Computer Science and Application 计算机科学与应用, 2026, 16(2), 505-513
Published Online February 2026 in Hans. https://www.hanspub.org/journal/csa
https://doi.org/10.12677/csa.2026.162079

Hans汉斯

系统失效智能检测技术研究

刘宏巍，任东伟

中国信息安全研究院有限公司，北京

收稿日期：2026年1月21日；录用日期：2026年2月19日；发布日期：2026年2月26日

摘　要

当前，高度复杂和紧密耦合的信息系统已在互联网、零售、金融、医疗等多个关键领域部署应用，系统一旦失效，不仅影响业务连续性，还可能造成人民生命财产安全重大损失。因此，关注系统失效问题，对有效提升系统可靠性与安全性，对加速社会数字化转型具有深远意义。本文聚焦系统失效原因、传播机理、引致结果等问题，总结相关理论研究，分析系统失效案例特征，简述多种系统失效检测技术代替传统失效检测方法，快速溯源、实时监测预警，最终基于以往项目经验设计一套应用系统实施方案框架，形成“理论引领－技术分析－应用落地”研究逻辑，为如何为企业用户提供安全稳定的软件应用环境提供参考。

关键词

系统失效，失效原因，智能检测，原型系统，解决方案

Research on Intelligent Detection Technology for System Failures

Hongwei Liu, Dongwei Ren

China Information Security Research Institute Co., Ltd., Beijing

Received: January 21, 2026; accepted: February 19, 2026; published: February 26, 2026

Abstract

Currently, highly complex and tightly coupled information systems have been deployed and applied in multiple key areas such as the Internet, retail, finance, and healthcare. Once these systems fail, it not only affects business continuity but may also cause significant losses to people's lives and property. Therefore, paying attention to system failure issues is of profound significance for effectively enhancing system reliability and safety, as well as accelerating the digital transformation of society. This paper focuses on issues such as the causes of system failures, propagation mechanisms, and resulting outcomes, summarizes relevant theoretical research, analyzes the characteristics of system failure cases, briefly describes various system failure detection techniques that replace

文章引用: 刘宏巍, 任东伟. 系统失效智能检测技术研究[J]. 计算机科学与应用, 2026, 16(2): 505-513.
DOI: 10.12677/csa.2026.162079

traditional failure detection methods, enable rapid traceability, real-time monitoring, and early warning, and ultimately designs an implementation framework for an application system based on previous project experience, forming a research logic of "theory guidance, technical analysis, application implementation". This provides a reference for how to provide a secure and stable software application environment for enterprises and users.

Keywords

System Failure, Cause of Failure, Intelligent Detection, Prototype System, Solution

1. 系统失效研究意义

一是发展趋势，软件失效零容忍。在软件定义时代下，软件已深度嵌入工业制造、金融医疗、航空航天等关键领域，从智能设备精准控制到金融网络稳定运行，从自动驾驶安全保障到能源系统智能调度，软件鲁棒性直接决定系统的安全性与可靠性。因此，软件失效必须零容忍，这不仅要求在代码层面消除设计缺陷，更需构建覆盖全生命周期防控体系，即：从需求理解开始杜绝设计漏洞，利用动态故障注入测试强化系统鲁棒性，借助大数据挖掘与机器学习实时监测运行状态，并依托微服务架构实现故障隔离与自动化修复，建立“预测–检测–响应–优化”闭环体系，确保软件成为可信赖的安全底座。

二是问题频出，复杂系统失效成为现实压力。数智化深度融合下，复杂系统规模与复杂度呈现指数级增长，其失效引发经济、安全乃至社会风险远超传统容错机制的承受范围。系统架构已从单一集中式走向分布式、微服务化，集成平台可能包含数十万计微服务组件、百万级代码行，各组件间又通过动态网络拓扑形成复杂依赖关系，此特性使单一模块的隐性缺陷可能触发级联反应。第二，失效后果从软件单个功能异常升级为多重因素叠加作用。第三，软件漏洞可能成为网络攻击入口，引发系统性安全危机。传统基于规则匹配与人工排查的软件失效分析模式在面对新挑战下已显现响应滞后。2023 年 7 月，美国发射空间飞行探测器“旅行者 2 号”被输入一系列例行命令后，触发“旅行者 2 号”天线方向 2 度变化，导致探测器系统与地球通信中断。2024 年 7 月，CrowdStrike 例行的 Falcon 传感器更新导致 850 万台 Windows 设备出现故障，错误配置更新与现有 Windows 配置冲突，导致全球 850 万台设备宕机。

