ICS 35.080

CCS L 77

T/CICC

中 国 指 挥 与 控 制 学 会 团 体 标 准

T/CICC 35011—2025

复杂软件系统环境适应性技术要求

Technical requirements for environmental adaptability of complex software systems

2025-09-12 发布　　2025-09-12 实施

中国指挥与控制学会　发布

目　　次

前言 V
1 范围 1
2 规范性引用文件 1
3 术语与定义 2
4 复杂软件系统环境适应性对象 4
4.1 环境要素 4
4.1.1 运行环境 4
4.1.2 自然环境 4
4.1.3 其它环境 4
4.1.4 综合环境 4
4.2 软件代码 4
4.3 软件文档 4
4.4 软件模型 4
4.5 第三方库 4
4.6 工具链与集成环境 4
4.7 运维环境 4
4.8 外部接口环境 4
5 复杂软件系统环境适应性定量要求 5
5.1 自然环境适应性定量要求 5
5.1.1 环境指标 5
5.1.2 性能指标 5
5.2 运行环境适应性定量要求 6
6 复杂软件系统环境适应性定性要求 8
6.1 自然环境适应性定性要求 8
6.1.1 温度适应性 8
6.1.2 湿度适应性 8
6.1.3 振动及冲击适应性 8
6.1.4 电磁干扰适应性 8
6.1.5 空间辐射适应性 8
6.1.6 位置坐标适应性 8
6.1.7 组件意外断联和失效适应性 9
6.1.8 意外断电适应性 9
6.2 运行环境适应性定性要求 9

6.2.1 适应性要求 9
6.2.2 软件易替换性要求 10
6.2.3 共存性要求 10
6.2.4 易安装性 10
6.2.5 移植完整性 10
6.2.6 验证与确认要求 10
6.3 其它环境适应性要求 11
6.3.1 区域与时间适应性 11
6.3.2 法律和政策适应性 11
6.4 综合环境适应性要求 11
6.5 软件环境适应性测试要求 11
7 复杂软件系统环境适应性技术支撑与方法 11
7.1 软件环境适应性分析方法 11
7.1.1 生命周期环境剖面分析 11
7.1.2 编制使用环境文件 12
7.1.3 确定环境类型及其量值 12
7.1.4 软件环境适应性影响矩阵分析 12
7.2 软件环境适应性设计方法 14
7.2.1 制定软件系统环境适应性设计准则 14
7.2.2 软件系统环境适应性设计 14
7.3 软件环境适应性测试方法 14
7.3.1 环境应力测试 14
7.3.2 环境适应功能测试 16
7.4 软件环境适应性评估方法 17
7.5 软件环境适应性改进方法 18
8 复杂软件系统环境适应性全生命周期过程与活动 18
8.1 需求分析阶段 18
8.1.1 自然环境适应性分析 18
8.1.2 运行环境适应性分析 18
8.1.3 其它环境适应性分析 18
8.2 设计与实现阶段 19
8.2.1 自然环境适应性设计与实现 19
8.2.2 运行环境适应性设计与实现 20
8.2.3 适应性设计 20
8.2.4 易替换性设计工作 21
8.2.5 共存性设计工作 22
8.2.6 易安装性设计 22

8.2.7 移植完整性设计 23
8.2.8 其它环境适应性设计与实现 24
8.2.9 综合环境适应性设计 24
8.3 测试阶段 24
8.3.1 自然环境适应性测试工作 24
8.3.2 运行环境适应性测试工作 24
8.3.3 其它环境适应性测试工作 24
8.3.4 综合环境适应性测试工作 24
8.4 使用阶段 25
8.4.1 自然环境适应性保证工作 25
8.4.2 计算环境适应性保证工作 25
8.4.3 其它环境适应性保证工作 25
8.5 维护与更新阶段 25
参考文献 26

前　言

本文件按照GB/T 1.1-2020《标准化工作导则 第1部分：标准化文件的结构和起草规则》的规定编写。

请注意本文件的某些内容可能涉及专利。本文件的发布机构不承担识别专利的责任。

本文件由中国指挥与控制学会提出并归口。

本文件起草单位：北京航空航天大学、杭州市北京航空航天大学国际创新研究院、可靠性与环境工程技术重点实验室、北京航空航天大学可靠性工程研究所、中国科学院声学研究所、中国船舶集团有限公司系统工程研究院。

本文件主要起草人：杨顺昆、侯展意、郝程鹏、吴梦丹、颜思玮、段峙宇、曾福萍、廖力鸣、张北、刘磊、范珈铭、马欣瑞、王丽、戴之光。

复杂软件系统环境适应性技术要求

1 范围

本文件规定了复杂软件系统环境适应性复杂软件系统环境适应性的应用对象、定量和定性要求、支撑技术与方法以及全生命周期的过程与活动。

本文件适用于复杂软件系统在需求分析、设计与实现、测试、使用和维护更新阶段的环境适应性保证，可供相关行业组织和研究机构使用。

2 规范性引用文件

下列文件中的内容通过文中的规范性引用而构成本文件必不可少的条款。其中，注日期的引用文件，仅该日期对应的版本适用于本文件；不注日期的引用文件，其最新版本(包括所有的修改单)适用于本文件。

GB/T 4797-2018	环境条件分类 自然环境条件
GB/T 8566-2022	系统与软件工程 软件生存周期过程
GB/T 14394-2008	计算机软件可靠性和环境适应性管理
GB/T 29833.1-2013	系统与软件可移植性 第1部分：指标体系
GB/T 29833.2-2013	系统与软件可移植性 第2部分：度量方法
GB/T 29833.3-2013	系统与软件可移植性 第3部分：测试方法
GB/T 34986-2017	产品加速试验方法
GJB 438C-2021	军用软件开发文档通用要求
GJB 4239A-2022	装备环境工程通用要求
T/CICC 35008-2025	复杂软件系统可靠性技术要求
IEC-60721-2020	环境条件分类（Classification of environmental conditions）
RTCA DO-160G-2010	机载设备环境条件与测试程序（Environmental conditions and test procedures for airborne equipment）

3 术语与定义

GB/T 29833.1-2013、GJB/Z-161-2012、GJB 4239A-2022和T/CICC 35008-2025确立的以及下列术语和定义适用于本文件。

3.1

复杂软件系统 complex software system

由大量相互依赖、相互作用的软件组件（计算机程序、模块、服务等）通过复杂的逻辑和物理关系连接而成，并遵循严格的规程（流程、协议、策略）进行协同运作，需动态应对内外部软件、硬件与环境的变化以实现复杂业务或关键领域目标的软件集成。其本质特征在于结构庞大、功能多样、环境多变、动态交互、任务复杂，通常具备多层次架构、模块协作、高度耦合以及较长的生命周期。

［来源：T/CICC 35008-2025，3.1］

3.2

环境适应性 environmental adaptability

软件在其生命周期内可能遇到的各种环境的作用下能实现其所有预定功能和性能和（或）不被破坏的能力。

［来源：GJB 4239A-2022，3.7，有修改］

3.3

运行环境适应性 operational environmental adaptability

软件在其生命周期内可能遇到的各种硬件、软件、配置项等环境要素的作用下能实现其所有预定功能和性能和（或）不被破坏的能力。

3.4

自然环境适应性 natural environmental adaptability

软件在其生命周期内可能遇到的各种温湿度、振动等非人为环境要素的作用下能实现其所有预定功能和性能和（或）不被破坏的能力。

3.5

环境剖面 environmental profile

软件系统使用中可能面临有关事件和条件的时间历程。

［来源：GJB 4239A-2022，3.5，有修改］

3.6

可移植性 portability

系统或软件能适应的软件或者硬件环境的能力。

［来源：GB/T 29833.1-2013，3.2，有修改］

3.7

组织环境适应性 organizational environment adaptability

系统与软件对运行环境的适应能力，涉及用户组织的业务运行环境。

［来源：GB/T 29833.1-2013，6，有修改］

3.8

通信适应性 communication adaptability

软件系统对传输模式调整的适应能力。

［来源：GB/T 29833.1-2013，6］

3.9

数据适应性 data adaptability

软件系统试图适应于规定的环境，其所使用的相关数据在规定环境下使用的完备程度。

［来源：GB/T 29833.1-2013，6］

3.10

安装正确性 installation correctness

遵循有效的安装指导，在安装完成后，软件系统能否正确的在试图适应的规定环境中运行。

［来源：GB/T 29833.1-2013，8］

3.11

安装影响性 installation impact

遵循有效的安装指导，软件系统的安装过程是否会影响到其它软件或设备的正常运行，或其它运行的软件或者设备是否会影响到安装过程的进行。

［来源：GB/T 29833.1-2013，8］

3.12

移植正确性 porting correctness

在移植之后,软件产品的被检测功能是否能够被正确无误的执行。

［来源：GB/T 29833.1-2013，9］

3.13

移植一致性 porting consistency

在移植之后，软件产品的被检测功能是否仍与在移植之前保持相同的操作步骤或者使用相同的执行流程。用户能否适应功能操作的变化。

［来源：GB/T 29833.1-2013，9］

4 复杂软件系统环境适应性对象

4.1 环境要素

环境要素需分类定义其对软件系统的影响机制及适配要求，根据GB/T 4797和GB/T 14394标准的相应规定，本文件将环境要素划分为运行环境、自然环境和其它环境。

