

	DIN IEC 60880 (VDE 0491-3-2)	DIN
	Diese Norm ist zugleich eine VDE-Bestimmung im Sinne von VDE 0022. Sie ist nach Durchführung des vom VDE-Präsidium beschlossenen Genehmigungsverfahrens unter der oben angeführten Nummer in das VDE-Vorschriftenwerk aufgenommen und in der „etz Elektrotechnik + Automation“ bekannt gegeben worden.	VDE
ICS 27.120.20 Ersatz für DIN IEC 60880:1987-08 und DIN IEC 60880-2 (VDE 0491-3-2):2001-12 Siehe jedoch Beginn der Gültigkeit Kernkraftwerke – Leittechnik für Systeme mit sicherheitstechnischer Bedeutung – Softwareaspekte für rechnerbasierte Systeme zur Realisierung von Funktionen der Kategorie A (IEC 60880:2006) Nuclear power plants – Instrumentation and control systems important to safety – Software aspects for computer-based systems performing category A functions (IEC 60880:2006) Centrales nucléaires de puissance – Instrumentation et contrôle-commande importants pour la sûreté – Aspects logiciels des systèmes programmés réalisant des fonctions de catégorie A (CEI 60880:2006) Gesamtumfang 108 Seiten DKE Deutsche Kommission Elektrotechnik Elektronik Informationstechnik im DIN und VDE		

Beginn der Gültigkeit

Diese Norm gilt ab 2007-08-01

Daneben dürfen DIN IEC 60880:1987-08 und **DIN IEC 60880-2 (VDE 0491-3-2):2001-12** noch bis 2008-12-31 angewendet werden.

Nationales Vorwort

Vorausgegangener Norm-Entwurf: E DIN IEC 60880 (VDE 0491-3-2):2004-07.

Für diese Norm ist das nationale Arbeitsgremium GUK 967.1 „Leittechnik für kerntechnische Anlagen“ der DKE Deutsche Kommission Elektrotechnik Elektronik Informationstechnik im DIN und VDE (<http://www.dke.de>) zuständig.

Die enthaltene IEC-Publikation wurde vom SC 45A „Instrumentation and control of nuclear facilities“ erarbeitet. Diese Norm, deren Anwendung international empfohlen wird, gibt das höchstmögliche Maß internationaler Übereinstimmung für das behandelte Sachgebiet wieder. Bei Fragen der Interpretation von Aussagen wird empfohlen, ergänzend auf die zugrunde liegende, in englischer und französischer Sprache erschienene IEC-Publikation zurückzugreifen. Die Auslegung kerntechnischer Anlagen in Deutschland wird wesentlich durch die Strahlenschutzverordnung, die BMI-Kriterien, die KTA-Regeln und die RSK-Leitlinien bestimmt.

Das IEC-Komitee hat entschieden, dass der Inhalt dieser Publikation bis zu dem auf der IEC-Website unter „<http://webstore.iec.ch>“ mit den Daten zu dieser Publikation angegebenen Datum (maintenance result date) unverändert bleiben soll. Zu diesem Zeitpunkt wird entsprechend der Entscheidung des Komitees die Publikation

- bestätigt,
- zurückgezogen,
- durch eine Folgeausgabe ersetzt oder
- geändert.

In der Reihe VDE 0491 sind als Normen erschienen:

- **DIN IEC 61226 (VDE 0491-1):2005-09**, *Kernkraftwerke – Leittechnische Systeme mit sicherheitstechnischer Bedeutung – Kategorisierung leittechnischer Funktionen (IEC 61226:2005).*
- **DIN IEC 61513 (VDE 0491-2):2002-10**, *Kernkraftwerke – Leittechnik für Systeme mit sicherheitstechnischer Bedeutung – Allgemeine Systemanforderungen (IEC 61513:2001).*
- **DIN IEC 61513/A100 (VDE 0491-2/A100):2005-02**, *Kernkraftwerke – Leittechnik für Systeme mit sicherheitstechnischer Bedeutung – Allgemeine Systemanforderungen – Nationaler Anhang ND (informativ): Mitgeltende Festlegungen aus anderen IEC-Normen.*
- **DIN IEC 60880 (VDE 0491-3-2):2007-08**, *Kernkraftwerke – Leittechnik für Systeme mit sicherheitstechnischer Bedeutung – Softwareaspekte für rechnerbasierte Systeme zur Realisierung von Funktionen der Kategorie A (IEC 60880:2006).*
- **DIN IEC 62138 (VDE 0491-3-3):2004-09**, *Kernkraftwerke – Leittechnik für Systeme mit sicherheitstechnischer Bedeutung – Softwareaspekte für rechnerbasierte Systeme zur Realisierung von Funktionen der Kategorien B oder C (IEC 62138:2004).*
- **DIN IEC 62117 (VDE 0491-4):2001-10**, *Kernkraftwerke – Druckwasserreaktoren – Überwachung ausreichender Kühlung des Kerns im abgeschalteten Zustand des Reaktors (IEC 62117:1999).*
- **DIN IEC 61839 (VDE 0491-5):2002-09**, *Kernkraftwerke – Auslegung von Warten – Analyse und Zuordnung der Funktionen (IEC 61839:2000).*
- **DIN IEC 62241 (VDE 0491-5-2):2006-05**, *Kernkraftwerke – Hauptwarte – Funktionen zur Meldung und Anzeige von Störungen (IEC 62241:2004).*
- **DIN IEC 60568 (VDE 0491-6):2006-11**, *Kernkraftwerke – Instrumentierung mit sicherheitstechnischer Bedeutung – Kerninstrumentierung zur Messung der Neutronenflussdichte in Leistungsreaktoren (IEC 60568:2006).*
- **DIN IEC 60709 (VDE 0491-7):2006-05**, *Kernkraftwerke – Leittechnische Systeme mit sicherheitstechnischer Bedeutung – Physikalische und elektrische Trennung (IEC 60709:2004).*

Als Norm-Entwürfe liegen vor:

- E DIN IEC 60987 (VDE 0491-3-1):2006-09, *Kernkraftwerke – Leittechnische Systeme mit sicherheitstechnischer Bedeutung – Anforderungen an die Hardware-Auslegung rechnerbasierter Systeme* (IEC 45A/623/CDV:2006).
- E DIN IEC 62340 (VDE 0491-10):2006-04, *Kernkraftwerke – Leittechnische Systeme mit sicherheitstechnischer Bedeutung – Versagen gemeinsamer Ursache (CCF)* (IEC 45A/601/CDV:2005).
- E **DIN IEC 60671 (VDE 0491-100)**:2006-05, *Kernkraftwerke – Leittechnische Systeme mit sicherheitstechnischer Bedeutung – Sicherstellung der Betriebsbereitschaft* (IEC 45A/604/CDV:2005).

Änderungen

Gegenüber DIN IEC 60880:1987-08 und **DIN IEC 60880-2 (VDE 0491-3-2):2001-12** wurden folgende Änderungen vorgenommen:

- a) Zusammenführung von DIN IEC 60880:1987-08 und DIN IEC 60880-2 (VDE 0491 Teil 3-2):2001-12;
- b) vollständige Bearbeitung und Aktualisierung der behandelten Softwareaspekte;
- c) Behandlung des gesamten Lebenszyklus nach DIN IEC 61513 (VDE 0491 Teil 2):2002-10;
- d) mit Ausnahme der neuen Abschnitte 6.3 „Periodische Prüfungen“, 9 „Softwareaspekte der Systemintegration“, 10.2 „Systemvalidierung“ und 12.4 „Operateurtraining“, welche Systemanforderungen der IEC 61513 ergänzen und daher bei einer künftigen Überarbeitung dort einzuarbeiten sind, Konzentration auf Softwareaspekte;
- e) bezüglich der weiteren Systemaspekte siehe DIN IEC 61513 (VDE 0491 Teil 2):2002-10;
- f) zur Aktualisierung von Verweisen der DIN IEC 61513 (VDE 0491 Teil 2):2002-10 auf die ersetzten Ausgaben der IEC 60880 und IEC 60880-2 siehe den neuen Anhang J.