三是范式创新，技术发展突破传统检测方法。当前应对软件失效的多数标准。仍以传统测试方法为主，对实时动态监测的技术路径缺乏明确指引。人工智能、边缘计算等新兴技术融入到规范标准，能够直面复杂系统失效分析核心痛点，既填补传统方法在动态传播建模和实时检测方面不足，也为智能运维提供理论和技术储备。自然语言处理(NLP)技术、知识图谱建模技术与流数据处理框架作为新一代人工智能与大数据技术的核心代表，其技术成熟度可支撑复杂系统实现实时智能监控与检测。三者通过协同作用，形成“数据感知–语义理解–关联推理–动态响应”闭环能力，提升系统实时性、准确性与智能化水平。

2. 近年系统失效现状及问题分析

近年系统失效事件呈现高发化、广域化、重损化态势，覆盖互联网、金融、能源、医疗、交通、航天、装备等多关键领域，影响范围从单一企业扩散至跨行业生态，损失程度已从经济损失上升至生命安全威胁。如表 1，本报告搜集了 110 多例自全球多行业典型系统失效案例，最大程度梳理系统失效特征及失效原因。如图 1，通过梳理样本案例发现，系统失效多发系统运行稳定性失控，其失效根本原因多为软

件设计先天缺陷、运行环境的动态扰动或运维管理的漏洞缺失等。此外，人工智能驱动的自动化攻击、供应链漏洞传导让系统面临“内忧外患”的双重压力。以上种种都在表明研究系统失效检测分析技术的重要性，以及如何运用人工智能手段能够快速识别预警潜在的系统失效风险。

Table 1. Showing a portion of the typical system failure sample information selected in this paper
表 1. 本文选取典型系统失效样本信息部分展示

序号	失效事件	行业	时间	失效过程	失效原因
1	雅虎报告违规	互联网	2016 年	大约 5 亿份可追溯到四年前的凭证数据泄露。	网络攻击 软件漏洞
2	服刑人员越狱	公检法	2015 年	系统运行故障导致超 3200 名美国囚犯在宣布的日期前获释。服刑人员平均提前 49 天释放。	软件突发失灵
3	彭博取消债务发行	金融	2016 年	英国债务管理办公室(DMO)因网络中的硬件和软件故障，导致网络流量过大，交易终端瘫痪了两个小时。	软硬件不兼容
4	空中客车软件错误警报	航空	2015 年	西班牙 A400M 飞机发现一个软件缺陷，导致空客发出警报。	软件设计漏洞
5	合作社食品双重收费	零售	2015 年	应用软件突发一次性技术故障导致客户被收取两次费用。	软件突发失灵
6	美“爱国者”防空导弹向己方或友军战机开火	装备	2003 年	软件系统中敌我识别系统模块存在漏洞，酿成多起“自相残杀”事件。	软件设计漏洞
7	英国政府新的在线农业支付系统被推迟	金融	2015 年	门户和规则引擎软件之间的集成问题，导致政府推迟 1.54 亿英镑的农村支付系统启动。	软件集成故障
8	对核电站的网络攻击	能源	2016 年	“网络攻击”造成核电站中断。	网络攻击软件漏洞
9	Interlogix 召回个人应急设备	网络安全	2016 年	因操作故障该设备在紧急情况下无法与安全系统通信，Interlogix 无线个人应急设备制造商召回了大约 67000 台设备	软硬件不兼容
10	癌症治疗与致死性放射治疗	医疗	2000 年	巴拿马城基于输入数据的顺序，治疗计划软件计算出并提供双倍剂量的辐射。	输入错误数据
11	软件失败，导致系统崩溃	金融	2012 年	RBS IT 人员升级 CA7 批处理软件失败，导致系统崩溃，影响数百万客户无法访问他们的账户，或者执行任何付款。	软件更新出错引致系统失灵
12	丰田普锐斯因软件故障召回	汽车	2014 年	编程错误导致汽车的油电混合动力系统关闭，丰田汽车在全球召回 190 万辆最新一代普锐斯汽车。	代码编程漏洞
	……				