4.1.1 运行环境

复杂软件系统应分别明确其适用的运行环境，包括CPU、内存、存储、接口、网络等与计算相关的资源条件和要求。

4.1.2 自然环境

复杂软件系统应分别明确其适用的自然环境。综合GB/T 4797、RTCA DO-160G-2010、IEC 60721-2020等标准，本文件将自然环境定义为包含温度、湿度、振动、电磁干扰和空间辐射等因素，并考虑两种自然环境可能间接导致的组件失效和意外断电因素。

4.1.3 其它环境

复杂软件系统需明确非计算及自然环境要素的适配要求，包括区域要求、法律和行政要求等。

4.1.4 综合环境

复杂软件系统需评估多环境要素交互作用下的整体适应能力，涵盖动态环境变化及复合场景的适配性要求。

4.2 软件代码

软件代码需开展环境适应性设计，确保在不同环境条件下稳定运行。

4.3 软件文档

软件文档在符合GJB 438-C等标准规范编写的基础上，还需明确环境适应性要求并规定环境适应性保证的具体工作方法，为软件设计、开发、使用和维护中的环境适应性保障提供依据。

4.4 软件模型

针对基于模型的软件开发方法设计与开发的软件模型，包含需求、架构和设计模型等，需具备环境适应性设计，确保在不同环境条件下稳定运行。

4.5 第三方库

第三方库是复杂软件系统的重要组成部分，其环境适应性直接影响系统的稳定性和兼容性。软件系统需明确第三方库的运行平台、操作系统和版本等兼容性要求，避免依赖冲突，并定期根据第三方库新版本的调整发布补丁或更新。

4.6 工具链与集成环境

编码、编译、测试和发布所使用的全套工具链需支持跨环境一致性，应尽可能确保开发、测试与生产环境中软件的行为一致；自动化测试与集成环境等需模拟目标部署环境，确保软件在各阶段的一致性。

4.7 运维环境

监控系统、日志分析平台等持续运维环境需实时感知环境变化并触发环境适应策略。

4.8 外部接口环境

API、SDK、硬件驱动等外部接口需适配多样化的硬件架构、操作系统、编程语言和依赖库等调用环境，支持多种规定的数据格式，确保互操作性。

5 复杂软件系统环境适应性定量要求

5.1 自然环境适应性定量要求

自然环境类定量要求如表1所示，计算所需的环境条件指标和性能指标在以下各节规定。

表 1 自然环境类定量要求

序号	指标	含义	计算方法	来源	备注
1.	环境风险极值	采用一定时间风险率统计得到的某一环境因素在一定时期内的最高值或最低值,是环境适应性设计的重要依据。	$ERE=\min_{T} I_E$ 或 $ERE=\max_{T} I_E$ T：统计时间； I_E：某一环境因素强度。	GJB 451B-2021	需求分析阶段使用
2.	环境损伤率	软件系统及在一定的环境条件下使用一定时间后性能变化量与初始值之比。	$A_{ED}=\frac{F_E-F_S}{F_S}\times 100\%$ F_E：软件在一定的环境条件下使用一定时间后性能； F_S：软件的原始性能。	GJB 451B-2021	设计与实现阶段使用
3.	环境损伤速率	软件系统在一定的环境条件下性能随时间变化的快慢程度,可采用单位时间内性能变化量来表示。	$R_{ED}=\frac{\Delta F(t)}{\Delta T}$ F(t)：软件随时间变化的性能； T：时间。	GJB 451B-2021	设计与实现阶段使用
4	环境耐受时间	软件系统在预期使用环境下性能和功能变化达到产品规范允许的影响程度或变化范围(容差)前的使用时间。	$T_{EE}=t_{UE}-t_{US}$ t_{UE}：软件在预期使用环境下性能和功能变化达到容差的时刻； t_{US}：软件使用开始时刻。	GJB 451B-2021	需求分析阶段使用
5.	系统可用度	软件系统在外部环境干扰下仍能处于可运行状态的概率	$X=\frac{A}{B}$ A =软件系统正常运行时间 B =软件系统总运行时间	GB/T 29833-2013	0≤X≤1，越接近1越好
6.	任务成功率	在环境变化条件下,软件系统任务成功完成的比例	$X=\frac{A}{B}$ A =软件系统成功运行的任务数 B =软件系统能适应且正确运行的任务总数	GB/T 29833-2013	0≤X≤1，越接近1越好

5.1.1 环境指标

自然环境指标用于度量本文件中规定的自然干扰因素，如温度、湿度、振动等，其中各种干扰因素度量需使用国际单位制；若使用特殊单位制，则研发团队内部必须统一。

5.1.2 性能指标

表1中建议选取的性能指标如下，其中正向指标可直接使用指标数值p，负向指标需要对指标数值p取倒数（1/p）或补数（1-p）后进行运算。

5.1.2.1 功能正确性指标

正向指标：单元测试覆盖率、功能测试通过率等；

负向指标：测试用例失败率等。

5.1.2.2 容错能力指标

正向指标；冗余备份切换成功率、故障自愈率等；

负向指标：平均无故障时间（MTBF）的、故障恢复时间（MTTR）的等。

5.1.2.3 速度指标

正向指标：吞吐量、数据处理速率、并发用户数量等；

负向指标：单次请求的处理延迟、平均响应时间、CPU/内存占用率、百分位延迟等。

5.2 运行环境适应性定量要求

运行环境适应性定量指标计算方法主要参考GB/T 29833-2013，如表2所示。

表 2 运行环境类定量要求

序号	类别	指标	含义	计算方法	来源	备注
1.	适应性	操作系统适应性	系统与软件对各种操作系统的适应能力。对于操作系统的适应性，需综合考虑其它相关度量特性	$X=\frac{A}{B}$ A=系统与软件能够适应的硬件数量 B=期望系统和软件能适应且正确运行的硬件环境类型的总数	GB/T 29833	$0 \le X \le 1$，越接近1越好。
2.	适应性	数据库适应性	在软件运行的新环境中数据库软件对软件正确运行的影响	$X=\frac{A}{B}$ A=系统和软件能成功适应的数据库个数 B=期望系统与软件能成功适应的数据库个数	GB/T 29833	$0 \le X \le 1$，越接近1越好。
3.	适应性	数据库适应性	在软件运行的新环境中数据库软件对软件正确运行的影响	$X=\frac{A}{B}$ A=系统和软件能成功适应的数据库个数 B=期望系统与软件能成功适应的数据库个数	GB/T 29833	$0 \le X \le 1$，越接近1越好。
4.	适应性	支撑软件适应性	在软件运行的新环境中数据库软件对软件正确运行的影响	$X=\frac{A}{B}$ A=系统和软件能成功适应的支撑软件个数 B=期望系统与软件能成功适应的支撑软件个数	GB/T 29833	$0 \le X \le 1$，越接近1越好。
5.	适应性	有效软件共存性	在移植之后,是否会影响到移植软件产品的正确使用	$X=\frac{A}{B}$ A=共同运行时,能让目标软件正常运行的软件个数 B=希望能与目标软件共存的软件个数	GB/T 29833	$0 \le X \le 1$，越接近1越好。

