ICS 35.080

VDI-RICHTLINIEN

Februar 2021
February 2021

VEREIN DEUTSCHER INGENIEURE

Softwarezuverlässigkeit
Software reliability

VDI 4011

Ausg. deutsch/englisch
Issue German/English

Die deutsche Version dieser Richtlinie ist verbindlich.

The German version of this standard shall be taken as authoritative. No guarantee can be given with respect to the English translation.

Inhalt Seite

Contents Page

VDI-Gesellschaft Produkt- und Prozessgestaltung (GPP)
Fachbereich Sicherheit und Zuverlässigkeit

VDI-Handbuch Zuverlässigkeit

Vorbemerkung

Der Inhalt dieser Richtlinie ist entstanden unter Beachtung der Vorgaben und Empfehlungen der Richtlinie VDI 1000.

Allen, die ehrenamtlich an der Erarbeitung dieser Richtlinie mitgewirkt haben, sei gedankt.

Weitere aktuelle Informationen sind im Internet abrufbar unter www.vdi.de/4011.

Einleitung

Es kann davon ausgegangen werden, dass jede hinreichend komplexe Software eine unbekannte Anzahl von Fehlern beinhaltet. Es existieren zwar Verfahren zur Schätzung der Anzahl dieser Fehler aus Erfahrungswerten, allerdings kann sie nicht mit Sicherheit bestimmt werden. Ebenso können nicht mit Sicherheit alle Fehler in einer Software lokalisiert werden. Um Zuverlässigkeitsanforderungen an Software gewährleisten zu können, ist daher in den meisten Fällen eine Kombination verschiedener Maßnahmen und Methoden notwendig. Diese Richtlinie gibt einen Überblick über entsprechende Methoden und Kriterien zur Entwicklung zuverlässiger Software, zur Bewertung der Softwarezuverlässigkeit sowie zur Bewahrung der Softwarezuverlässigkeit bei Modifikationen.

Das Verhalten einer Software bei der Ausführung, und damit auch die Art und Häufigkeit, mit der sich latent vorhandene Fehler manifestieren, ist abhängig von den Betriebsbedingungen (Nutzerverhalten, Eingabeparameter, Umgebung wie das Betriebssystem, Speicherverlauf, parallel laufende Anwendungen etc.). Diese hängen im Detail von Zufallsgrößen (Daten, Anforderungen, Benutzereingaben und deren zeitliche Abfolge etc.) ab. Daher wird Softwarezuverlässigkeit im Rahmen dieser Richtlinie aufgefasst als die Wahrscheinlichkeit, dass Software unter festgelegten Betriebsbedingungen und innerhalb eines festgelegten Zeitintervalls ihre geforderte Funktion erfüllt. Eine Nichterfüllung der geforderten Funktion wird als ein Softwareversagen bezeichnet.

Die in einer Software vorhandenen Fehler können u.a. logische Fehler hinsichtlich der Anforderungen, Spezifikationsfehler (das heißt, Anforderun-

Preliminary note

The content of this standard has been developed in strict accordance with the requirements and recommendations of the standard VDI 1000.

We wish to express our gratitude to all honorary contributors to this standard.

Further current information is available on the Internet at www.vdi.de/4011.

Introduction

It can be assumed that any sufficiently complex software contains an unknown number of errors. Although there are several experience-based procedures for estimating the amount of residual errors, they cannot be determined with certainty. Similarly, it is not possible to locate all errors in a software with certainty. In most cases, a combination of different measures and methods is therefore necessary to guarantee reliability requirements for software. This standard provides an overview of appropriate methods and criteria for developing reliable software, assessing software reliability, and maintaining software reliability in case of modifications.

The behaviour of a software during execution, and thus the type and frequency with which latent errors manifest themselves, depends on the operating conditions (user behaviour, input parameters, environment such as the operating system, memory history, applications running in parallel, etc.). These conditions depend in detail on random variables (data, requirements, user input and their temporal sequence, etc.). For this reason, software reliability in the context of this standard is understood as the probability that software will perform its required function under specified operating conditions and within a specified time interval. A failure to perform the required function is referred to as a software failure.

The errors present in a software can be, among other things, logical errors with regard to the requirements, specification errors (i.e. requirements

gen wurden falsch beschrieben), systematische Fehler im Design oder der Implementierung, Entwurfsfehler in der Lösung, Implementierungsfehler im Einzelnen, Dokumentationsfehler oder auch Fehler im Laufzeitsystem sein. Hierauf wird im Folgenden im Detail eingegangen.

Eine Zuverlässigkeitsanalyse von Software ist schwierig, da sich zum einen Software, im Gegensatz zu physikalischen bzw. Hardwaresystemen, diskontinuierlich verhält und zum anderen in vielen Fällen keine genauen, auf empirischen Untersuchungen beruhenden Daten und Aussagen über Zuverlässigkeitskenngrößen, z.B. die Ausfallrate λ, von Software erhoben werden können. Hierbei spielt u.a. die Schwierigkeit, den Betriebsbedingungen entsprechende Testfälle unabhängig voneinander generieren zu können, eine Rolle.

Eine hohe Zuverlässigkeit von Softwaresystemen wird durch die Anwendung und das Zusammenspiel verschiedener Maßnahmen erreicht. Diese sind:

a) organisatorische Maßnahmen (Normen, Vorgehensmodelle etc.)