备注：样本案例数据来源一部分来源于《航空航天中软件错误事件的分类》，另一部分来自于各大互联网官方网站。

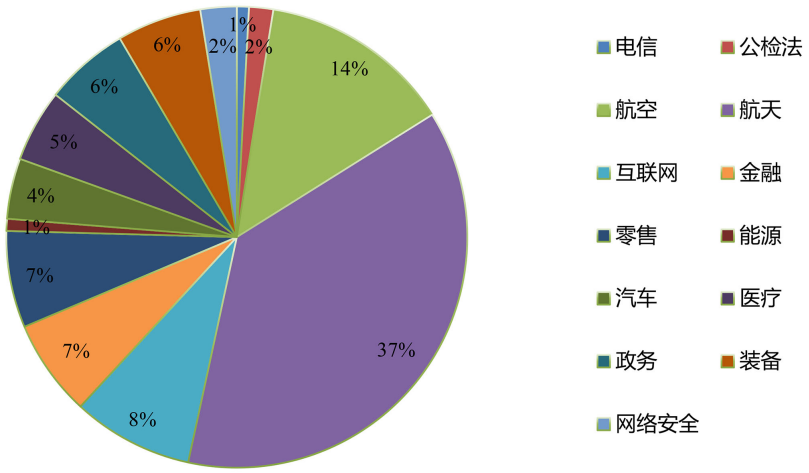

Figure 1. Shows the percentage distribution of typical system failure cases by industry sector as selected in this paper
图 1. 本文选择典型系统失效案例行业分类占比

如图 2，从系统故障可看出，系统失效诱因复杂多变，其中系统失灵、软件设计漏洞人为操作失误等在系统失效直接原因中占比较高。

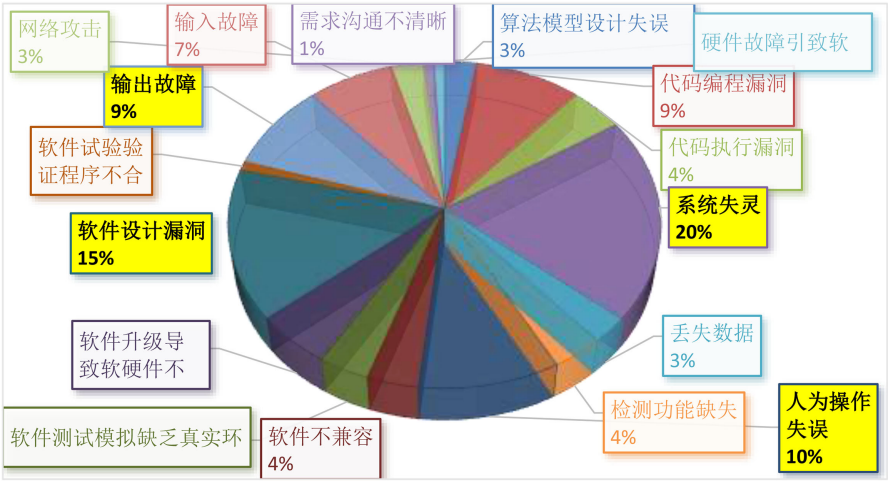

Figure 2. Distribution of specific causes of system failures in the sample data
图 2. 样本数据系统故障具体原因分布情况