表 2　运行环境类定量要求计算方法（续）

序号	类别	指标	含义	计算方法	来源	备注
6.	适应性	组织环境的适应性	系统与软件对于环境的适应能力	$X=1-\frac{A}{B}$ A=在用户的业务环节中运行测试期间没有完成任务或不足以使任务满足适当级别的运行的功能数 B=运行测试的功能总数	GB/T 29833	$0 \le X \le 1$，越接近1越好。
7.	适应性	通信适应性	系统与软件对传输模式调整的适应能力	$X=\frac{A}{B}$ A=用户能够成功适应的通信模式的数量 B=用户期望能够适应的通信模式的数量	GB/T 29833	$0 \le X \le 1$，越接近1越好。
8.	适应性	数据适应性	系统与软件对数据变化的适应能力	$X=\frac{A}{B}$ A=应成功或者运行并被观察到的数据种类数 B=期望能在软件适应的环境中运行的数据科类总数	GB/T 29833	$0 \le X \le 1$，越接近1越好。
9.	易替换性	数据的连续使用	系统与软件在变更之后是否能继续使用相同的数据	$X=\frac{A}{B}$ A=在其它更换的软件中使用并证实能继续使用的数据个数 B=在其它更换的软件中使用并计划能继续重新使用的数据个数	GB/T 29833	$0 \le X \le 1$，越接近1越好。
10.	易替换性	功能的内含性	系统与软件在变更之后是否能继续使用相同的数据	$X=\frac{A}{B}$ A=在新版软件中产生类似结果而无需变更的功能数 B=由要更换的其它系统与软件提供的有类似功能并已测试过的功能数	GB/T 29833	$0 \le X \le 1$，越接近1越好。
11.	易安装性	安装正确性	在移植过程中,遵循有效的安装指导,在安装完成后,软件产品是否能顺利运行	$X=\frac{A}{B}$ A=软件在新环境下安装成功的次数 B=软件在新环境下安装的总次数	GB/T 29833	$0 \le X \le 1$，越接近1越好。
12.	易安装性	安装影响性	在移植过程中,软件安装过程中或安装完成后是否会影响到其它软件或者环境的运行及设置	$X=\frac{A}{B}$ A=在安装过程中以及安装完毕后，运行状态受到影响的软件数量 B=在该环境下运行的软件总数	GB/T 29833	$0 \le X \le 1$，越接近0越好。
13.	易安装性	安装难易性	在移植过程中,软件产品的安装步骤是否能够通过简易的用户操作来实现	$X=\frac{A}{B}$ A=在安装过程中以及安装完毕后，运行状态受到影响的软件数量 B=在安装过程中，总共需要的操作步骤次数	GB/T 29833	$0 \le X \le 1$，越接近0越好。

表 2　运行环境类定量要求计算方法（续）

序号	类别	指标	含义	计算方法	来源	备注
14.	易安装性	安装灵活性	在移植过程中，软件是否可以定制安装	$X=\frac{A}{B}$ A=在安装过程中，可由用户进行定制的步骤次数 B=在安装过程中，总共需要的操作步骤次数	GB/T 29833	0 ≤ X ≤ 1，越接近1越好。
15.	移植完整性	移植正确性	在移植之后，软件产品的被检测功能是否能够被正确无误的执行	$X=1-\frac{A}{B}$ A=测试或验证不符合的功能点个数 B=需测试或验证的功能点个数	GB/T 29833	0 ≤ X ≤ 1，越接近1越好。
16.	移植完整性	移植一致性	在移植之后,软件产品的被检测功能是否仍与在移植之前保持相同的操作步骤或者使用相同的执行流程。用户能否适应功能操作的变化	$X=1-\frac{A}{B}$ A=在测试过程中,当移植后,功能点执行的步骤或者流程与之前的操作不一致,可能影响用户的使用和理解,或者用户认为无法接受的操作调整,诸如此类的功能点的个数 B=在测试过程中,需要测试或验证一致性的功能点个数	GB/T 29833	0 ≤ X ≤ 1，越接近1越好。

6　复杂软件系统环境适应性定性要求

6.1　自然环境适应性定性要求

6.1.1　温度适应性

应明确软件系统可正常工作的最高温度及最低温度条件。对温度变化率敏感的软件需规定温度变化率的阈值。可采用高低温循环试验对软件系统的温度适应性进行试验验证。

6.1.2　湿度适应性

应明确软件系统可正常工作的湿度条件，确保其在高湿条件散热能力降低时依然达成规定的功能要求；在低湿度条件下，不受静电干扰等因素的影响。

6.1.3　振动及冲击适应性

应明确软件系统可正常工作的振动和冲击条件，避免软件系统受到振动影响而导致功能无法正常实现。对于运行在振动和冲击环境下的软件，需采取措施降低或消除振动和冲击对数据、图像、声音等信号采集的干扰，同时需对振动和冲击可能发生的硬件失效、接插件松动等问题进行防护，提升数据质量、避免软件崩溃或数据丢失。

6.1.4　电磁干扰适应性

对于运行环境可能存在强电磁干扰的软件，所有涉及电信号及电磁波信号的数据传输与通信逻辑均需对潜在的电磁干扰进行防护，保证软件正常运行，避免数据传输错误、软件崩溃和数据丢失。

6.1.5　空间辐射适应性

对于可能受到空间辐射等因素影响的软件系统，如运行在深空探测系统中的软件，需在软件层面开展空间辐射适应性设计防护，避免或降低空间辐射造成比特翻转所带来的软件系统数据流和控制流异常。

6.1.6　位置坐标适应性

对于具备位置坐标（经纬度、距离和高度等）处理功能的软件系统，需对异常或特殊的坐标数据进行识别和处理，避免异常或特殊位置坐标数据处理不当导致软件工作异常。

6.1.7 组件意外断联和失效适应性

软件系统应针对软件系统组件意外断连和失效的异常情况进行防护，非必需组件失效（传感器、伺服器等）时程序不应崩溃，必需组件（电源、CPU、内存等）失效时程序不应丢失关键运行数据。

6.1.8 意外断电适应性

软件系统应避免意外断电造成的数据丢失、损坏或程序执行紊乱，且供电恢复后应支持恢复到正常的工作状态。

6.2 运行环境适应性定性要求

6.2.1 适应性要求

6.2.1.1 通用要求

运行环境适应性通用要求如下：

a） 软件应在规定的运行环境条件下满足全部功能和性能要求；

b） 软件设计应考虑运行环境可能发生的变化，确保在变化后仍可维持预期功能；

c） 应避免与特定运行环境的过度耦合，采用标准化接口和可替换的模块设计；

d） 应在设计阶段明确运行环境的适配范围，并建立相应的验证计划。

6.2.1.2 硬件适配性

软件应在源代码级别适配设计要求内的多种硬件，并预留设计要求外其它平台的扩展能力。

6.2.1.3 操作系统适应性

软件应在源代码级别适配设计要求内的多种操作系统，操作系统升级和补丁不应影响软件的核心功能。

6.2.1.4 数据库与支撑软件适应性

软件的数据库与支撑软件适应性如下：

a） 软件的可执行文件应适配设计要求内的多种数据库及其它支撑软件，可执行文件适配困难的，可在源代码级别进行适配；

b） 软件应支持跨不同版本数据库和支撑软件的数据迁移；

c） 软件的任何发布版本均应明确数据库和支撑软件的版本范围。

6.2.1.5 组织环境的适应性

软件的可执行文件应适用于多种类型用户组织的业务运行环境和工作流程。

6.2.1.6 通信与网络适应性

软件应支持多种规定的通信模式和网络类型，并对通信和网络的不良工作状态进行处理，避免通信异常造成软件时序错误、卡死、崩溃或数据丢失。

6.2.1.7 数据与接口适应性

软件应支持多种规定的数据与接口类型，并对数据转换过程中异常情况进行处理，避免数据异常造成软件错误、崩溃或数据丢失。

6.2.1.8 并发适应性

针对可能被并发调用的程序或程序片段，应对共享资源进行保护，避免数据读写冲突。

6.2.2 软件易替换性要求

6.2.2.1 数据的连续使用性

软件产品在替换同一环境中相同用途的旧版本或其它同类产品时，应确保用户或维护者能够访问和操作所有原有数据。新版本软件应完全兼容旧版本的输入输出数据文件，并保留所有功能特性。

