Object-Oriented Modeling of Embedded-Software in the Automotive Environment

Michael Benkel, Mehdi Hannouz

Object-Oriented Modeling of Embedded-Software in the Automotive Environment

ERST Conference 2004

Author
Michael Benkel
Director Product Marketing

Aonix Germany
Brauereistr. 11
41352 Korschenbroich
Phone: +49 2161 9985811
Fax: +49 2161 9985820
michael.benkel@aonix.de

Co-Author
Mehdi Hannouz
Pre-Sales Consultant UML/MDA
OMG-Certified UML Professional

Aonix France
66-68 Av Pierre Brossolette
92247 Malakoff, France
Phone : +33 1 41 48 14 73
Fax : +33 1 41 48 10 20
mehdi.hannouz@fr.aonix.com

Biographical details:

Michael Benkel has more than 10 years of experience as a consultant in OO projects, with the main focus on OO modeling in OMT, Booch and finally UML notation.
He is also pioneer in Model Driven Architecture

Today he is the Product Manager of Aonix Modeling Environment. In this role he was the Project Manager for the OMOS Project described in this paper.

Introduction

The development of *motor-vehicle control-systems* software is characterized largely by customized demands on functionality, real-time capability and memory-requirements. These requirements vary not merely from one auto-manufacturer to another but also between different projects of the same manufacturer. Furthermore, this user- and project-specific software must be developed efficiently with regard to costs, time and quality. A structured and easily expandable software architecture which describes complete product families is necessary in order to measure up to the requirements of embedded real-time systems in the future. This software must support the forming of variants and the re-use of code effectively without leading to additional run-time- and memory requirements.

The modeling concept for control-systems software described in this article uses object-oriented techniques *while avoiding* additional run-time- and memory requirements, with C as the target language. OMOS also desrcibes how to model and form variants of software systems. OMOS is thus more than a predefined OO2C Mapping.

The tool-support for OMOS is a development by Aonix (www.aonix.de) based on the product family Software through Pictures.

The development of software for ECU‘s

The demands on functionality and security of control systems software in vehicles are constantly on the increase. This results in a significant growth in the amount of software in these systems and consequently in the complexity of the software. This poses the problem, especially for the auto-supplies industry, of how to keep the increasing variety of control systems software for the various auto-manufacturers and their models under control. These demands can only be met by the use of a structured and expandable software-architecture as well as efficient software development.

A high degree of re-use of existing components and code, as supported by inheritance in object-oriented technology, is necessary to achieve these goals. Re-use in this sense means that existing components, which have already been used in a particular context, are used again in a new context, for example in other projects. Quality is raised by the re-use of software which has been tested and already been employed. Components which are repeatedly used become more and more stable and the re-use also contributes to savings in time and money in the creation of the software.

The use of inheritance allows the forming of variants and consequently the alternative use of components. The goal of forming variants is to define new components, building on those which already exist, whereby parts of the functionality are changed or supplemented. This alternative use finally makes it possible to use variants of a component instead of the component itself, without having to change the coding of the component. Not all concepts of object-oriented technology are directly applicable in developing control-systems software, due to the requirements on run-time and memory. Object-oriented languages such as C++ are not sophisticated enough for real-time systems and because of this one often has to fall back on the programming language C.

OMOS at a glance

The modeling concept OMOS is tailored to the development of control-systems software. The consideration of rigorous demands on memory and run-time is thereby extremely important. OMOS confines itself to a certain subset of object-oriented modeling, which makes an optimized mapping to C-code possible. Known concepts, such as object, class and inheritance as well as inline- and virtual methods etc. are emulated in C. Multiple inheritance, template classes and dynamic generation of objects are not used.
Figure 1 shows the process in development with OMOS.

