一种航天用Linux系统工程化方法

张天湘[1],黄建军[2],李皖玲[1],孙　勇[1]
(1.北京空间飞行器总体设计部,北京 100094;2.长城信息股份有限公司,湖南 长沙 410205)

摘　要: 随着航天器功能的越发复杂和软件规模的提高,航天领域常用操作系统存在驱动、第三方库资源不够丰富等问题。Linux系统具有稳定、开源、支持丰富等特点,被广泛应用于高可靠领域,因此有望满足航天领域复杂应用下操作系统的需求。但Linux系统代码量庞大,在航天领域软件工程化和可靠性保证工作方面开展困难。结合航天工程应用实际,提出并实践了按商用和自研,分别对Linux系统软件进行质量保证的工程化方法,为航天用操作系统相关产品的研制提供工程化思路。

关键词: 航天; Linux; 工程化; 可靠性; 方法
中图分类号: V 44　　**文献标志码:** A　　**DOI:** 10.19328/j.cnki.2096-8655.2023.05.018

An Engineering Method for Aerospace Linux System

ZHANG Tianxiang[1], HUANG Jianjun[2], LI Wanling[1], SUN Yong[1]
(1.Beijing Institute of Spacecraft System Engineering, Beijing 100094, China;
2.Great Wall Information Industry Co., Ltd., Changsha 410205, Hunan, China)

Abstract: With the increasing complexity of spacecraft functions and the improvement of software scale, there are many problems in the operating systems commonly used in the aerospace industry, e.g., insufficient drivers and third-party library resources. Linux system has the characteristics such as stability, open source, and rich support, and has been widely used in high-reliability areas. Therefore, it is expected to meet the demands of complex applications to the aerospace industry. However, the amount of Linux system code is huge, and it is difficult to carry out aerospace software engineering and reliability assurance work. Based on the practical application of aerospace engineering, in this paper, an engineering method for Linux system software quality assurance is proposed according to the commercial and self-developed demands, which provides engineering ideas for the development of aerospace operating system related products.

Key words: aerospace; linux; engineering; reliability; method

0　引言

随着技术的发展,尤其伴随载人航天、深空探测等重大航天工程的实施,航天器功能和复杂度不断得到提高,航天应用的软件产品越来越多,规模越来越大[1-4]。对高可靠和稳定操作系统应用的需求越来越大,面对功能接口复杂的任务需求时,SpaceOS、Vxworks、QNX等航天领域常用操作系统[5-9]存在驱动和第三方库资源不够丰富的问题[10-13]。同时,基于Linux内核的系统具有稳定性、安全性、开源、便于移植、第三方支持广泛等优点,被广泛应用于服务器领域、银行系统、汽车、高铁、军用等高可靠场景[14-17]。在面对复杂功能和接口的航天领域应用时,基于Linux的操作系统具有应用的可行性[18-20]。

航天作为高投入高风险行业,软件产品质量直接影响航天任务的成败,因此对软件的可靠性和安全性提出了更高的要求。通过保证软件的可靠性,提高和确定整个产品的可靠性在航天领域至关重要[21-23]。其中软件工程化是保证软件质量的有效方法[14]。由于国内航天领域要求软件代码需经过充分分析和评测[24],因此Linux系统在航天领域的应用面临分析、评测等工程化问题。基于Linux内核的系统软件规模庞大(根据应用情况一般高达数百万行),按照正常

收稿日期: 2023-05-15; **修回日期:** 2023-08-29
作者简介: 张天湘(1986—),男,高级工程师,硕士,主要研究方向为载人航天器人机交互系统设计。

的工程化方案无法实施,本文结合工程实际,探讨航天领域嵌入式Linux应用的工程化方法。

1 工程化思路

在载人航天仪表系统领域,典型的基于嵌入式Linux的系统软件组成如图1所示。由图1可知,该软件主要有UBOOT模块、Linux内核、板级支持包模块和图形库模块。

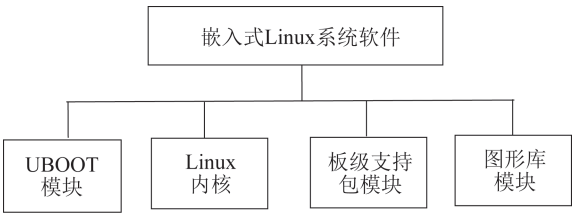

图1 嵌入式Linux系统软件框架

Fig. 1 Framework of the embedded Linux system

由于Linux系统软件的规模庞大,为了提高软件的可靠性,保证人力物力的最优化使用,就需要在测试过程中有重点的进行分析和评测[26]。因此需对系统软件的代码进行重点与非重点的划分,根据不同的情况,开展可靠性保证工作。主要流程如图2所示。