6.2.2.2 功能的内含性

替换后的软件产品应保留或增强旧版本的核心功能，且操作逻辑保持一致，不应在未告知用户的情况下直接移除旧版本核心功能或改变核心操作逻辑。

6.2.3 共存性要求

6.2.3.1 资源共享性

软件应支持与其它软件在同一计算资源下安装和运行，应按需申请资源并及时释放，系统资源不足时应优先暂停当前任务并提示用户，不应崩溃或丢失数据。

6.2.3.2 功能兼容性

软件应提供标准化接口，确保与其它系统或模块的交互顺畅，避免因接口不匹配导致的功能异常。

6.2.3.3 环境兼容性

软件应支持与其它软件在同一环境下安装和运行，不应随意修改操作系统或其它软件的配置及数据，

6.2.3.4 用户界面兼容性

软件在与其它应用同时运行时，用户界面不应产生视觉、交互或焦点上的冲突，确保用户能在多任务环境下高效、无干扰地操作所有软件。

6.2.4 易安装性

6.2.4.1 安装正确性

软件在安装过程中须准确无误地完成所有必要步骤，确保安装后的软件功能完整且运行正常。

6.2.4.2 低安装影响

软件安装过程中应尽量不影响系统服务和其它应用的正常运行。

6.2.4.3 安装简便性

软件须尽量降低安装过程中的操作复杂度和学习成本。

6.2.4.4 安装灵活性

软件的安装过程应适应不同用户需求和系统环境。

6.2.4.5 安装效率要求

软件应尽量压缩安装文件大小并加快安装速度，减少安装总时间。

6.2.5 移植完整性

6.2.5.1 移植正确性

软件在移植过程中须能准确无误地完成所有必要的迁移步骤，确保移植后的软件功能完整且运行正常。

6.2.5.2 移植一致性

软件在移植后应能够保持与原环境中的行为和表现一致，并确保用户在使用过程中不会感受到明显差异。

6.2.6 验证与确认要求

需组织运行环境适应性专项测试，覆盖硬件、操作系统、数据库、支撑软件、网络等，确保软件的运行环境适应性符合要求。

6.3 其它环境适应性要求

6.3.1 区域与时间适应性

对于在全球多个国家或地区部署的软件，应适配不同国家或地区的时间、语言、度量衡等差异因素。

6.3.2 法律和政策适应性

软件应严格遵守和尊重国际和当地的法律法规、标准规范以及公序良俗，避免引发社会争议。

6.4 综合环境适应性要求

复杂软件系统应在多种环境应力同时作用条件下（如高温叠加高湿、电磁干扰叠加振动、高温叠加低可用CPU资源等）保持规定的核心功能的可用性。为此，应在环境适应性设计防护的基础上，开展典型组合环境测试，验证系统在复合环境下的稳定性与鲁棒性；此外，宜在极端叠加条件下进行加严测试，评估系统在复杂环境下的极限适应能力。

6.5 软件环境适应性测试要求

复杂软件系统环境适应性的测试应覆盖自然环境适应性、运行环境适应性以及其它环境适应性的全面测试，保证软件的环境适应性符合要求。

7 复杂软件系统环境适应性技术支撑与方法

7.1 软件环境适应性分析方法

本节阐述的分析方法，通过识别生命周期环境剖面和量化环境-功能-任务影响矩阵两个核心步骤，系统性地评估各类环境因素对软件的综合影响，旨在为软件环境适应性的针对性设计、测试与评估提供决策支持。

7.1.1 生命周期环境剖面分析

软件系统的状态包括需求分析、设计与实现、测试、使用以及维护和更新五种。确定软件系统生命周期剖面时，根据生命周期环境剖面的状态事件预计的发生地点和特征，结合环境应力产生的机理，确定各种事件可能遇到或产生的运行环境和诱发环境类型，并按时序列出。

在环境剖面分析前，需明确以下事项：

a） 软件使用方案和保障方案；

b） 有关相同或相似软件或平台环境特性及环境测量数据；

c） 生命周期环境剖面的提交进度和程序（必须确定的事项）；

d） 特殊条件或局限性说明。

在软件系统领域，生命周期环境剖面应特别考虑以下事项：

a） 环境数据采集：传感器精度变化、失效模式、外部干扰（如光照变化、噪声、遮挡等）；

b） 数据与信息环境：数据分布漂移、数据质量退化、多源异构数据同步性；

c） 程序算法运行环境：软件计算延迟、计算平台变化（CPU/边缘计算/FPGA）、资源调度策略等；

d） 网络与通信环境：带宽波动、链路切换、延迟抖动、数据丢包；

e） 安全威胁环境：敏感信息窃取、勒索病毒等。

应尽可能用文字、图表和统计特性说明生命周期环境剖面中各种环境应力大小。这些应力值应来自

平台测量结果，或参考相关标准、数据库、仿真结果，必要时可通过分析计算获得。

7.1.2 编制使用环境文件

通过收集软件系统的实际环境数据，形成使用环境文件，为确定环境适应性要求提供依据。环境数据收集应包括但不限于：

a） 实际部署场景的自然环境数据特征（温湿度、振动和冲击、分辨率、帧率、噪声水平等）；

b） 任务场景下的网络条件与计算资源占用情况；

c） 不同外部条件下的算法精度与响应时间；

d） 平台运行日志、异常事件记录及维护历史。

当无法从历史数据库或类似系统获取足量数据时，应制定环境数据实测计划，包括：

a） 测量数据类型（传感器原始数据、计算延迟、能耗等）及精度要求；

b） 测试位置与任务剖面（如无人机飞行剖面、无人车行驶路线等）；

c） 采集工具与设备（数据采集卡、网络监测模块、功耗监测设备等）；

d） 数据处理与归纳方法（统计分析、数据清洗、特征提取等）。

应将收集和实测得到的数据进行处理，形成使用环境文件。

7.1.3 确定环境类型及其量值

系统分析各种环境对软件系统性能、安全性及可靠性等的影响，确定关键因素，同时要求确定需要考虑的环境种类及其量值或确定这些量值采用的准则，从而得到环境要求，包括但不限于：

a) 自然环境量值：温度、湿度、电磁干扰、振动等；

b) 数据与信息环境量值：数据漂移幅度、噪声水平、缺失率等；

c) 算法与模型运行环境量值：计算延迟阈值、内存占用上限、模型精度下限；

d) 网络环境量值：最大可接受延迟、带宽下限、丢包率阈值；

e) 安全威胁量值：可能的 DDOS 攻击请求频率和流量等。

7.1.4 软件环境适应性影响矩阵分析

基于复杂软件系统的软硬件结构、系统功能、不同任务类型与使用环境条件，构建环境适应性影响矩阵，分析自然环境因素、运行环境因素和其它环境因素对软件系统的影响，对软件系统环境适应性进行重要性排序，作为软件系统环境适应性设计、测试与评估的基础。

软件环境适应性影响矩阵包含结构与环境要素矩阵、功能与结构要素矩阵、任务和功能要素矩阵、功能与环境要素矩阵以及任务与环境要素矩阵五个矩阵。前三者分别描述结构与环境、功能与结构、任务与功能要素之间的关系，通过专家打分得到，并可采用层次分析法提高打分的准确性；后两者基于前三者计算得到，可反映环境要素直接对功能和任务的影响，作为分析的结果输出。

结构和环境要素矩阵$M_{Stru-Env}$为M行N列的矩阵，列含义为N种软件及硬件系统所涉及到的结构要素，如CPU、内存、机械硬盘等；行含义为M种环境要素，如高温、高湿、振动等。矩阵中每一个数值代表结构和环境要素的严重程度分级，取值分别为0-5，分别表示无影响、较低影响、中低影响、中影响、较高影响和高影响，结构如表3所示。

表 3　结构和环境要素矩阵

环境要素	结构要素				
	结构要素1	结构要素2	结构要素3	...	结构要素N
环境要素1					
环境要素2					
环境要素3					
...					
环境要素M					

功能和结构要素矩阵$M_{Func-Stru}$为N行P列矩阵，行含义为N种结构要素，与$M_{Stru-Env}$中的结构要素含义相同；列含义为P种系统功能，如计算功能、存储功能等。矩阵中数值表示功能要素与结构要素的相关程度，分级为0到5，分别表示不相关、低相关、中低相关、中等相关、中高相关和高相关，矩阵结构如表4所示。

