Comments on the Vestal Task Model and Its
Scheduling in Time-Triggered Systems

Gerhard Fohler, Technische Universität Kaiserslautern, Germany
fohler@eit.uni-kl.de

vestal task model critique; time triggered systems, offline scheduling tables

First, we briefly argue that Vestal task model scheduling (Vetos) is only a small aspect of mixed criticality system design and thus not sufficient to address mixed criticality systems.

Then, we survey some issues related to Vetos in the context of time-triggered systems. We sketch two methods for Vetos based on offline scheduling tables: one applying principles of mode changes for time-triggered systems for construction of scheduling tables from scratch, the other based on legacy tables.

1 Time Triggered Systems

Current practice in many domains, including (the safety-critical components of) automotive and avionics systems, which must meet multiple assurance requirements up to the highest criticality levels (e.g., DAL A in RTCA DO-178B or SIL4 in EN ISO/IEC 61508) favors a time-triggered approach (TT) [3]. In such TT systems, non-interference of safety-critical components by non-critical ones is ensured by strict isolation between components of different criticalities: Although such isolation facilitates the certification of the safety-critical functionalities, it can cause very low resource utilization.

A first result [1] shows proof-of-concept of mixed criticality real-time scheduling based on the TT approach. It is based on the offline construction of two coordinated schedule tables and an online mechanism to handle a change in criticality.

2 Construction of new tables

We sketch an algorithm [6] for constructing the schedule tables and their run-time execution that result in far more efficient resource utilization and enhanced flexibility; in contrast to the highly simplified proof-of-concept tables obtained by the techniques of [1], these tables can be used for actual system implementation. With the low criticality schedule table obtained by our algorithms, the system designer can guarantee correct system behavior based on his assumptions; with the high criticality scheduling table, the high-criticality applications are guaranteed based on CAs’ pessimistic assumptions. We construct schedule tables such that we can switch from the low to
the high criticality schedule table at any point in time — that is, the tables are shown to possess what is commonly known as the switch through property. We separate the workload of low and high criticality jobs by splitting the WCETs and the jobs such that for each job we obtain only one WCET. We split high criticality jobs into a portion of the WCET which is contained in both criticality levels and the remaining WCET of the high criticality WCET. As a result, we obtain two jobs: first, a job whose WCET represents the original WCET based on designer assumptions. Second, a job which represents the additionally needed WCET to fulfill CAs’ requirements, i.e. the introduced pessimism by the CAs. Precedence constraints between these two jobs ensure the switch through property and the guaranteed WCET in the high criticality case for high criticality jobs. We devise a novel search-tree based mechanism that allows for the simultaneous construction of both schedule tables. We use heuristics to reduce the complexity of the search tree and the amount of backtracking that is needed during the search process. Based on the scheduling decision in the low criticality schedule table and possibly backtracking, slots in the high criticality schedule table are scheduled such that we obtain the switch through property.

3 Legacy tables

In legacy TT systems, i.e., with existing, certified tables, this algorithm cannot be applied as it requires existing schedule tables to be changed, incurring substantial effort for recertification. At runtime, the low-criticality schedule table is executed until a high criticality job shows high criticality behavior and then the system switches to the high criticality schedule table. This solution shows a low runtime overhead but at cost of inflexibility.

We sketch a method [5] to add the handling of criticality changes to existing schedule tables for legacy TT systems. It analyzes the existing table and properties of the high-criticality job set off-line. A simple online mechanism then executes the jobs according to the existing table, manages a change of criticality, and then continues to execute the high-criticality job set. In case the existing schedule table is not suitable for the given mixed criticality job set, indications for its modification can be given. While in this case recertification may become necessary, the efforts will be lower than reconstruction of the schedule table from scratch.

Our method is based on slot-shifting [2] which was originally designed to add flexibility to TT systems with acceptable runtime overheads [4]. It takes original task set and a constructed scheduling table as input. It does not depend on a particular offline table construction algorithm. As the table is constructed offline, complex constraints, such as distributed systems, end-to-end deadlines, precedence, etc can be considered. It analyzes table and constraints to determine unused resources and leeways, which are represented as spare capacities offline. These can be used to provide flexibility and handle firm aperiodic tasks at runtime. Here, we build upon the offline analysis part and spare capacities to handle changes in criticality.
References