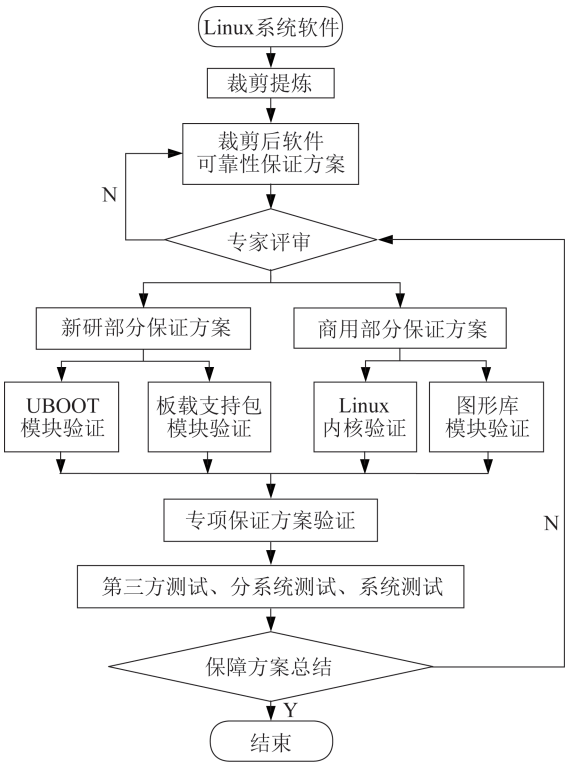

图2 基于Linux的系统软件工程化流程

Fig. 2 Process of Linux system software engineering

1) 结合用户层需求梳理应用层涉及到的操作系统功能模块,选用稳定的内核版本,使用专业工具,进行代码裁剪和提炼。主要包括对操作系统的代码进行功能代码定位,划分代码重点与非重点,缩小代码范围。

2) 针对划分结果和圈定范畴,制定软件可靠性保证方案,经评审后实施。

3) 根据不同的模块,分别制定专项的可靠性验证措施。其中UBOOT模块和板级支持包模块为新研软件,按照航天软件工程化实施细则实施;对于商用Linux内核和图形库模块,根据提炼重点进行针对性的测试和保证。

4) 各模块经过验证后,根据Linux操作系统的特点,采取相应的专项保证措施。

5) 完成开发方软件可靠性保证工作后,进行第三方测试、分系统级和系统级测试验证,经评审通过后,完成软件保证总结。

2 工程化方法

2.1 代码裁剪和提炼

代码裁剪针对具体的应用需求和系统内核版本进行裁剪,降低系统无关冗余代码,主要针对内核和图形库模块。UBOOT模块仅保留系统需要的引导模块,代码量由百万万余行降低至不足3 000行,板级支持包为7 000余行。裁剪稳定版本的内核,规模由千万行经过裁剪降至40余万行,图形库规模由80万行裁剪至30万行。

根据应用需求的重要程度和安全等级,对裁剪后的Linux系统软件的任务功能进行梳理,并按照是否影响任务成败因素划分关键等级,作为代码提炼的初始条件,同时搭建硬件动态测试环境。

针对Linux系统软件各功能及函数的调用分析,采用Valgrind工具完成。使用Valgrind工具运行应用软件,操作应用软件的各项功能,动态分析系统软件调用情况,得出系统软件函数调用关系图。关系图能够清楚地显示函数调用关系、调用次数、占用百分比等特征。通过对关系图展开综合分析,得出系统关键函数表。统计软件提炼了Linux内核和图形库各能模块的关键函数,为后续软件可靠性质量保证划定了范围。