表 4　功能和结构要素矩阵

结构要素	功能要素			
	功能要素1	功能要素2	...	功能要素P
结构要素1				
结构要素2				
结构要素3				
...				
结构要素N				

任务和功能要素矩阵$M_{Task-Func}$为P行Q列矩阵，行含义为P种系统功能要素，列含义为Q种系统中的任务要素，元素含义为任务要素对该功能要素的依赖程度，分级为0到5，分别为不依赖、少量依赖、较少依赖、中等依赖、较高依赖和高依赖，如表5所示。

表 5　任务和功能要素矩阵

功能要素	任务要素		
	任务要素1	...	任务要素Q
功能要素1			
功能要素2			
...			
功能要素P			

功能和环境要素矩阵$M_{Func-Env}$的值为$Norm5(M_{Stru-Env} \times M_{Func-Stru})$，为M行P列矩阵，其中Norm5指的是将矩阵的所有元素线性归一化后，将每一个元素乘5，因此元素的范围是$[0, 5]$之间的实数。矩阵元素数值的含义为环境要素对功能要素的影响程度高低，如表6所示。

表 6　功能和环境要素矩阵

环境要素	功能要素			
	功能要素1	功能要素2	...	功能要素P
环境要素1				
环境要素2				
环境要素3				
...				
环境要素M				

任务和环境要素矩阵$M_{Task-Env}$的值为$Norm5(M_{Func-Env} \times M_{Task-Func})$，为M行Q列矩阵，矩阵中元素的含义为环境要素对任务要素的影响程度高低，范围是$[0, 5]$之间的实数，如表7所示。

表 7　任务和环境要素矩阵

环境要素	任务要素		
	任务要素1	...	任务要素Q
环境要素1			
环境要素2			
环境要素3			
...			
环境要素M			

7.2　软件环境适应性设计方法

7.2.1　制定软件系统环境适应性设计准则

应基于相关标准、行业经验与软件系统特性，制定环境适应性设计准则，供设计人员使用，并作为设计评审的依据。根据软件系统的不同，设计准则可适当裁剪，降低环境适应性设计和评审的成本。

7.2.2　软件系统环境适应性设计

在软件设计过程中应根据环境适应性设计准则，采用适当的技术和设计策略，使软件达到规定的环境适应性水平。此外，环境适应性设计应与自然环境测试、运行环境测试、其它环境测试以及综合环境适应性测试紧密结合，充分利用测试信息，对发现的环境适应性薄弱环节采取设计措施加以纠正。

7.3　软件环境适应性测试方法

根据软件环境适应性分析的结果，依据重要性排序，针对相应的模块、功能和任务，施加相应的环境干扰因素，开展软件环境适应性单项测试与组合测试，验证软件系统在规定环境下的功能及性能是否满足设计要求。

7.3.1　环境应力测试

环境应力测试方法参考GB/T 34986-2017的规定制定，为在不改变失效模式和失效机理的前提下，对复杂软件系统施加规定的步进、循环等环境应力，直到其发生故障或达到要求的应力水平。

本文件中的试验应力包含：温度、湿度、振动和冲击、静电放电、空间辐射、位置坐标、组件失效断联和意外断电等自然环境应力，以及CPU占用率、内存占用率、并发量等运行环境应力。

7.3.1.1　步进应力测试

7.3.1.1.1　定义及适用对象

软件步进应力测试是一种逐步增加环境应力强度（如温度、湿度、电压、振动等），以暴露软件系统在极端条件下的潜在缺陷和失效模式，并给出实测失效阈值的环境适应性测试方法，广泛使用于各种软件系统的环境适应性测试。

7.3.1.1.2　步进应力测试基本流程

软件步进应力测试流程分为四个阶段：

a） 测试准备：调研产品环境、硬件平台特性和软件需求，识别薄弱环节并设置监测点；

b） 应力设计：确定初始应力（如40℃）、步长（如10℃）、每步持续时间（如30分钟）及终止条件（如85℃或功能失效）；

c） 测试实施：监控功能、性能、资源使用及错误日志，记录环境参数和异常现象；

d） 结果分析：评估工作极限与破坏极限，分析失效模式并建立可靠性模型。

7.3.1.1.3 步进应力分类

步进应力测试中，步进应力可分为以下类型：

a） 温度步进测试：通过逐步升高或降低环境温度；

b） 湿度步进测试：逐步增加或降低湿度水平；

c） 机械应力步进测试：以逐步增强的振动或冲击模拟物理扰动；

d） 特殊环境步进测试（盐雾/辐照/低气压）：针对盐雾、辐射或低气压等极端条件，设计步进测试；

e） 硬件断联步进：逐个断开多个相同功能或不同功能的硬件；

f） 供电切断步进：逐步降低供电能力，直至系统停止工作；

g） CPU占用率步进：逐步增加计算负载；

h） 内存压力步进：通过持续消耗可用内存；

i） 存储I/O步进：逐步加大磁盘读写压力；

j） 网络与并发步进：逐步增加网络负载和用户并发数；

k） 复合应力步进：组合温度-湿度、温度-振动等多重应力，模拟真实环境中多因素交互导致的复杂失效模式。

7.3.1.2 循环应力测试

7.3.1.2.1 定义及适用对象

本测试方法适用于需承受复杂环境应力循环变化的军用电子设备软件、工业控制系统软件及通信设备软件。通过模拟交变、波动的循环应力条件，快速暴露系统在长期服役中可能出现的资源泄漏、性能退化及环境适应性缺陷。

7.3.1.2.2 实施步骤

软件系统的循环应力测试需遵循以下流程：

a） 测试准备：明确软件运行环境应力类型及其范围，并配置监控工具；

b） 环境配置：在受控环境中构建目标场景（如高温高湿、资源受限）并记录软件初始状态；

c） 应力施加：采用交变式循环方案施加应力，每轮循环后执行功能验证；

d） 实时监控：监测进程稳定性、资源泄漏率及错误率等关键指标；

e） 结果分析：关联故障与环境条件的因果关系，修复后重复特定循环验证优化效果。

7.3.1.2.3 循环应力分类

循环应力测试中，循环应力分为如下类别：

a） 温度及湿度循环变化；

b） 振动能量及冲击的加速度循环变化；

c） 静电电位循环变化；

d） 空间辐射强度循环变化；

e） 位置信息循环变化；

f） 外设连接时断时续；

g） 反复断电与上电测试；

h） CPU负载、内存占用率循环变化；

i） 并发量循环变化；

j） 存储压力循环变化；

k） 网络或其它通信方式的时延、带宽和背景流量等循环变化。

7.3.1.3 加速应力测试

7.3.1.3.1 定义及适用对象

本测试方法适用于需要在恶劣环境下长期运行的软件系统。通过短时间内施加远超正常运行强度的负载，快速发现系统在长期运行后才会暴露的隐性缺陷，如资源泄漏、性能衰减、并发竞争及系统恢复能力不足等问题。

