into two main categories: *Structure* and *Behaviour* diagrams. Structure diagrams define the static architecture of a model, representing conceptual and physical elements (classes, objects, interfaces and physical components) and their relationships and dependencies. Behaviour diagrams describe the dynamic model, representing the interactions and the activities.

4. UML FOR OBJECT-ORIENTED DESIGN OF PRODUCT FAMILIES

The system family paradigm aims towards developing several applications out of a domain with just one underlying architecture. The foundations of this core architecture are the common properties. The development effort for a system as part of a family is lower than the effort for building a single system. As a result, the development process can be performed with little efforts in a shorter time. Modelling and development of common parts and variants have to be supported by methods and notations. UML can be applied to model variants during analysis and design.

After functional decomposition of the system a feature model is used to describe systems mandatory, optional, and alternative properties (features). An important part of the feature model is the hierarchically organized feature diagram. The tree's root specifies the concept being described; the nodes represent the features. A feature is mandatory unless an empty circle is attached to its node, indicating an optional feature. An arc spanning two or more edges of feature nodes depicts a set of alternative features. Fig. 2 shows an example of a simplified feature diagram that lists possible features of a linear modular handling system. Mandatory features whose super-features are neither optional nor included in sets of alternatives represent the common features shared by all family members. All other features are called variable features. Family members may differ from each other with respect to these variable features, because they may or may not choose to implement a variable feature. For each family member there is a list called configuration map containing possible variable features for a member. The configuration map references elements of the design model that are affected by the variable features.

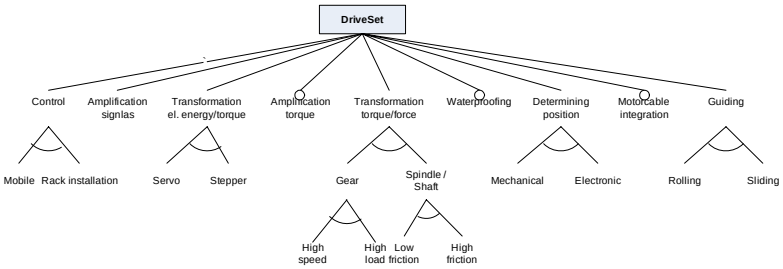

Figure 2. Feature model DriveSet

According to [6] the process for developing system families using feature models looks as follows: With given requirements and domain analysis information the modelling process is started. The feature model describes the common and variable features, which has to be designed in the next step. The system family architecture is abstract and has to be instantiated for a family member through a configuration step. The resulting family member configuration contains just the needed features for the specific problems addressed by this member.
UML can be used in order to describe all aspect of a model, e.g. architecture, static structure, dynamic behaviour, and interfaces. The stereotype «variant» designates the model elements as being variable. Elements annotated with this stereotype have a tagged value with the key "feature" referring the name of a feature in the feature model. Within a configuration step features have to be selected. Fig. 3 illustrates the usage of the UML in component diagram. Further activity and class diagram can be built describing classes, interface, collaborations and their relationships in the system model (Fig. 4).

5. CASE STUDY DRIVE SETS

For the purposes of a ongoing R&D-project a systematic design approach involving the customer within design process was developed. This approach was applied to support and maintain development of modular positioning and handling systems called *DriveSets*, being a scalable frame based parametric range, evolved through the integration of OEM-components. *DriveSets* form a "virtual product family" in accordance with the predefined sets of systems properties (*load carrying capacity, maximum speed, repeatability, operating area, design type, stroke*) described as a product frame. Their specific value for each family member is determined at the design phase on ground of the mapped customer requirement and constraints providing a sort of ***tailored customization***. The design approach compromises four stages and includes use of UML-diagrams (component, activity and class diagrams).

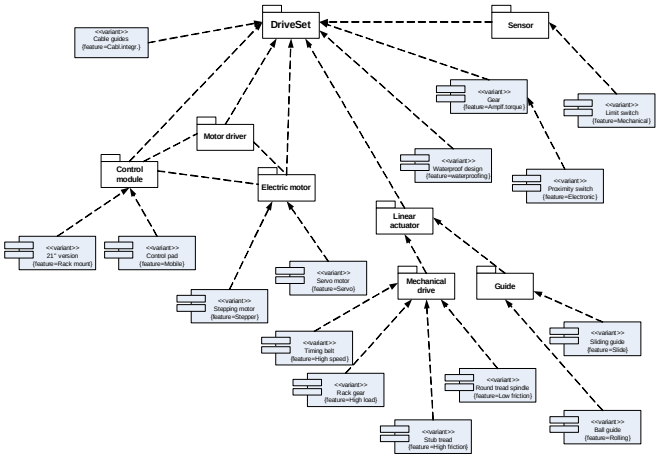

Figure 3. Component diagram DriveSet

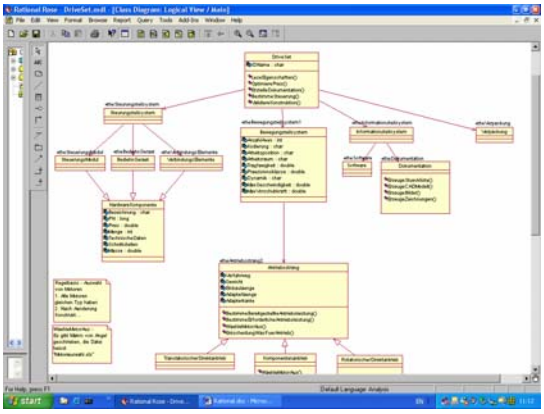

Figure 4. Part of the class diagram DriveSet

6. CONCLUSION

Application of Object-oriented design approach can facilitate design engineers in managing complexity of the problem by abstracting knowledge and behaviour and encapsulating them with objects. It simplifies the tasks of reusing, maintaining, and extending families of models. The object-oriented system models have the following advantages:

- *computability* - the process model is not just a descriptive, but a computable and enables automatic program code generation, it could help to bridge the gap between systems and software engineering;
- *reusability* - the established design objects can be reused;
- *reconfigurabilility* - the modular nature of objects allows for an object to be replaced by another.

Further Object-oriented approach offers a consistent way for describing the model and variability of product families using the UML in the early development phase. The elaborated UML diagrams can serve as a common communication basis between different stakeholders participating in the design process. Hence the cost and efforts within development stage drop. The presented approach possesses improving potential and wide application field.

7. REFERENCES

[1] Gupta, A., Kurniawan, S.H., Tseng M.M.: Product Design and Development in Concurrent Enterprise, Proceedings of EC 04, Hong Kong (2004).

[2] Sinha, R., Liang V.: Modelling and Simulation Methods for Design of Engineering Systems, Journal of Computing and Information Science in Engineering 1, 84 (2001).

[3] Felfernig, A., Friedrich, G., Jannach, D. et al.: UML as Knowledge Acquisition Frontend for Semantic Web Configuration Knowledge Bases, Workshop on Rule Markup Languages for Business Rules on the Semantic Web '02, Sardinia, Italy (2002).

[4] Liang, W.Y., O'Grady, P.: An Object-Oriented Approach to the Concurrent Engineering of Electronics Assemblies. Computers in Industry, Vol. 47, pp 239-254, 2003.

[5] Braun, P., Rappl, M.: Model based Systems Engineering - A Unified Approach using UML. Proc. of the 2nd European Systems Engineering Conference (2000)

[6] Riebisch. M., Böllert K., Streitferdt, D. Franczyk, B.: Extending the UML to Model System Families. IDPT 2000, USA.