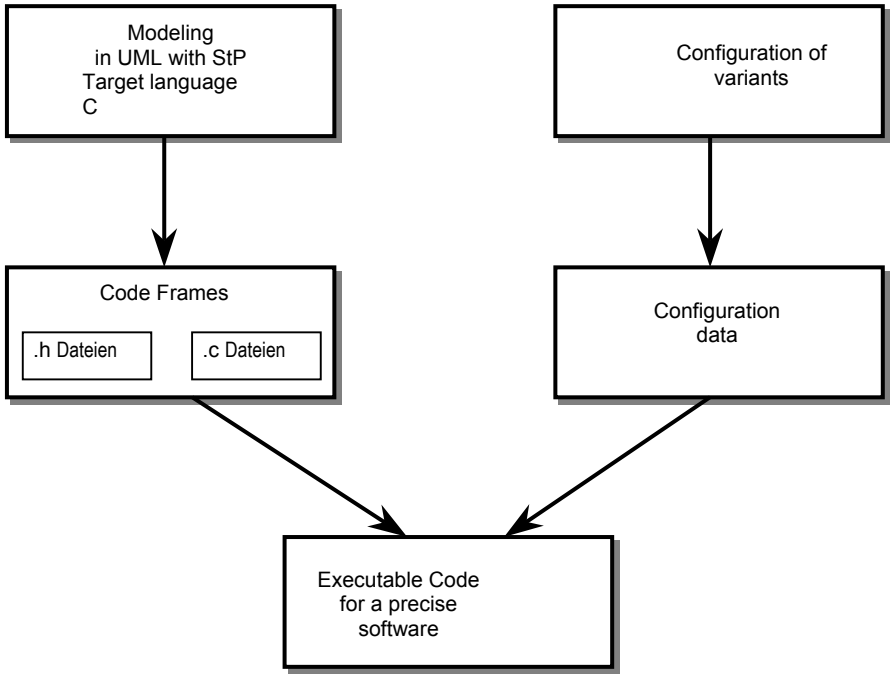

Figure 1: Development Process with OMOS

The modeling occurs in UML notation with a profile fitted to the particular circumstances. The configuration of a precise control-systems software and the code generation are then tailored to this UML profile.

UML Profile for OMOS

UML stereotypes are used to differentiate between 1-Class and N-Class. An N-Class is equivalent to the class perception known from object orientation and can be instantiated as often as desired. Contrary to this, there can only be one instance of classes with the stereotype 1-Class. This characteristic provides great potential for optimization in the implementation in C with regard to run-time and memory.. For example, all attributes of the instance of a 1-Class can be implemented as global variables in C. The dereferencing of objects and the transfer of object references thus no longer apply.

This potential for optimization can be used to great advantage in the development of control systems software as the number of 1-Classes is very high (for example motor, gears, pedal).

Composition, communication and variant relationship based on inheritance are counted among the relationships supported by OMOS.

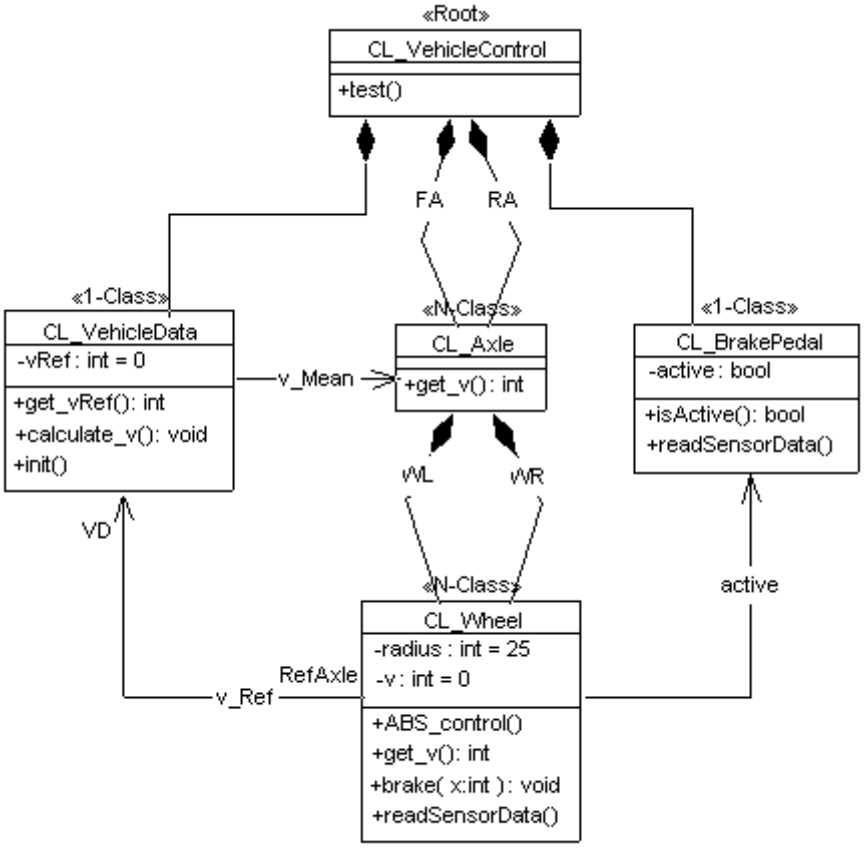

Figure 2: Modeling with OMOS

Figure 2 shows the modeling of a simple control systems with the use of the OMOS profile. Classes such as CL_BreakPedal are given the stereotype 1-Class, as they are only found once in a vehicle. As axles and wheels are found more than once in a vehicle they are given the stereotype N_Class. The composition relationships with the names FA, RA and WL, WR determine that there are exactly two instances of CL_Axle with two 2 instances of CL_Wheel each.

Forming Variants

A variant is expressed in UML notation by an inheritance relationship. A new (class) variation is modeled as a sub-class of a more general class variation. The new variant inherits the structure, the behavior and thereby the interface of the more general variation. The new class variation can be supplemented with

additional attributes, methods and relationships and can overload the implementation of inherited methods. Inherited characteristics cannot be dispensed, as this would damage the conformity with the interface of the super class. The variant relationship describes a sort of pool of possible variations of control-systems software in a class model.