系统软件中的UBOOT模块主要完成Linux系统引导的工作,参照对应平台进行自主研发。板级

支持包模块主要包括与外设有关的驱动代码。2个模块代码规模较小，均按新研制软件开展工程化工作。

2.2　模块测试和保证方案

为保证Linux系统软件的可靠性，需对裁剪提炼后的系统软件进行各项测试，包括单元测试、部件测试、配置项测试、人工走查、静态分析和动态测试，同时，分析系统软件的故障模式及失效模式。从工程化可实施的角度，对系统软件的不同模块，分别制定工程化方案。

1）新研模块（UBOOT和板级支持包）

UBOOT和板级支持包需配合全部源代码的静态分析、单元测试、人工走查和功能测试，进行配置项测试。

2）商用模块（Linux内核及图形库）

Linux内核和图形库选用的商用稳定版本，措施主要包括对裁剪和提炼的代码中的关键函数进行单元测试，结合用户应用软件完成动态跟踪测试，最后进行完整的配置项测试验证。

基于Linux系统软件进行配置项测试，主要测试项目包括功能测试及性能测试。其中系统引导模块报告系统的运行环境初始化、系统代码搬移和运行跳转。系统内核模块负责任务管理与调度、任务间同步与通信、时钟管理、中断管理、异常管理功能、内存管理、文件系统管理、I/O系统、C语言函数库、软件升级和下载功能。图形库程序模块包括显卡初始化模块、基本图元绘制接口、着色接口、基本控件接口、页面生成接口、帧缓存接口、坐标转换、投影变换函数等接口。

在开展相关测试的同时，需进行系统软件的故障树分析和失效模式分析，保证系统软件的可靠性，主要工作内容如下。

1）故障树分析

通过对系统软件进行故障分析，指导软件的可靠性和安全性设计，同时确定软件测试的重点和内容，保证软件的可靠性和安全性。对系统软件进行软件故障树分析，得出故障发生时的底事件。为确保系统软件的功能正常完成，通过设计、审查、测试等手段，避免保证相应底事件的发生。

2）失效模式分析

对操作系统进行软件失效模式及影响分析（System Failure Mode and Effects Analysis，SFMEA），经分析找出软件所有可能存在的、危害软/硬件综合系统可靠安全运行的失效模式，进而分析其产生的软件或硬件的失效原因及影响，并在设计上采取相应的改进措施，以保证软/硬件综合系统的可靠安全运行。

2.3　专项保证方案

为了保证基于Linux内核的系统软件的可靠稳定，根据用户应用和系统联合展开针对性的专项测试，具体测试项目如下所示。

1）供调用资源复核

编写系统软件测试用例，监测系统软件对应用软件提供的各种资源，包括文件系统、内存资源、寄存器、数据缓冲区等进行测试，分析数据正常及异常时资源的使用情况，验证系统软件在正常情况下、临界情况及溢出情况下是否有保护机制及容错措施。将分析结果及使用约束，落实到系统软件和应用软件的接口文档中，作为其应用软件研制的依据。

2）内存监测分析

由于应用程序持续的动态使用内存，如果不及时释放，随着时间延长，内存使用量越来越大，最终崩溃。Linux系统软件的内存空间是经过重映射的，用户使用的内存地址不对应实际内存地址。内存地址分为内核态和用户态，内核访问内存地址和应用软件访问内存地址分开，不能直接互相访问，需调用特殊的函数访问，确保用户进程任务无法破坏系统内存，保障系统稳定。用户进程任务使用内存地址也完全独立，使得各个用户进程之间不影响。

通过Valgrind检测应用程序进程，分析内存使用释放情况，杜绝应用软件运行中出现内存泄漏、数组越界、除数为零、非法地址访问等操作的可能性，确保系统可靠运行。

3）数学库函数验证

应用软件会根据用户需要进行数学计算，调用系统软件的数学库函数。如果数学库函数的计算值有误，会导致应用任务执行失败，因此需逐一验证系统软件内的数学库函数。具体验证方法是：针对某一函数，在不同的取值空间范围内取10个数据进行测试，包含对边界数据的测试，确保数学库函数计算的数据准确可靠。