Frühere Ausgaben

DIN IEC 60880:1987-08

DIN IEC 60880-2 (VDE 0491-3-2):2001-12

Nationaler Anhang NA (informativ)

Zusammenhang mit Europäischen und Internationalen Normen

Für den Fall einer undatierten Verweisung im normativen Text (Verweisung auf eine Norm ohne Angabe des Ausgabedatums und ohne Hinweis auf eine Abschnittsnummer, eine Tabelle, ein Bild usw.) bezieht sich die Verweisung auf die jeweils neueste gültige Ausgabe der in Bezug genommenen Norm.

Für den Fall einer datierten Verweisung im normativen Text bezieht sich die Verweisung immer auf die in Bezug genommene Ausgabe der Norm.

Eine Information über den Zusammenhang der zitierten Normen mit den entsprechenden Deutschen Normen ist in Tabelle NA.1 wiedergegeben.

Tabelle NA.1

Europäische Norm	Internationale Norm	Deutsche Norm	Klassifikation im VDE-Vorschriftenwerk
EN 50129:2003	–	DIN EN 50129 (VDE 0831-129):2003-12	VDE 0831-129
–	IEC 60671:1980 wird ersetzt durch IEC 60671:200x	– DIN IEC 60671 (VDE 0491-100):200	– VDE 0491-100
–	IEC 60987:1989 wird ersetzt durch IEC 60987:200x	– DIN IEC 60987 (VDE 0491-3-1):200x	– VDE 0491-3-1
EN 61069-2:1994	IEC 61069-2:1993	DIN EN 61069-2:1994-10	–
EN 61131-3:2003	IEC 61131-3:2003	DIN EN 61131-3:2003-12	–
–	IEC 61226:2005	DIN IEC 61226 (VDE 0491-1):2005-09	VDE 0491-1
–	Vorgänger: IEC 61226:1993	Vorgänger: DIN IEC 61226 (VDE 0491-1):1997-07	VDE 0491-1
EN 61508-1:2001	IEC 61508-1:1998 + Corr. 1:1999	DIN EN 61508-1 (VDE 0803-1):2002-11	VDE 0803-1
EN 61508-2:2001	IEC 61508-2:2000	DIN EN 61508-2 (VDE 0803-2):2002-12	VDE 0803-2
EN 61508-3:2001	IEC 61508-3:1998 + Corr. 1:1999	DIN EN 61508-3 (VDE 0803-3):2002-12	VDE 0803-3
EN 61508-4:2001	IEC 61508-4:1998 + Corr. 1999	DIN EN 61508-4 (VDE 0803-4):2002-11	VDE 0803-4
–	IEC 61513:2001	DIN IEC 61513 (VDE 0491-2):2002-10 + DIN IEC 61513/A100 (VDE 0491-2/A100):2005-02	VDE 0491-2 VDE 0491-2/A100
–	IEC 62138:2004	DIN IEC 62138 (VDE 0491-3-3):2004-09	VDE 0491-3-3
–	ISO/IEC 9126-1:2001	–	–
–	ISO/IEC TR 9126-2:2003	–	–
–	ISO/IEC TR 9126-3:2003	–	–
–	ISO/IEC TR 9126-4:2004	–	–

Europäische Norm	Internationale Norm	Deutsche Norm	Klassifikation im VDE-Vorschriftenwerk
–	IAEA Guide NS-G-1.2	–	–
–	IAEA Guide NS-G-1.3	–	–

Nationaler Anhang NB (informativ)

Literaturhinweise

DIN EN 50129 (VDE 0831-129):2003-12, *Bahnanwendungen – Telekommunikationstechnik, Signaltechnik und Datenverarbeitungssysteme – Sicherheitsrelevante elektronische Systeme für Signaltechnik; Deutsche Fassung EN 50129:2003.*

