Abstract

Requirements for technical products, e.g. cars or planes, increase continuously. A cooperation between mechanical, electrical and software components is necessary to cope with these increasing requirements. Systems built of such components are called mechatronic systems.

A system like that can collect information about its environment using sensors. Additionally, it is able to communicate or cooperate with other mechatronic systems. Software enabling this interaction is safety critical, i.e. erroneous behaviour of the software can cause high financial costs or in worst case can claim human life. On the other hand this kind of software is quite complex and the state space of such software is usually infinite. Therefore testing on its own is not sufficient to satisfactorily show the software's correctness. Automatic formal verification approaches like model checking are only capable to handle systems with a finite state space. Contrary, theorem proofers can cope with infinite state spaces but require inputs of users with a strong background on formal methods.

For these reasons the thesis at hand introduces an approach which is compositional and can automatically and formally verify the software implementing the interaction between several mechatronic systems. The idea is based on an existing approach called meachatronic UML.

In mechatronic UML a system state is characterized by the existing mechatronic systems and their interaction and can be specified as a graph. This fact is exploited by the introduced approach. State changes like starting or stopping an interaction can be specified as graph transformation rules. In addition it is possible to locally verify the safety requirements of interest. A graph, and therewith also the state specified by this graph, is correct if it fulfils all given safety requirements. Given a set of graph transformation rules and a set of safety requirements the approach shows that the rules cannot transform a correct graph into an incorrect one. The reachability of a certain graph is not considered. Thus the verification checks whether the safety requirements are inductive invariants of the overall system.