IAF MICROGRAVITY SCIENCES AND PROCESSES SYMPOSIUM (A2)
Facilities and Operations of Microgravity Experiments (5)

Author: Mr. Benjamin Weps
DLR (German Aerospace Center), Germany, benjamin.weps@dlr.de

Mr. Daniel Lüdtke
German Aerospace Center (DLR), Germany, daniel.luedtke@dlr.de
Mr. Tobias Franz
DLR (German Aerospace Center), Germany, tobias.franz@dlr.de
Dr. Olaf Maibaum
German Aerospace Center (DLR), Simulation and Software Technology, Germany, Olaf.Maibaum@dlr.de
Dr. Thijs Wendrich
University Hannover, Germany, wendrich@iqo.uni-hannover.de
Mr. Hauke Müntinga
ZARM University of Bremen, Germany, hauke.muentinga@zarm.uni-bremen.de
Dr. Andreas Gerndt
German Aerospace Center (DLR), Simulation and Software Technology, Germany, Andreas.Gerndt@dlr.de

A MODEL-DRIVEN SOFTWARE ARCHITECTURE FOR ULTRA-COLD GAS EXPERIMENTS IN SPACE

Abstract

Developing software for large and complex experiments is a challenging task. It must incorporate many requirements from different domains, all with their own conceptions about the overall systems. An additional level of complexity is added if the experiment is conducted autonomously during a sounding rocket flight. Without a proper software architecture and development techniques, achieving and maintaining a high code quality is a very cumbersome task.

This paper describes the architecture and the model-driven development approach we used to implement the control software of the experiments in the MAIUS-1 mission (matter-wave interferometry in microgravity). In this mission, the software had to handle around 150 experiments in six minutes autonomously and adapt to changes in the control flow according to real-time data from the experiment.

The MAIUS-1 mission was the first mission to create Bose-Einstein condensates in space and conduct other experiments with ultra-cold gases on a sounding rocket. Besides the scientific goals in the area of quantum-optics, other important objectives of the mission were the miniaturization and further development of laser systems, vacuum components, optical sensors, and other related technologies. To fulfil these goals, new experimental hardware has been created which had to be integrated and tested with the software of the experiment computer.

The custom-made hardware and the considerable number of domains involved brought up many challenges for the software engineering. To face all these challenges of developing software with this high complexity, we chose to follow a model-driven software development approach. Several domain-specific languages (DSLs) accompanied with specialized tools were created to allow the physicists and electronic engineers to describe system components and the experiments in a domain-specific way. These descriptions were then automatically transformed in C++ code for the flight software. This way we could actively incorporate all the domains involved in conducting the experiment directly in building the flight software without compromising the software quality.

We created a versatile software platform not only for the MAIUS-1 mission but also for upcoming missions with similar experiments and hardware. With our approach we were able to generate around 84% of the source code for the final flight software from the domain-specific models. Besides the improvement of the development process, the code generation made a significant contribution to the overall software quality as almost all manual coding of error-prone boilerplate code could be mitigated.