7.3.1.3.2 实施步骤

该技术应按照如下步骤实施：

a） 明确测试目标，确定关键测试对象；

b） 设计测试方案：

1）时间压缩加速测试：分析用户操作剖面，设计高频执行的测试场景和用例；

2）应力增强加速测试：识别关键应力因素（如CPU、内存、IO、网络、并发用户数），设计应力加载曲线，配置极限环境参数；

3）故障注入加速测试：识别关键故障点，设计注入策略和注入点，准备故障注入工具。

c） 在受控的测试环境中执行设计好的测试方案，全面监控并采集数据；

d） 进行根因分析；

e） 对于应力增强和故障注入测试，验证在应力解除或故障清除后，系统能否自动恢复正常服务；

f） 计算加速因子，生成测试报告。

7.3.1.3.3 实施要点

在实施过程中，应注意如下要点：

a） 加速应力测试应在隔离、可控的测试环境中进行；

b） 测试设计应基于真实的运行剖面；

c） 应有紧急中止和快速恢复的预案。

7.3.2 环境适应功能测试

7.3.2.1 适应性与移植测试

适应性与移植测试验证软件在不同环境（硬件、操作系统、数据库等）中的兼容性和功能完整性，具体方法包括：

a） 硬件环境适应性测试：在多种硬件配置（如不同CPU架构、内存大小、显卡性能）下运行软件，验证功能是否符合要求，且不同硬件下软件行为一致；

b） 操作系统适应性测试：在不同操作系统的不同版本中测试安装与运行，验证软件执行的正确性及不同环境下行为的一致性；

c） 数据库及支撑软件：测试软件在规定的不同数据库及浏览器、办公软件等支撑软件环境下执行的正确性以及一致性。

7.3.2.2 软件安装测试

安装测试用于验证软件安装过程的正确性、完整性及结果符合性，具体流程包括：

a） 安装前检查：核对安装文档的完整性、安装介质可用性及系统环境兼容性，确保满足安装前提条件；

b） 安装过程验证：测试不同操作系统环境下安装流程的可行性，验证安装选项组合（如典型/自定义安装）是否符合设计要求，并检查安装中断、回滚等异常处理的可靠性；

c） 安装后确认：检查安装目录、文件结构及注册表项是否与设计一致，验证软件安装后能否正常启动运行，核心功能是否完整可用。

7.3.2.3 软件替换测试

替换测试验证软件版本升级或替换过程的平滑性及数据一致性，具体方法包括：

a） 替换前准备：备份旧版本数据及配置，验证新版本安装包的完整性和兼容性，确保替换操作可回退；

b） 替换过程测试：执行旧版本卸载与新版本安装的连续操作，验证替换流程的自动化程度及中断恢复能力，检查配置文件、用户数据的迁移准确性；

c） 替换后验证：确认新版本功能符合需求，历史数据完整可用，旧版本残留的无效数据被彻底清理，且系统资源占用恢复正常。

7.3.2.4 软件共存测试

共存测试验证多软件同时运行时的资源竞争隔离及功能互不影响性，具体方法包括：

a） 资源冲突检测：同时运行目标软件与其它常用应用，监测 CPU、内存、端口等资源占用是否冲突，系统响应是否延迟；

b） 功能独立性验证：在多软件共存环境下，分别执行各软件的核心功能，确认功能逻辑无干扰，数据读写不互相覆盖或锁定；

c） 环境稳定性测试：长时间运行多软件组合，检查系统是否崩溃、进程异常终止或出现内存泄漏，确保整体环境持续稳定。

7.3.2.5 其它环境适应性测试方法

7.3.2.5.1 区域与时间适应性测试

区域与时间的适应性测试方法如下：

a） 构建多区域环境模拟系统，通过虚拟化等技术，借助国家或地区、时区、语言等的切换工具，验证软件时间显示、定时任务触发逻辑、度量衡及货币单位以及翻译的正确性；

b） 使用自动化测试工具，多机并行测试多语言的文本输入、存储及检索功能，检查字符编码兼容性以及文字排版正确性等。

7.3.2.5.2 法律和政策适应性测试

法律与政策适应性测试方法如下：

a） 采用法规适配性扫描与评估工具，检测软件代码中的政策关键词库，自动标记数据存储位置、用户授权流程等潜在违规点，并对违规点的内容进行测试；

b） 结合自动化校验与人工审核，验证敏感操作是否触发合规提示或阻断机制。

7.4 软件环境适应性评估方法

根据上述软件系统环境适应性测试结果，分析影响软件系统功能和性能的环境适应性因素的类型与边界，并利用统计或深度学习等方式，构建出软件系统环境适应性评估模型，可对软件系统实现环境适应性的趋势分析、评估与预测。

7.5 软件环境适应性改进方法

根据上述软件系统环境适应性评估模型与结果，对软件系统开展环境适应性的防护改进，应对关键环境因素进行监测和预警，必要时采用功能降级、静默等方式避免、降低或缓解高风险的环境条件所可能带来的灾难性后果。

8 复杂软件系统环境适应性全生命周期过程与活动

8.1 需求分析阶段

8.1.1 自然环境适应性分析

需求分析阶段的工作内容核心为全面识别和量化软件系统在其全生命周期内可能遭遇的自然环境条件，并将其转化为具体的自然环境适应性要求。具体工作内容包括：

a） 识别软件系统预期的部署和运行环境，明确温度、湿度、振动、电磁干扰、空间辐射等关键自然环境因素，并依据相关标准或实测数据量化其正常工作范围与极限阈值；

b） 分析各类自然环境因素对系统硬件及软件功能的潜在影响，评估其风险等级，并定义软件层级应采取的容错、降级或恢复策略；

c） 针对盐雾、霉菌等特殊或极端自然环境，识别其特有的影响机制，并将其作为专项需求纳入软件适应性设计考量；

d） 建立并维护覆盖全生命周期的环境风险清单，系统性地记录各类自然环境风险及其对应的初步应对方法。

8.1.2 运行环境适应性分析

在需求分析阶段，需系统性分析为基础，清晰界定软件对运行环境的依赖与兼容边界，得到运行环境适应性要求。为达成此目标，可进行以下工作：

a） 开展硬件平台兼容性分析，界定软件支持的处理器架构、内存、存储及外设等范围，并提炼硬件差异的适配要求；

b） 制定并维护软件依赖兼容性列表，明确操作系统、数据库、中间件及各类支撑软件的类型与版本范围；

c） 推行以开放标准为优先的数据与接口策略，明确数据格式、接口协议及国际化适配要求，规避私有格式带来的移植壁垒；

d） 实施网络环境影响评估，分析不同网络条件对软件功能和性能的影响，并制定相应的容错与自适应要求。

8.1.3 其它环境适应性分析

在需求分析阶段，为识别其它环境适应性要求，可开展以下的工作：

a） 实施国际化与本地化需求分析，识别目标市场的语言、时区、度量衡、文化习俗等特征，并形成具体需求；

b） 开展目标市场的法律合规性尽职调查，特别是针对数据保护、隐私政策、内容审查等方面，形

成合规性要求清单。

8.2 设计与实现阶段

8.2.1 自然环境适应性设计与实现

8.2.1.1 温度适应性设计

在软件温度适应性方面，设计阶段需要开展如下工作：

a） 动态功耗管理设计：设计软件控制的功耗调节算法，根据温度传感器数据动态调整计算负载。例如，高温时主动降频或切换至低功耗模式，减少芯片发热；

b） 热状态监测与响应设计：软件实时解析温度传感器数据，触发温度阈值告警或执行降级策略（如关闭非核心任务）；

c） 算法鲁棒性设计：关键算法需内置传感器的温度补偿逻辑，确保在温度漂移时输出仍保持稳定。

8.2.1.2 湿度适应性设计

软件湿度适应性方面需开展的设计工作如下：

a） 腐蚀风险预测设计：软件集成湿度传感器数据，结合时间序列分析预测腐蚀风险，并主动调整工作模式；

b） 数据校验强化设计：在潮湿环境中强化通信数据的冗余校验与重传机制，防止因电路漏电导致的数据错误。

8.2.1.3 振动和冲击的适应性设计

振动适应性设计主要面向具备加速度/角速度数据输入，或操作对振动和冲击敏感的外设的软件，具体设计工作如下：

a） 数据滤波设计：对于直接测量加速度、角速度等动力学数据的软件系统，采取滤波等措施降低数据毛刺造成的影响；

b） 振动影响降低设计：对于振动和冲击间接影响数据质量的情况，如摄像头画面模糊等，采取措施降低振动影响，优化数据质量；

c） 硬件失效防范设计：对于振动和冲击可能造成硬件失效的情况（如机械硬盘磁头运动异常、接口松动等），采取措施避免软件逻辑异常，必要时主动暂停或禁用振动敏感硬件，避免硬件受损；