4）编译选项复核

软件编译过程中的编译选项会影响软件编译

后的可执行文件。调试系统后,修改编译选项生成调试版本,调试版本与最终版本程序相比,只是在程序中加入了调试信息,其他相同。对系统软件及应用软件编译过程中用到的编译选项进行复核,不能随意变更软件的编译选项。

5) 错误码提示分析

系统软件定义了一套错误码,在软件出现错误或崩溃时,提供错误信息帮助用户快速定位错误位置。针对系统软件错误码,编译测试用例进行错误码提示分析,保证错误提示的可靠。验证错误码提示时,检查系统软件对错误状态是否有相应的保护及容错机制,分析对应用软件功能实现的影响。

6) 异常处理

系统软件应具备异常处理(CPU执行指令异常、非法地址访问等)的功能,使用应用软件进行边界测试、访问非法地址等手段制造异常,检测系统软件的异常处理能力,确保囊括所有的异常情况。

7) 图形库模块分析

图形库采用开源软件稳定版本,被广泛应用于工业控制系统、功能手机、数码相框等类型的产品中。此外,对系统软件的图形库模块进行专项分析,测试图形库实现的按钮、文本框、编辑框、输入法等,对界面显示和视频流显示的画中画效果、显示切换效果编写测试用例进行测试。图形库的测试用例涵盖图形库的各个功能,测试图形显示效果符合应用需求。

8) 运行瓶颈分析

进行系统软件的瓶颈分析是实现系统软件最优性能的常用方法。系统硬件定型后,系统各个子系统资源确定,一般与系统性能相关的子系统有CPU、内存、驱动和网络。分析应用软件资源占用率,明确存在影响应用软件的运行瓶颈的子系统,通过优化程序结构,实现系统运行最优化。

9) 运行竞态分析

Linux系统软件是多进程多线程系统,各个进程和线程是并行运行的。进程和线程的数据共享采用共享内存、队列、信号灯等方式。并行运行的各个进程和线程会产生竞态冒险等问题,所以对应用软件使用竞态分析,合理运行自旋锁、信号灯等机制,确保程序运行稳定、结果唯一。

2.4 第三方评测与系统验证

完成开发方的软件可靠性保证后,设备需进行第三方评测与系统验证。操作系统和应用程序共同运行在真实设备平台上,并以真实设备连接至设备各外部接口,实现对设备的输入、输出及响应。在分系统级和系统级测试环境中,模拟真实信息输入,开展产品软硬件性能联合测试。同步开展的第三方评测专家结合工程化方案中的评测项目和内容,按照工程化要求,完成对UBOOT和板级支持包代码的需求说明、编码规范检查、人工走查、动态测试等相关项目。对商用部分的Linux内核及图形库提炼的函数接口及代码文件,参考保证方案有重点的完成相关项目。

完成开发方、分系统用户、总体用户和第三方评测等所有验证工作后,基于Linux内核的系统软件,经航天器真实运行环境下的考核验证,利用航天器初样和正样进行各项测试试验,完成软件可靠性工程化的考核,最终执行航天任务。

3 工程化实践

为应对单粒子等空间环境,在工程实施过程中,在产品设计层面增加2项措施:1) 设计监控进程中,监测其他用户进程,针对周期性进程采用计数监测,发现计数异常越界后重启该进程;针对非周期性进程采用状态监测,对非预期状态进行进程重启。2) 系统程序内核和文件系统采用分区双备份方案,在启动过程中,识别系统复位日志,选择可靠的固件加载系统软件。

截至目前,工程化后的系统软件已随载人航天器在轨工作2年,工作状况稳定。

4 结束语

本文根据航天应用的需求和Linux系统的特点,结合工程实际,提出一种基于Linux的系统软件工程化方法,并加以实践。按照新研部分和商用部分的划分系统软件,根据应用的需求和软件的特点,对不同的软件模块制定相应的测试和保证方案,验证基于Linux内核的系统软件可靠性和安全性,满足航天器型号软件工程化管理的要求。在航天任务实践中增加监护进程和双备份固件方法,保证了系统和产品的可靠运行,为其他相关领域操作系统软件的应用和工程化提供了一个解决思路。

参考文献

[1] 杨宏.从空间实验室到空间站的总体设计思路[J].航天器工程,2022,31(6):7-14.