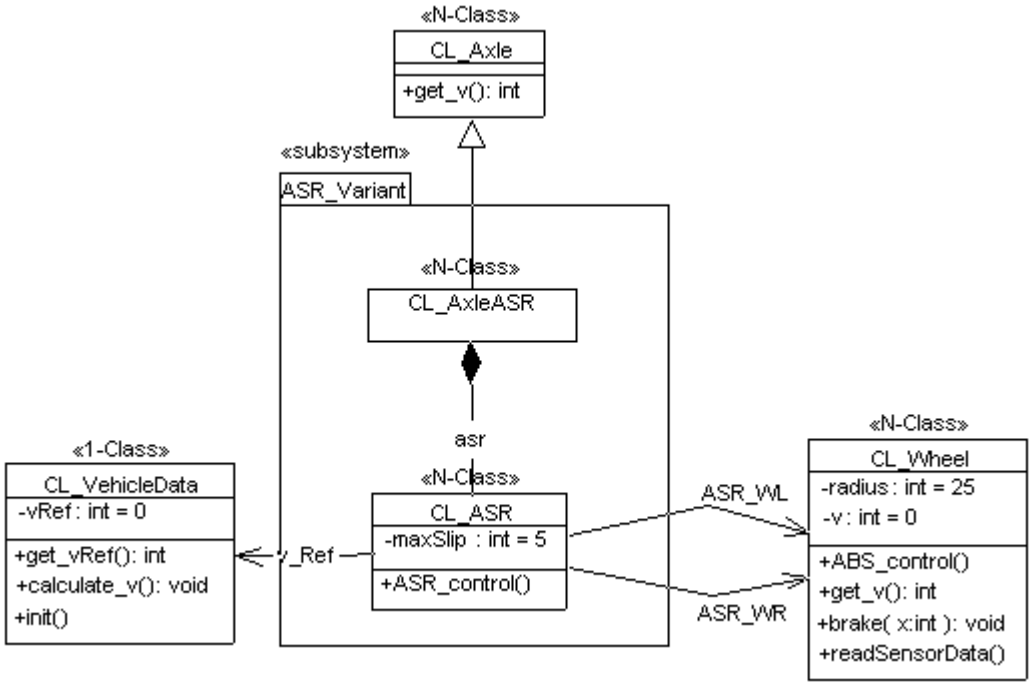

Figure 3: Forming Variants with OMOS

In the above example the control-system software is expanded by an ASR functionality.

Configuration

OMOS models with variants always contain more than one SW system. A configuration thus describes all instances which make up a particular system as well as to which class they belong. The description of the objects takes place in a class diagram with UML object symbols.

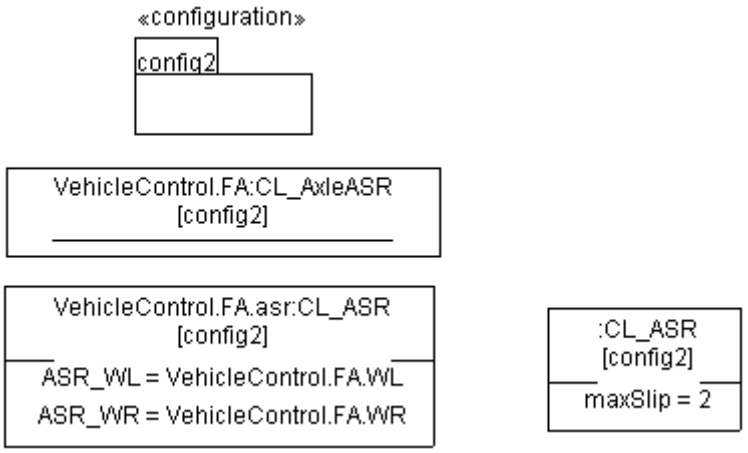

Figure 4: Configuration with OMOS

The above example configurates an ASR system on the rear axle. Instances of the classes CL_AxleAsr and CL_ASR are configured for this purpose. The communication relationships ASR_WL, ASR_WR and the attribute maxSlip are also initialized in the case of the instance CL_ASR.

Code generation

The generation of source code from OMOS models is based on templates. These templates then come into use depending on the stereotype of a particular class. The use of templates ensures that all classes are generated consistently and that a technological transfer to embedded C++ is later possible.

The target language is C as regards real-time capability and memory requirements. In addition, corresponding source is generated for the modeled configuration. The instances for a certain software system are declared and the class attributes initialized in these configuration data.

Summary

OMOS is a concept for modeling control systems software. The key aspects of this concept are the forming of variants in a model and the mapping of these models to the programming language C. OMOS is thus more than a predefined OO2C Mapping.

Today, the major areas of use for OMOS are to be found in the automotive environment, where there is an array of successful projects to be found. The concepts can also be applied to other industrial sectors.

World Headquarters
Aonix
Batîment B
66/68, Avenue Pierre Brossolette
92247 Malakoff cedex
France
Tel: +33 1 4148-1000
Fax: +33 1 4148-1020
E-Mail: info@aonix.fr
www.aonix.fr, www.aonix.com

Germany
Aonix GmbH
Emmy-Noether-Str. 11
D-76131 Karlsruhe
Tel: +49 721 986530
Fax: +49 721 9865398
Email: info@aonix.de
www.aonix.de