综合上述系统故障具体原因分析，可大概聚拢分为三类重点(表 2)，即：任务规范与系统设计失败、系统之间未对齐、任务验证与终止。

Table 2. System failure classification definitions
表 2. 系统失效分类定义

系统失效原因	定义范围	软件故障	故障失效原因
任务规范与系统设计失败	这类失效源自系统设计层面的缺陷，包括对任务要求理解不当、会话管理不良，或者系统的角色与职责定义不清/不被遵守等。系统没有正确执行应做的事情，失效原因多为任务本身描述或约束有纰漏，或系统没有按用户要求来做。	违背任务规范	系统未能遵循给定任务指定的约束或要求，从而导致次优或错误的结果。
		违背角色规范	系统未能遵循为其分配的角色所定义的职责和约束，导致该系统行为表现像不同角色的系统。
		步骤重复	在执行流程中不必要地重复之前已经完成步骤，导致任务完成延迟或引入错误。
		丢失对话历史	系统发生意外的上下文截断，忽略最近交互历史记录，并回退到之前的某个对话状态。
		未意识到终止条件	系统缺乏对触发交互终止所需条件的识别或理解，导致交互在不必要情况下继续进行。
系统之间未对齐	该类别涵盖由于沟通效率低下、协作不力、系统之间行为发生冲突，以及系统行为逐渐偏离最初设定的任务目标而引发的各类失效。	对话重置	对话被意外或无理由地重新启动，可能导致丢失之前上下文信息以及在交互中已经取得进展。
		未能请求澄清	系统在面对不清晰或不完整数据时，没能主动请求额外信息以明确指令，可能导致后续采取错误行动。
		任务偏离	系统的行为或系统的焦点偏离给定任务的既定目标或核心关注点，导致执行不相关或无效率的动作。
		信息隐瞒	系统未能分享或传达其所掌握的、并且如果分享就可能影响其他系统决策的重要数据或见解。
		忽略其他系统输入	系统无视或未能充分考虑系统中其他系统提供的输入或建议，导致做出次优决策或错失协作良机。
		推理行为不匹配	系统逻辑推理过程与其最终实际采取的行动之间存在偏差，导致出现意料之外或不希望发生的行为。

续表

任务验证与终止	该类别涵盖由于执行过早终止，以及缺乏足够的机制来保障交互、决策和最终结果的准确性、完整性和可靠性而导致失效。	过早终止	在所有必要的信息交换完成或所有目标都已达成之前就结束了对话、交互或任务，这可能导致最终结果不完整或不正确。
		无验证或验证不完整	系统完全或部分地省略了对任务结果或系统输出进行适当检查或确认的环节，这可能使得错误或不一致性在未被察觉的情况下扩散开来。
		错误验证	在系统迭代过程中未能充分地验证或交叉核对关键信息或决策，可能导致系统中引入错误或潜在的漏洞。

3. 理论研究概述

研究国内外技术文献，对于理解和实现系统失效的智能检测具有不可替代的重要性。从上述分析的系统失效原因可见，系统失效故障类型多样且相互关联，单纯依赖统计数字无法揭示深层机理与应对路径。而技术文献正是将这类图表“激活”为知识体系的关键桥梁。国内研究从追踪模仿走向自主创新，形成了贴合我国工程实际的分析框架，为智能检测系统提供本土化的模型基础与场景理解。国外研究则以其深厚理论积累与前沿技术迭代见长，推动了检测技术从被动响应向主动预测的演进。国外对系统失效理论研究起步早，在失效分析机构建设、多领域研究以及复杂系统失效研究等方面较深入，注重基础理论创新突破，技术方法迭代迅速。Google 的 OSS-Fuzz 项目通过 AI 驱动的模糊测试，在 OpenSSL 中发现 26 个漏洞，包括存在二十年的缺陷。Preeti Baderiya 等[1]介绍自动静态分析、图挖掘、分类器等检测方法，指出在数据不平衡等方面问题，强调提升软件可靠性的研究目标与方向。总而言之，国内外文献共同构建了系统失效智能检测的知识底座，通过系统梳理文献，将图中静态的“故障占比”转化为动态的“检测逻辑”，最终构建起融合因果分析、模式识别、实时预警与自主决策的智能检测系统，真正实现从“知其然”到“知其所以然”再到“防其未然”的跨越。

