Model Driven Development with Mechatronic UML

Wilhelm Schäfer

University of Paderborn

Abstract. Model Driven Development with Mechatronic UML Visual languages form a constituent part of a well-established software development paradigm, namely model driven development. The structure and functionality of the software is precisely specified by a model which can be formally analyzed concerning important (safety and liveness) properties of the system under construction. Executable code is automatically generated from the model.

Although model-driven development has been recognized as a potential to improve significantly the productivity and quality of software. success stories are restricted to particular domains, mainly in business applications. Other domains, especially embedded systems employ model-driven development only in very special cases and on a limited scale, namely in the continuous world of control theory. This is due to the complex nature of the software of advanced mechatronic (or embedded) systems which includes complex coordination between system components under hard real-time constraints and reconfiguration of the control algorithms at runtime to adjust the behavior to the changing system goals. In addition, code generation must obey very limited and very specific resources of the target platform, i.e. the underlying hardware or operating system. Finally, techniques for modeling and verifying this kind of systems have to address the interplay between the discrete world of complex computation and communication and the "traditional" world of continuous controllers. The safety-critical nature of these systems demands support for the rigorous verification of crucial safety properties.

Our approach, called Mechatronic UML addresses the above sketched domain by proposing a coherent and integrated model-driven development approach. Modeling is based on a syntactically and semantically rigorously defined and partially refined subset of UML. It uses a slightly refined version of component diagrams, coordination patterns, and a refined version of statecharts including the notion of time. Code generation obeys the resource limits of a target platform. Verification of safety properties is based on a special kind of compositional model checking to make it scalable. The approach is illustrated using an existing cooperative project with the engineering department, namely the Railcab project (www.railcab.de). This project develops a new type of demand-driven public transport system based on existing and new rail technology.

G. Stapleton, J. Howse, and J. Lee (Eds.): Diagrams 2008, LNAI 5223, p. 4, 2008.