DIN EN 61069-2:1994-10, *Leittechnik für industrielle Prozesse – Ermittlung der Systemeigenschaften zum Zweck der Eignungsbeurteilung eines Systems – Teil 2: Methodik der Eignungsbeurteilung (IEC 61069-2:1993); Deutsche Fassung EN 61069-2:1994.*

DIN EN 61131-3:2003-12, *Speicherprogrammierbare Steuerungen – Teil 3: Programmiersprachen (IEC 61131-3:2003); Deutsche Fassung EN 61131-3:2003.*

DIN EN 61508-1 (VDE 0803-1):2002-11, *Funktionale Sicherheit sicherheitsbezogener elektrischer/elektronischer/programmierbarer elektronischer Systeme – Teil 1: Allgemeine Anforderungen (IEC 61508-1:1998 + Corrigendum 1999); Deutsche Fassung EN 61508-1:2001.*

DIN EN 61508-2 (VDE 0803-2):2002-12, *Funktionale Sicherheit sicherheitsbezogener elektrischer/elektronischer/programmierbarer elektronischer Systeme – Teil 2: Anforderungen an sicherheitsbezogene elektrische/elektronische/programmierbare elektronische Systeme (IEC 61508-2:2000); Deutsche Fassung EN 61508-2:2001.*

DIN EN 61508-3 (VDE 0803-3):2002-12, *Funktionale Sicherheit sicherheitsbezogener elektrischer/elektronischer/programmierbarer elektronischer Systeme – Teil 3: Anforderungen an Software (IEC 61508-3:1998 + Corrigendum 1999); Deutsche Fassung EN 61508-3:2001.*

DIN EN 61508-4 (VDE 0803-4):2002-11, *Funktionale Sicherheit elektrischer/elektronischer/programmierbarer elektronischer sicherheitsbezogener Systeme – Teil 4: Begriffe und Abkürzungen (IEC 61508-4:1998 + Corrigendum 1999); Deutsche Fassung EN 61508-4:2001.*

E DIN IEC 60671 (VDE 0491-100):2006-05, *Kernkraftwerke – Leittechnische Systeme mit sicherheitstechnischer Bedeutung – Sicherstellung der Betriebsbereitschaft (IEC 45A/604/CDV:2005).*

E **DIN IEC 60987 (VDE 0491-3-1):2006-09**, *Kernkraftwerke – Leittechnische Systeme mit sicherheitstechnischer Bedeutung – Anforderungen an die Hardware-Auslegung rechnerbasierter Systeme (IEC 45A/623/CDV:2006).*

DIN IEC 61226 (VDE 0491-1):2005-09, *Kernkraftwerke – Leittechnische Systeme mit sicherheitstechnischer Bedeutung – Kategorisierung leittechnischer Funktionen (IEC 61226:2005).*

DIN IEC 61513 (VDE 0491-2):2002-10, *Kernkraftwerke – Leittechnik für Systeme mit sicherheitstechnischer Bedeutung – Allgemeine Systemanforderungen (IEC 61513:2001).*

DIN IEC 61513/A100 (VDE 0491-2/A100):2005-02, *Kernkraftwerke – Leittechnik für Systeme mit sicherheitstechnischer Bedeutung – Allgemeine Systemanforderungen – Nationaler Anhang ND (informativ): Mitgeltende Festlegungen aus anderen IEC-Normen.*

DIN IEC 62138 (VDE 0491-3-3):2004-09, *Kernkraftwerke – Leittechnik für Systeme mit sicherheitstechnischer Bedeutung – Softwareaspekte für rechnerbasierte Systeme zur Realisierung von Funktionen der Kategorien B oder C (IEC 62138:2004).*

Kernkraftwerke – Leittechnik für Systeme mit sicherheitstechnischer Bedeutung – Softwareaspekte für rechnerbasierte Systeme zur Realisierung von Funktionen der Kategorie A