   Organisatorische Maßnahmen zur Erhöhung der Softwarezuverlässigkeit haben das Ziel, definierte und kontrollierbare Prozesse zu gestalten.

b) konstruktive Maßnahmen (Modularisierung, Wiederverwendung, Verwendung von Beschreibungsverfahren wie UML und entsprechenden Werkzeugen sowie fehlervermeidende Programmiersprachen und zugehörige Laufzeitsysteme etc.)

   Konstruktive Maßnahmen zur Erhöhung der Softwarezuverlässigkeit haben das Ziel, eine strukturierte Vorgehensweise in der Entwicklung zu ermöglichen.

c) analytische Maßnahmen (Softwareprüfung, Code-Review etc.)

   Analytische Maßnahmen zur Erhöhung der Softwarezuverlässigkeit haben das Ziel, möglichst viele in der Software enthaltene Fehler und Mängel vor Inbetriebnahme zu entdecken.

Die Betrachtung der Fehlerratenentwicklung (Reifegrad) im Betrieb allein erlaubt keine Aussage über zukünftige Fehlereintritte, da Änderungen in der Nutzung oder von Eingabeparametern und Updates die Eintrittswahrscheinlichkeit von Fehlern unvorhersehbar ändern können. Allerdings ist eine solche Analyse geeignet, die Kritikalität von Softwarebausteinen zu bewerten und Rückschlüsse für die Entwicklung im Sinne von Aufwandsplanung für die Entwicklung und den Test zu erhalten.

were described incorrectly), systematic errors in the design or implementation, design errors in the solution, implementation errors in detail, documentation errors or errors in the runtime system. These are discussed in detail below.

A reliability analysis of software is difficult for two reasons. First one is that software behaves discontinuously, in contrast to physical or hardware systems. On the other hand, in many cases it is not possible to obtain precise data and statements about reliability parameters, e.g. the failure rate λ, that can be collected from software. The difficulty of being able to independently generate test cases corresponding to the operating conditions plays a role here, among other things.

A high reliability of software systems is achieved by the application and interaction of different measures. These are:

a) organisational measures (standards, process models, etc.)

   Organisational measures to increase software reliability have the goal of designing defined and controllable processes.

b) constructive measures (modularisation, reuse, use of description methods such as UML and corresponding tools, as well as error-avoiding programming languages and associated runtime systems, etc.)

   Constructive measures to increase software reliability aim to enable a structured approach to development.

c) analytical measures (software testing, code review, etc.)

   Analytical measures to increase software reliability are intended to detect as many errors and defects contained in the software as possible before it is put into operation.

The consideration of the error rate development (maturity level) in operation alone does not allow any statement about future error occurrences, since changes in use or in input parameters and updates can change the probability of errors occurring in an unforeseeable way. However, such an analysis is suitable for evaluating the criticality of software components and for drawing conclusions for development in terms of planning the effort required for development and testing.

Diese Maßnahmen beeinflussen sich gegenseitig und verfolgen die folgenden Ziele:

a) Fehlerabwehr

   Entstehung von Softwarefehler über einen verbesserten Entwicklungsprozess und konstruktive Maßnahmen möglichst vermeiden.

b) Fehleroffenbarung und -korrektur

   Vor der Inbetriebnahme möglichst viele der enthaltenen Softwarefehler über ein sicherheitsbezogenes Vorgehensmodell aufdecken.

c) Fehlerauswirkungsausschluss

   Gefährliche Auswirkungen von Softwarefehlern während des Betriebs verhindern.

1 Anwendungsbereich

Diese Richtlinie ist branchenübergreifend anwendbar, für alle Domänen gültig und bezieht sich nicht ausschließlich auf eingebettete Systeme. Generell wurde sie konzipiert mit dem Ziel, auf Systeme angewendet zu werden, bei denen Software eine Wertschöpfung darstellt.

Sie richtet sich grundsätzlich an Ingenieure, insbesondere an Personen, die an der Entwicklung von Software beteiligt sind (beispielsweise Softwareentwickler, Systemarchitekten, (technische) Projektleiter), aber auch an Gutachter sowie Anwender, Auftraggeber, Nutzer oder Kunden.

Die Richtlinie ist als Empfehlung gedacht, die Softwarezuverlässigkeit planbarer und greifbarer macht.

Die funktionale Sicherheit und IT-Sicherheit werden in dieser Richtlinie nicht berücksichtigt. Zudem ist es nicht Ziel dieser Richtlinie, Lehrbücher oder vertiefende Fachartikel zu ersetzen. Eine allgemeine Übersicht über die bestehende Normenlandschaft und fehlende Aspekte soll nicht gegeben werden.

These measures influence each other and pursue the following objectives:

a) error prevention

   Avoid the occurrence of software errors as far as possible through an improved development process and constructive measures.

b) error disclosure and correction

   Before commissioning, detect as many of the software errors contained in the system for example by using a safety-related procedure model.

c) fault effect exclusion

   Prevent dangerous effects of software errors during operation.

1 Scope

This standard applies across all industries, is valid for all domains, and does not apply exclusively to embedded systems. In general, it is designed to be applied to systems where software adds value.

It is basically addressed to engineers, in particular to persons involved in the development of software (e.g. software developers, system architects, (technical) project managers) but also to experts as well as users, clients, or customers.

The standard is intended as a recommendation that makes software reliability more predictable and concrete.

Functional safety and IT security are not considered in this standard. Furthermore, it is not the aim of this standard to replace textbooks or in-depth technical articles. A general overview of the existing standards landscape and missing aspects should not be given.