谢瑞生[2]提出从软硬件比较入手，分析软件失效内外部、主客观原因，涵盖需求、设计、编码等阶段差错，以及安装、培训、运行环境等外部因素，综合进行因果分析。赵昶宇等[3]介绍软件失效模式与影响分析(SFMEA)，软件按生命周期分为系统、功能、接口、详细四级，阐述各层级分析方法，关注软件失效关键变量。孟令中等[4]提出一种软件失效模式自动生成方法，利用文本挖掘技术，经分类、聚类、标签提取和文本匹配实现自动生成，并通过实验验证该方法的有效性，将其表示为失效原因与影响的组合。钱丽等[5]聚焦基于大数据挖掘的软件失效安全风险自动评估方法，通过改进聚类算法采集风险信息、计算失效概率并构建评估矩阵，实验显示其准确性优于传统方法，可有效评估软件失效风险。郭世杰等[6]提出基于知识图谱的软件测试知识检索方法，构建包含测试项、用例、缺陷的知识图谱，结合多级词库分词和图计算检索，实验显示该方法准确率、召回率优于传统方法，提升测试知识复用效率。复旦大学杨振国团队提出“材料失效分析鱼骨图综合分析法”，覆盖设计、制造、运行等 11 个维度，并建立失效案例数据库，推动失效分析从经验驱动向数据驱动转型。李晓雪等[7]利用 Daikon 提取程序不变量，通过实验分析“随机、分支覆盖、分块覆盖”三种测试方法对软件失效数据的影响。

4. 智能检测技术研究

针对上述对系统失效原因梳理以及对国内外技术文献的理解，系统失效智能检测技术要借助多源异构数据融合处理、自然语言处理(NLP)等技术群，剖析处理文本信息，实现系统失效数据融合处理与特征提取，整合各类复杂数据，构建系统失效模式知识图谱，开展快速溯源与根因分析。依托流式数据处理等技术，实时监测预警，同时进行漏洞检测与修复，全方位提升检测分析能力。