d） 特殊交互设计：对于使用重力感应、手势感应等方式进行用户交互输入的软件，提供其它交互方式作为备份，避免在特定振动/冲击环境下无法交互。

8.2.1.4 电磁干扰的适应性设计

面向电磁干扰的软件环境适应性设计工作如下：

a） 设计针对采集到的信号进行抗干扰滤波处理的算法，消除或削弱电磁干扰影响；

b） 设计数据传输过程中引入校验机制，检测和纠正数据错误；

c） 设计异常处理机制，确保受到干扰后自动恢复；

d） 设计根据环境干扰情况，通过软件动态调整通信参数（如传输速率、功率控制等）的算法，以适应不同干扰强度。

8.2.1.5 空间辐射的适应性设计

空间辐射适应性的设计工作如下：

a） 设计三模冗余存储等软件纠错方法；

b） 使用ECC内存管理等方式保护内存，并自动检测纠正单比特翻转；

c） 设计检查点机制，周期性保存任务状态至非易失存储，故障后从最近检查点恢复运行。

8.2.1.6 位置坐标适应性设计

位置坐标的适应性设计过程可采用如下工作：

a） 设计位置坐标数据（如经纬度、高度等）的实时监测算法，识别超出合理范围或明显偏离预期的异常数据，并进行过滤或修正，避免异常坐标影响系统正常运行；

b） 设计针对特殊位置坐标数据（如极点、赤道、本初子午线、负高度值等）的处理逻辑；

c） 设计不同地理坐标系之间的坐标转换算法，确保在多坐标系环境下仍能准确解析和使用位置信息；

d） 设计近似位置信息替代逻辑，在 GPS 或其他定位信号丢失或减弱的情况下，短暂采用惯性导航、历史轨迹推演等技术，提供近似位置信息，保证功能连续性；

e） 设计实时评估当前定位数据的可信度算法，并根据可信度动态调整位置坐标相关的处理逻辑。

8.2.1.7 组件意外断联的适应性设计

组件意外断联的适应性设计可采取如下工作：

a） 设计使用心跳、轮询等方式，检测硬件是否工作，若未正常工作则需触发降级或切换流程；

b） 设计硬件访问行为的超时机制，若连续超时未响应则判定为断连或故障，并不应影响与该硬件无关的其它执行流程；

c） 设计任务迁移与降级运行逻辑，当检测到硬件故障时，软件自动将计算任务迁移至备用硬件或降低计算精度，保障核心功能运行；

d） 设计资源隔离与进程保护逻辑，通过内存隔离和进程监控，隔离故障硬件关联的软件模块，防止崩溃扩散。例如，硬盘保护性停转时，暂停相关 I/O 进程并启用内存缓存；

e） 设计自适应负载调整逻辑：根据硬件可用性动态调整任务队列。如振动环境下，减少机械硬盘写入频率，优先使用 SSD 或内存缓存。

8.2.1.8 意外断电的适应性设计

为防止意外断电导致数据丢失，可进行以下设计：

a） 正确编程：利用写前日志、事务机制等处理方式，提升程序针对异常断电行为的防护能力；

b） 上电自检：软件重启后校验非易失性存储中的数据完整性，自动恢复至断电前状态；

c） 冗余备份：重要数据双副本存储（如本地+云端），断电时优先保存差异数据，降低备用电源负载。

8.2.2 运行环境适应性设计与实现

8.2.3 适应性设计

8.2.3.1 硬件适配性设计

硬件适配性的设计工作如下：

a） 通过适配层支持多种硬件平台，包括不同处理器架构、内存容量和外设配置等，屏蔽不同硬件和平台的差异；

b） 支持根据硬件差异灵活伸缩，如根据 CPU 核数调整并行任务数、根据内存大小调整读取数据量等；

c） 对关键硬件资源应进行运行时检测，根据资源状况动态调整运行参数；

d） 对硬件更换或扩展提供最小化适配方案，减少系统停机时间。

8.2.3.2 操作系统适应性设计

操作系统适应性的设计工作如下：

a） 采用跨平台的操作系统和编程语言接口标准；

b） 将特定操作系统功能调用封装在独立的适配模块中；

c） 开发兼容性逻辑，保证操作系统升级或补丁可在不修改核心功能模块的情况下完成适配。

8.2.3.3 数据库与支撑软件适应性设计

数据库与支撑软件适应性的设计工作如下：

a） 通过标准化数据访问接口实现数据存取，减少对特定数据库的依赖

b） 支持至少两类不同的数据库管理系统，或在同类系统的不同版本间具备迁移能力；

c） 明确对中间件、运行库等支撑软件的版本范围和兼容性策略。

8.2.3.4 组织环境适应性设计

组织环境适应性的设计工作如下：

a） 开发灵活的权限管理和角色分配功能，以适应组织内部结构变化；

b） 提供可配置的业务流程管理工具，支持动态调整和优化组织业务流程；

c） 具备外部环境变化的响应能力，支持政策、市场和技术趋势的快速适配；

d） 提供内置的沟通与协作工具，确保跨部门、跨岗位信息共享的高效性和安全性；

e） 支持模块化设计和功能扩展，满足组织持续改进和长期发展的需求。

8.2.3.5 通信与网络适应性设计

通信与网络适应性设计方面，可进行以下工作：

a） 设计通信协议适配层，支持规定的多种通信设备、协议及可配置的协议参数；

b） 设计自适应机制，对网络的带宽、延迟、丢包等条件的波动进行补偿或重试，避免软件崩溃或数据丢失。

8.2.3.6 数据与接口适应性设计

数据与接口适应性的设计工作如下：

a） 设计广泛使用且开放的数据格式和加密算法，不宜使用私有数据格式或私有加密算法；

b） 设计对输入数据的不同字符编码、时间格式、度量单位的适配和处理，避免乱码或数据错误。

8.2.3.7 并发适应性设计

对于可能被并发调用的程序或程序片段，应开展基于信号量、锁、原子变量等形式防护共享资源的设计工作。

8.2.4 易替换性设计工作

8.2.4.1 数据的连续使用性设计

数据的连续使用性设计工作如下：

a） 设计兼容性逻辑，新版本软件应完全兼容旧版本软件生成的数据库记录、配置文件、用户工程文件等数据；此外，新版本软件应支持旧版本软件数据的所有功能特性；

b） 若新旧版本的数据格式差异较大、兼容困难时，应设计自动化数据转换工具。

8.2.4.2 功能的内含性设计

对于新版本软件替换旧版本软件的情形，新版本软件应保留或增强旧版本的核心功能，且操作逻辑保持一致，不应导致旧版本核心功能出现异常。具体设计工作包括：

a） 保证用户可通过相同或更简化的操作路径完成原有任务；

b） 提供有关功能接口变更的明确兼容性说明及适配工具；

c） 通过接口封装或模拟层，确保兼容依赖旧版本功能的第三方集成（如插件、服务调用等）。

8.2.5 共存性设计工作

8.2.5.1 资源共享性设计

资源共享性可通过以下设计工作进行保证：

a） 资源隔离设计：软件在共享硬件资源（CPU、内存、存储等）时，设计资源分配隔离逻辑，避免因资源争抢导致性能降级（如内存泄漏或CPU占用率超过阈值）；

b） 动态调整设计：设计支持资源占用的动态调节，例如在检测到其它软件高负载时自动释放非关键资源；

c） 冲突处理设计：设计冲突处理逻辑，当资源不足时，应通过日志记录或告警通知用户，而非直接崩溃。

8.2.5.2 功能兼容性设计

功能兼容性可通过以下设计工作进行保证：

a） 设计检测API或服务接口被其它软件错误调用或占用的逻辑，并支持自动处理或提示用户处理；

b） 进行读写权限隔离，避免共享数据库或文件时的数据污染；

c） 对多版本并行加载或静态链接进行设计，以避免版本冲突。

8.2.5.3 环境兼容性设计

环境兼容性可通过以下设计工作进行保证：

a） 最小化系统级配置修改，充分检查配置项读写，避免覆盖系统和其它软件的配置；

b） 声明依赖的系统服务或第三方服务的版本要求，并支持服务不可用时的降级处理；

c） 对于虚拟机或云环境中的情况，还需设计虚拟机或云环境中的资源调度策略。

8.2.5.4 用户界面兼容性设计

用户界面兼容性可通过以下设计工作进行保证：

a） 根据目标平台的窗口及通知管理规范设计界面，保证不无故抢占系统焦点、不无故干扰用户当前正在其它应用中的操作、不滥用通知和弹窗等；

b） 设计过程中不断评估视觉和声音的干扰，检查图形界面是否会对用户或其它应用的视觉呈现造成干扰，声音应提供便捷的音量调节选项，不应覆盖系统或其它应用正在播放的音频；

