SYMPOSIUM ON SAFETY, QUALITY AND KNOWLEDGE MANAGEMENT IN SPACE ACTIVITIES (D5)
Safety of Vehicules and Ground Segment for Aerospace Missions (1)

Author: Prof. Jie Li
Beijing Special Engineering Design and Research Institute (BSEDI), China, shanjh@163.com

Dr. Jin-Hui Shan
Beijing Special Engineering Design and Research Institute (BSEDI), China, shanjh@163.com
Dr. Zong-Sheng Cao
Beijing Special Engineering Design and Research Institute (BSEDI), China, shanjh@163.com
Mrs. Sheng-Fen Niu
Beijing Special Engineering Design and Research Institute (BSEDI), China, shanjh@163.com

RESEARCH ON ACTIVE TESTING METHOD OF SOFTWARE IN SPACEFLIGHT LAUNCHING SITES BASED ON CONTRACTS

Abstract

Software plays an important part in spaceflight launching sites, especially in manned spaceflight engineering. Software faults may damage facilities and equipments, bring about heavy loss of money, even threaten the safety of astronauts. Software testing is one of the most important techniques used to assure the quality of spaceflight software products at present. Contracts which are commonly written as annotations of source code are used to describe the valid behavior of the software formally in the software design phase. Contracts have the form of assertions, pre-conditions, post-conditions, etc. In this paper, the Buckus-Naur Form is defined to describe the contracts grammar in the C programming language, based on which an active testing method is proposed to detect software faults according to contracts. At first, contracts following the defined grammar are inserted into the program by programmer. Then these contracts are extracted. Negative operation is applied to each contract in turn by looking up templates, where the negative contract describes the fault state of the software. After which some program paths are constructed, and each negative contract is translated into the corresponding codes to be inserted into one program path appropriately. At last path-oriented test data generation method is applied to obtain one test datum for each program path to be traversed as follows: executing the statements along the program path using an arbitrary input, then deriving a system of constraints in the input variables, solving the system of constraints iteratively to obtain the desired test datum. The test datum obtained will discover a software fault and make the fault reoccur stablely. A supporting tool is developed and put into practice to validate the effectiveness of our method. The practice result shows that our method is effective to discover software faults which the existing software testing methods can not discover easily, and will push forward the software testing in our spaceflight launching sites.