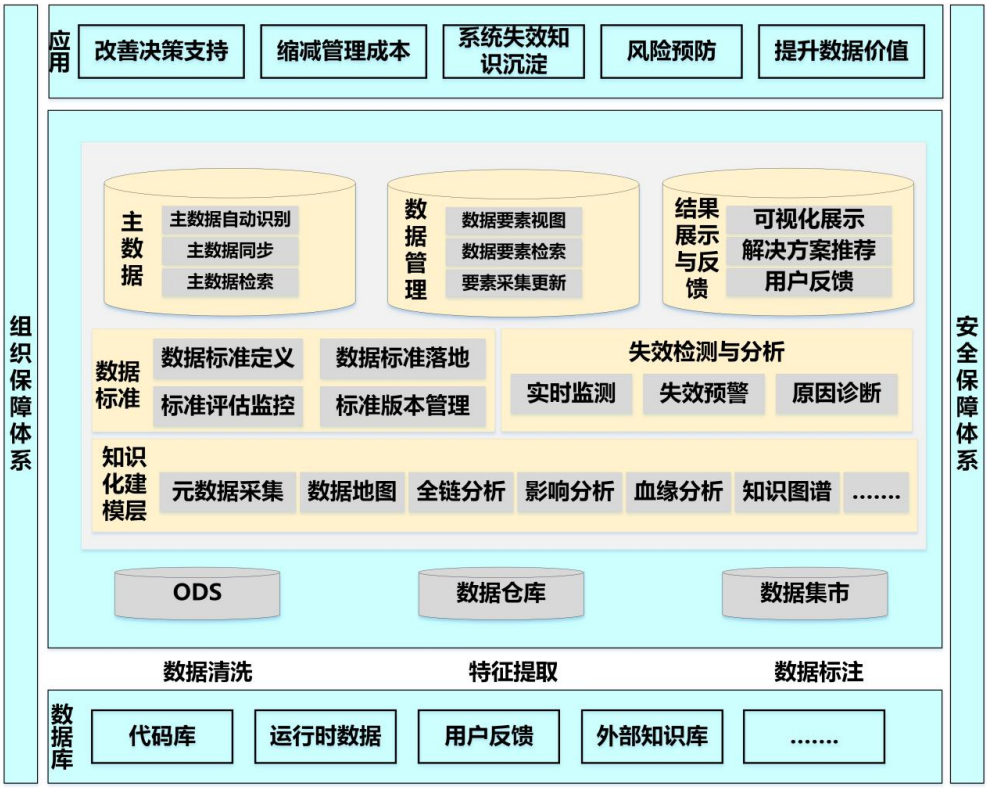

Figure 3. Architecture diagram of the intelligent failure detection prototype system
图 3. 系统失效智能检测原型系统架构图

基于上述技术群的系统失效智能检测系统应用而生：

系统失效智能检测系统旨在通过智能化技术实现设备故障的实时监测、精准诊断与快速响应，构建一套具备自愈能力的软件系统安全防护体系。如图 3，在技术架构方面，系统失效智能检测基于大语言模型，借助大语言模型强大的语义理解和知识融合能力，结合知识化建模技术，构建全面的软件失效知识体系，对软件失效进行精准检测与深入分析，实现对软件失效的早期预警、原因诊断和解决方案推荐。在系统架构方面，系统失效智能检测支持用户自定义领域本体并动态扩展，以此为基础支持对用户自有结构化数据或第三方知识库数据进行知识导入和知识合并，支持从专项数据库、典型方法库等数据库中抽取软件失效术语、软件失效相关实体、活动事件等知识节点，以及知识节点之间的关联关系，支持针对用户在业务开展过程中会不断更新软件失效领域数据和增加的新类型业务数据，实现领域对象数据的动态知识抽取与知识融合，并能以知识图谱的形式进行存储、展示和分析。

如表 3，系统失效智能检测系统是通过多源数据采集与智能预处理功能，全面整合软件全链路数据，包括代码仓库、运行日志、监控指标及用户反馈等结构化与非结构化信息，并经过数据清洗、标准化标注及敏感信息脱敏处理，构建高质量分析基础。基于知识图谱动态建模技术，系统定义失效模式实体与语义关系，结合大语言模型与规则引擎实现自动化知识抽取，通过流处理框架支持实时更新与冲突检测，形成完整的失效知识体系。在核心检测环节，系统运用大语言模型驱动代码静态分析，精准识别空指针引用、内存泄漏等典型缺陷，并通过日志语义理解实现错误聚类与根因推测。针对用户反馈中的模糊描述，系统能自动转化为精准技术术语，显著提升问题定位效率。通过因果关系推理与影响范围分析，系统不仅实现失效根因的精准定位至代码模块级别，还能评估连锁影响并预测未来失效趋势，为预防性维护提供决策依据。系统还提供知识驱动的个性化解决方案推荐，结合可视化界面直观展示失效分布与影

响关系，支持交互式深度查询。通过实施冗余备份机制与全面培训体系，确保系统在关键组件故障时保持运行连续性，同时提升运维人员的故障处置能力，最终形成集检测、分析、预测、决策支持于一体的智能防护体系。

Table 3. Functional design of the intelligent detection and analysis solution for system failures
表 3. 系统失效智能检测分析解决方案功能设想