c） 设计过程中保证优先使用操作系统提供的标准UI控件和主题，以确保外观和行为与系统及其它应用保持一致，降低用户学习成本；

d） 正确设计响应输入事件的处理逻辑，尽量降低全局输入信号的依赖，且不应独占或拦截可能发往其它应用的输入信号。

8.2.6 易安装性设计

8.2.6.1 安装正确性设计

安装正确性可通过以下设计工作进行保证：

a） 安装程序中集成自动检测系统环境的能力，确保安装路径、依赖库等配置正确无误；

b） 设计安装完成后的验证机制（如校验和、功能测试）以确保软件的完整性和可用性；

c） 对安装过程中出现的错误提供明确的错误提示和解决建议，避免因错误导致安装失败或系统异常。

8.2.6.2 安装影响性设计

安装影响性可通过以下设计工作进行保证，避免安装影响其它软件或被其它软件影响：

a） 保证安装过程不对系统核心配置（如注册表、环境变量）覆盖或修改，若需修改，应提前告知用户并提供备份机制；

b） 保证软件安装过程较少依赖或影响其它软件，不影响其它已安装软件的正常运行，避免因文件冲突或资源占用导致系统性能下降；

c） 提供卸载功能，确保卸载后能够彻底清除所有相关文件和配置，不残留无用数据。

8.2.6.3 安装难易性设计

安装难易性可通过以下设计进行降低：

a） 尽量简化安装过程，减少用户输入步骤，支持一键安装或默认配置安装；

b） 开发集成在软件内部的安装向导和操作说明，确保普通用户无需专业培训即可完成安装；

c） 开发多语言安装管理界面，满足不同语言用户的需求，提升用户体验。

8.2.6.4 安装灵活性设计

安装灵活性可通过以下设计和实现方法进行保证：

a） 设计和实现自定义安装选项支持，如选择安装路径、组件或功能模块，满足用户的个性化需求；

b） 设计和实现多种安装模式（如完整安装、最小安装、自定义安装）支持，以适应不同的硬件条件和用户需求；

c） 设计和实现静默安装和脚本化安装支持，便于批量部署和自动化管理。

8.2.6.5 安装效率设计

安装效率可通过以下设计和实现方法进行保证：

a） 优化安装步骤，减少非必需的文件拷贝和配置操作，提升安装速度；

b） 优化安装过程对系统资源的占用，避免因高资源消耗导致系统卡顿或崩溃；

c） 提供安装进度提示和预计完成时间，帮助用户合理安排安装过程。

8.2.7 移植完整性设计

8.2.7.1 移植正确性设计

移植正确性可通过以下设计方法进行保证：

a） 设计和实现移植后的验证机制，如功能测试、性能测试等，以确保软件的完整性和可用性；

b） 设计和实现移植过程的错误提示和解决建议。

8.2.7.2 移植一致性设计

移植一致性可通过以下设计工作进行保证：

a） 移植后的软件设计为与原环境中的软件相同的功能逻辑、用户界面和操作流程；

b） 设计数据校验逻辑，校验移植后数据的完整性和一致性，避免因环境差异导致数据丢失或逻辑错误；

c） 设计配置检测逻辑，检测和主动适配移植后新环境的硬件和软件配置，确保性能表现符合要求。

8.2.8 其它环境适应性设计与实现

在设计与实现阶段，为保证其它环境适应性，可进行以下设计：

a） 采用代码与资源分离的国际化框架，以支持多语言、多时区和多度量衡的动态配置与切换；

b） 构建合规驱动的数据处理与隐私保护架构，通过技术手段实现用户授权、数据加密、内容审查等功能，以满足不同市场的法规要求。

8.2.9 综合环境适应性设计

在复合环境条件下开展的综合环境适应性设计工作主要如下：

a） 设计复合条件触发的策略（如湿度阈值触发高温防护升级）；

b） 设计不同环境因素冲突时的解决策略（如降频散热与增强错误校验算力冲突），按实时风险动态切换主策略；

c） 进行环境耦合建模与预测算法设计：建立环境交互模型，通过多传感器数据融合，预测多环境因素作用下的连锁失效；

8.3 测试阶段

8.3.1 自然环境适应性测试工作

自然环境适应性的测试工作主要分为模拟测试、试验测试和现场测试三类，可按需选用以下的工作：

a） 模拟测试：使用环境模拟设备或仿真软件模拟目标环境条件，测试软件在特定环境下的功能及性能。可根据实际应用场景设计不同的测试用例和组合测试用例。

b） 试验测试：实验室环境中，使用专用试验设备（如环境试验箱、振动台、电磁干扰发生器等）对软件或其运行载体施加可量化的环境应力，验证软件在特定物理条件下的功能稳定性与性能边界；

c） 现场测试：在目标环境中进行现场测试，验证软件在实际环境下的适应性。应记录测试过程中的环境参数、软件运行状态和测试结果，并进行分析和评估。

8.3.2 运行环境适应性测试工作

在测试阶段，可通过以下工作保证软件的运行环境适应性：

a） 在测试阶段进行运行环境适应性专项测试，覆盖硬件、操作系统、数据库、支撑软件、网络等；

b） 在确认测试阶段完成全部运行环境适应性指标的验证；

c） 建立运行环境适应性回归测试用例库，在版本升级时重复执行；

d） 测试结果应形成可追溯的验证记录和分析报告。

8.3.3 其它环境适应性测试工作

在测试阶段，为保证其它环境适应性，可进行以下工作：

a） 对软件在所有语言环境下的功能、性能和界面展示进行测试并评估，重点关注不同语言环境下的字符处理、日期格式、货币格式等问题；

b） 测试和评审软件是否符合目标市场的法律法规要求，例如数据保护、隐私政策、内容审查等。

8.3.4 综合环境适应性测试工作

在测试阶段，为测试多种环境因素共同作用的综合环境适应性，应将多种环境因素组合作用于软件，并评估软件是否可完成规定的功能。

8.4 使用阶段

8.4.1 自然环境适应性保证工作

使用与维护阶段的保障工作核心为建立持续监控、预警并迭代优化的闭环管理体系。为达成此目标，可进行以下工作：

a） 部署覆盖环境参数与软件运行状态的实时监控体系，实现对极端自然环境条件的有效预警；

b） 建立持续性的数据分析机制，研判环境风险的变化趋势，并结合用户反馈，形成驱动适应性优化的数据支撑；

c） 明确并执行标准化的维护与更新规程，确保更新活动在适宜的环境下进行，并具备应对环境干扰与更新失败的安全回退能力。

8.4.2 计算环境适应性保证工作

在使用与维护阶段，其核心策略是确保软件在运行环境不断演进的过程中，能够保持业务的连续性和服务的稳定性。为达成此目标，可进行以下工作：

a） 建立常态化的资源消耗监控与用户反馈收集渠道，为运行环境的适应性优化提供决策依据；

b） 实行透明化的版本变更管理，在软件更新时，明确声明新版本对运行环境的依赖变更；

c） 提供平滑的升级路径与方案，支持新旧版本短时共存或并行运行，并为高可用系统规划不停机更新路径；

d） 将版本回退作为标准应急预案，确保在更新引发严重兼容性问题时，能够快速恢复服务。

8.4.3 其它环境适应性保证工作

在使用与维护阶段，为保证其它环境适应性，可进行以下工作：

a） 建立目标市场法律与政策的持续监控机制，快速响应新规定与政策，并规划软件的适配更新；

b） 实施本地化资源的持续迭代，根据用户反馈和市场拓展，不断完善语言包和区域配置等。

8.5 维护与更新阶段

软件维护与更新时需进行的环境适应性保证工作如下：

a） 环境保证：应明确提示维护与更新所必需的自然环境条件；

b） 过程保护：需防护更新过程中可能遇到的环境干扰，若更新失败后仍可恢复到旧版本运行；

c） 多版本共存：应支持新旧版本在同一运行环境中共存，确保业务连续性以及依赖本软件的其它软件的完备性；

d） 无缝升级：对于高可用性要求的服务，需采用不停机更新方法，保证更新中服务的可用性。

e） 法律动态适配：更新内容后需经过审查，保证合规性；

f） 回退机制：更新失败时自动触发版本回滚；

参 考 文 献

［1］ GB/T 11457-2006 信息技术 软件工程术语

［2］ GJB/Z 141-2004 军用软件测试指南

［3］ GJB 439A-2013 军用软件质量保证通用要求

［4］ GJB 841A-2024 通用质量特性问题报告、分析和纠正措施系统

［5］ GJB 2725B-2024 军用校准和测试实验室能力通用要求

［6］ GJB 2786A-2009 军用软件开发通用要求

［7］ GJB 5716-2006 军用软件开发库、受控库和产品库通用要求

［8］ GJB 5880-2006 软件配置管理

［9］ ISO/IEC 25010-2023 Systems and software engineering — Systems and software Quality Requ- irements and Evaluation (SQuaRE) — Product Quality Model