[2] 黄建青,罗鹏,戴志晃,等.空间站多终端音频集中处理网络架构实现[J].上海航天(中英文),2022,39(S2):161-167.

[3] 赵丽,李超,王冰,等.基于JetsonTX2处理器的星载操作系统设计与验证[J].航天器工程,2022,31(5):51-59.

[4] 倪光南,朱新忠.自主可控关键软硬件在我国宇航领域的应用与发展建议[J].上海航天(中英文),2021,38(3):5-10.

[5] LEPPINEN H, NIEMELA P, SILVA N, et al. Developing a Linux-based nanosatellite on-board computer: Flight results from the Aalto-1 mission[J]. IEEE Aerospace and Electronic Systems Magazine, 2019,34(1):4-14.

[6] MERL R,GRAHAM P.A low-cost,radiation-hardened single-board computer for command and data handling [C]// Aerospace Conference. IEEE,2016:1-8.

[7] BIRBACH O, FRESE U, BAUML B. Realtime perception for catching a flying ball with a mobile humanoid [C]// IEEE International Conference on Robotics and Automation. IEEE,2011:5955-5962.

[8] 詹盼盼,郭坚,何熊文,等. 一种面向航天的可靠实时操作系统设计[J].空间控制技术与应用,2016,42(5):47-51.

[9] 陆文,徐锋,吕建.一种开放环境下的软件可靠性评估方法[J].计算机学报,2010,33(3):11.

[10] LEPPINEN H. Current use of linux in spacecraft flight software[J]. IEEE Aerospace & Electronic Systems Magazine,2017,32(10):4-13.

[11] 高吉星,虞志刚,冯旭,等.面向空间网络的虚拟路由器实现方法[J].中国电子科学研究院学报,2021,16(9):869-876.

[12] 潘恬,李星辰,薛文浩,等.一种基于容器的低轨卫星网络协议测试床[J].计算机学报,2022(9):045.

[13] KIM D, JOE H, KIM H. Implementing space separation functionality into Linux-based spacecraft computer[C]// Digital Avionics Systems Conference. IEEE,2014:253-260.

[14] GOMES J, BAGNASCHI E, CAMPOS I, et al. Enabling rootless Linux Containers in multi-user environments: The udocker, tool[J]. Computer Physics Communications,2018,232(1):84-97.

[15] JIE Z, YANG L, SHU Y, et al. NHPP-based software reliability model considering testing effort and multivariate fault detection rate[J]. Journal of Systems Engineering and Electronics,2016,27(1):260-270.

[16] Ahmed abdelsalam cesare roseti francesco zampognaro. steady-state performance evaluation of linux TCPs versus TCP wave over leaky satellite links [J]. China Communications2017,14(3):17-28.

[17] OKAMYRA H, DOHI T. A novel framework of software reliability evaluation with software reliability growth models and software metrics [C]//IEEE International Symposium on High-assurance Systems Engineering.IEEE,2014.97-104.

[18] YAMADA S.Software reliability modeling: fundamentals and applications [M]. Merlin: Springer Publishing Company,Incorporated,2014.1-38.

[19] PRINCY B A, SRIDHAR S. Measuring software reliability and release time using SRGM t001. Int' 1 Journal of Scientific Research and Education, 2014, 2(5):785-796.

[20] LI X D, YIN Y F, et al. Software reliability analysis considering correlated component failures with coupling measurement framework [J]. Journal of Systems Engineering and Electronics,2015,26(5):1114-1126.

[21] ALLENDE I,GUIRE N M,et al. Towards linux based safety systems-a statistical approach for software execution path coverage [J]. Journal of Systems Architecture,2021,116(1):102047.

[22] 汪玲,马志伟,乔永强,等.航天型号软件工程化要求[S].北京:中国航天科技集团,2013.Q/QJA 30A—2013.

[23] 王凤武.谈LINUX系统优化与安全加固[J].中国设备工程,2022(12):3.130-132.

[24] 李克文,雷永秀,张震涛.基于构件影响因子的软件可靠性评估方法[J].计算机工程与设计,2022,43(1):165-170.

[25] 何敏飞.探索软件测试精细化质量管理之路[J].中国质量,2021(9):109-112.