系统功能	具体功能	功能说明
多源数据采集与智能预处理	全链路数据整合	1. 支持从代码仓库(Git)、日志系统(如 ELK)、监控工具(Prometheus)、用户反馈(JIRA 工单)及第三方漏洞库(CVE)等来源自动采集数据。 2. 通过 API 接口或文件上传方式兼容结构化与非结构化数据(如代码、文本日志、时序指标)。
	数据清洗与标准化	对采集到的原始数据进行清洗，去除噪声、重复数据和不完整记录，同时按照特定的规则和标准对数据进行标注，如将数据标记为正常或异常、不同类型的失效模式等，以便大语言模型进行学习和分析。
	敏感数据脱敏	自动识别并加密隐私字段，确保合规性。
知识图谱动态建模与更新	本体定义与关系建模	基于行业标准定义失效模式、组件、原因、影响等实体，构建“导致”“修复”“属于”等语义关系。
	自动化知识抽取	1. 使用 LLM 解析代码注释、日志描述，抽取实体与关系(如从日志“ERROR: NullPointerException”提取“空指针异常”实体)。 2. 结合规则引擎(如正则表达式)提高关键模式(如 OutOfMemoryError)的抽取效率。
	实时动态更新	通过流处理框架(如 Kafka)实时注入新数据，支持增量学习与冲突检测(如避免重复实体)。
大语言模型(LLM)驱动的失效检测	代码静态分析	1. 缺陷识别：LLM 解析代码，检测潜在问题。 2. 模式匹配：利用大语言模型对软件数据中的文本信息进行理解和分析，通过与预定义的失效模式知识库进行匹配，识别出软件中可能存在的各种失效模式，如空指针引用、数组越界、内存泄漏等。 3. 定期维护和检查：建立主动维护协议和定期检查，实时监控硬件和软件组件的健康状和性能。常规诊断、软件更新和预防性维护有助于在新出现的问题危及系统功能或任务目标之前发现和解决这些问题。
	日志语义理解	1. 错误聚类：LLM 将非结构化日志映射到标准失效模式。 2. 根因推测：生成多假设解释，按置信度排序。
	用户反馈解析	将模糊描述(如“页面卡顿”)转化为技术术语(如“前端渲染阻塞”或“API 响应延迟”)。
根因分析与定位	因果关系推理	基于大语言模型对软件系统的整体理解以及对相关数据的关联分析，系统能够推断出软件失效的可能原因，分析不同因素之间的因果关系，确定导致失效的关键因素和事件链。
	精准定位	将失效原因精准定位到具体的代码模块、函数或操作步骤，帮助开发人员快速找到问题所在，提高故障排查和修复的效率。
影响评估与预测	影响范围分析	评估软件失效对整个系统及其相关组件、功能的影响范围，预测可能引发的连锁反应和次生问题，为采取相应的应对措施提供依据。
	未来趋势预测	根据历史数据和当前软件状态，利用大语言模型预测软件未来可能出现失效的概率、时间和类型，帮助团队提前做好预防和准备工作，优化软件维护计划和资源分配。
解决方案推荐	知识驱动建议	结合软件失效知识图谱和大语言模型的知识储备，系统能够为每种识别出的失效模式推荐相应的解决方案和修复策略。
	个性化推荐	考虑到不同软件系统的特点、开发环境和业务需求，系统会根据具体情况对推荐的解决方案进行个性化调整。

续表

可视化交互与决策支持	直观展示	以直观易懂的可视化界面展示软件失效检测分析的结果，包括失效模式的分布、原因分析的关系图、影响范围的可视化呈现等，使开发人员和相关人员能够快速理解复杂的分析结果。
	交互查询	支持用户通过交互方式查询详细的分析报告、历史数据对比、特定失效模式的详细信息等，方便用户深入了解软件失效情况，同时提供用户反馈接口，以便根据实际情况对系统进行优化和改进。
冗余与备份		实施冗余硬件组件和备份系统，以减轻故障对任务执行的影响。冗余可以包括关键传感器、执行器和通信接口的重复，以及备用电源，以确保在发生主要系统故障时操作的连续性。
培训和技能示教		为负责系统软件操作、维护和故障排除的人员提供全面的培训计划和技能发展机会。强调熟练掌握硬件和软件组件的处理和故障排除能力，以及在压力下的态势感知和决策能力，降低行业场景中系统故障概率。

以输入故障导致系统失效为例。

输入故障作为系统失效的诱因之一，其引发系统失效的过程并非简单的线性因果，而是一个遵循“感染－传播－执行”模型的动态毒化链条。它始于一个看似微小、异常的输入数据，最终却能瓦解整个系统稳定态，其核心机制在于：输入故障充当扰动复杂系统内部平衡的初始能量，通过软件既定的逻辑与数据通路进行传播与放大，最终因系统的容错边界被击穿而导致功能丧失。典型的失效过程始于感染阶段：当异常输入(如一个超出范围的数值、一个过早到达的信号或一个格式错误的字符串)被系统接收时，它并未被有效的输入验证或过滤机制所拦截，从而成功“侵入”系统内部，形成一个局部的错误状态。例如，一个类型不匹配的输入可能在动态语言中产生一个非预期的对象，或一个速率过快的网络数据流可能悄然填满并溢出有限的缓冲区。随后在传播阶段，局部的错误状态并非静止，而是通过程序既定的依赖关系网络进行扩散。这种传播主要有两种路径：数据依赖传播和控制依赖传播。在这个阶段，系统的某些防御机制(如异常处理、输入重试)可能被触发。当故障效应进入执行阶段，表现为用户或环境可感知的系统级失效。此时，被毒化的内部状态导致直接的功能性失效，如计算得出完全错误的结果(错误值导致)、服务崩溃(内存访问违规导致)、或产生非法的输出指令。也可能是更隐蔽的失效，如系统进入死循环或活锁(时间错误导致)、关键数据被静默破坏(范围错误导致)、或安全防线被突破(数量错误引发的缓冲区溢出被利用执行任意代码)。

利用智能检测技术构建针对输入故障的“监测－预警－阻断”三位一体防御体系，是避免系统失效的关键路径。智能规则引擎与机器学习模型相结合的前置验证层，对输入数据进行实时深度扫描。引入动态程序分析与传播路径追踪技术，在故障感染初期即实施干预。通过轻量级插桩实时监控关键数据流与控制流，一旦检测到输入错误引发状态异常，系统可立即依据预设的传播模型评估影响范围，并触发告警或启动隔离机制。利用模型学习系统在历史输入负载下的行为时序模式，可预测缓冲区溢出、轮询超时等速率与定时问题，并动态调整资源分配或插入延迟，从源头规避传播链的形成。这套智能检测融合了规则推理、异常感知与预测性决策的智能检测框架，不仅能够实现输入故障的精准识别与实时预警，更能通过主动的适应性调控，在故障尚未扩散成系统级失效前便将其遏制。

5. 结语

系统失效智能检测技术突破传统检测方法的局限，为不同失效场景提供个性化修复策略，为复杂软件系统的可靠性保障提供创新性解决方案思路。该技术研究可广泛应用于智能制造、自动驾驶等关键领域，对提升系统韧性、降低运维成本具有重要实践价值。

参考文献

[1] Baderiya, P., Gupta, C. and Dubey, S. (2023) A Review on Software Fault Detection Mechanisms and Fault Prevention Mechanisms in Networks. *International Journal of Wireless and Ad Hoc Communication*, **6**, 34-42. https://doi.org/10.54216/ijwac.060203

[2] 谢瑞生. 软件失效原因分析[J]. 软件可靠性与评测技术, 2009, 27(3): 13-19；

[3] 赵昶宇, 刘云卿. 浅谈软件失效模式与影响分析[J]. 科技与创新, 2018(21): 1-2, 5.

[4] 孟令中, 王航, 薛云志, 武斌, 马兰. 软件失效模式的自动生成方法研究[J]. 计算机科学与探索, 2018, 12(11): 1758-1766.

[5] 钱丽, 胡俊, 王美荣, 沈桂芳, 陈平. 基于关联维计算的软件失效混沌识别研究[J]. 山东理工大学学报(自然科学版), 2018, 32(1): 21-25.

[6] 郭世杰, 余祥, 李勇, 司徒凌云. 基于知识图谱的软件测试知识检索方法研究[C]//中国指挥与控制学会. 第十三届中国指挥控制大会论文集(上册). 北京: 兵器工业出版社, 2025: 779-785.

[7] 李晓雪, 丁佐华, 胡觉亮. 测试方法对软件失效数据影响的实验分析[J]. 浙江理工大学学报, 2014, 31(7): 429-433.