Inhalt

	Seite
Einleitung.....	9
1 Anwendungsbereich	11
2 Normative Verweisungen.....	11
3 Begriffe.....	12
4 Formelzeichen und Abkürzungen	18
5 Allgemeine Anforderungen für Software-Projekte	19
5.1 Einleitung	19
5.2 Softwaretypen	21
5.3 Vorgehensweise zur Software-Entwicklung	21
5.4 Software-Projektmanagement	23
5.5 Software-Qualitätssicherungsplan	23
5.6 Konfigurationsmanagement.....	24
5.7 Zugriffsschutz.....	25
6 Software-Anforderungen.....	27
6.1 Spezifikation der Software-Anforderungen	27
6.2 Selbstüberwachung	27
6.3 Periodische Prüfungen	28
6.4 Dokumentation.....	28
7 Auslegung und Implementierung	29
7.1 Prinzipien der Auslegung und Realisierung.....	29
7.2 Sprachen und zugehörige Übersetzer und Werkzeuge.....	31
7.3 Detaillierte Empfehlungen.....	32
7.4 Dokumentation.....	34
8 Software-Verifizierung	34
8.1 Software-Verifizierungsvorgang.....	34
8.2 Software-Verifizierungstätigkeiten	35
9 Softwareaspekte der Systemintegration	39
9.1 Softwareaspekte des Systemintegrationsplans.....	39
9.2 Systemintegration	40
9.3 Verifizierung des integrierten Systems	40
9.4 Prozeduren zur Fehlerbehebung	41
9.5 Softwareaspekte zum Bericht über die Verifizierung des integrierten Systems	41
10 Softwareaspekte der Systemvalidierung	42
10.1 Softwareaspekte des Systemvalidierungsplans	42

	Seite
10.2 Systemvalidierung	42
10.3 Softwareaspekte im Bericht zur Systemvalidierung	42
10.4 Prozeduren zur Fehlerbehebung.....	43
11 Software-Modifizierung.....	43
11.1 Prozedur zur Modifizierungsanfrage	44
11.2 Prozedur zur Durchführung einer Software-Modifizierung	45
11.3 Software-Modifizierung nach Auslieferung	46
12 Softwareaspekte bei Installation und Betrieb	47
12.1 Installation der Software auf der Anlage	47
12.2 Zugriffsschutz der Software auf der Anlage	47
12.3 Anpassung der Software an die Bedingungen auf der Anlage	47
12.4 Operateurtraining.....	48
13 Vorkehrungen gegen CCF durch Softwarefehler	48
13.1 Allgemeines	48
13.2 Software-Auslegung gegen CCF.....	49
13.3 Ursachen und Auswirkungen von CCF durch Softwarefehler.....	50
13.4 Realisierung von Diversität.....	50
13.5 Abwägung der Vor- und Nachteile bei Verwendung von Diversität	51
14 Software-Werkzeuge für die Erstellung von Software.....	51
14.1 Allgemeines	51
14.2 Auswahl der Werkzeuge.....	52
14.3 Anforderungen an Werkzeuge.....	52
15 Qualifizierung vorgefertigter Software	57
15.1 Allgemeines	57
15.2 Allgemeine Anforderungen	57
15.3 Bewertungs- und Beurteilungsprozess.....	58
15.4 Anforderungen an die Systemintegration und Modifizierung vorgefertigter Software.....	66
Anhang A (normativ) Software-Sicherheitslebenszyklus und Details von Software-Anforderungen	67
Anhang B (normativ) Detaillierte Anforderungen und Empfehlungen für Auslegung und Realisierung.....	69
Anhang C (informativ) Beispiel für anwendungsorientiertes Software-Engineering (Software-Entwicklung mit anwendungsorientierten Sprachen)	82
Anhang D (informativ) Sprache, Übersetzer, Linkage-Editor	86
Anhang E (informativ) Software-Verifizierung und Prüfung.....	88
Anhang F (informativ) Typische Liste für Software-Dokumentation	95
Anhang G (informativ) Überlegungen zu CCF und Diversität.....	96
Anhang H (informativ) Werkzeuge zur Erstellung und Prüfung von Spezifikation, Auslegung und Realisierung.....	100
Anhang I (informativ) Anforderungen betreffend vorgefertigte Software.....	102
Anhang J (informativ) Zusammenhang zwischen IEC 61513 und dieser Norm.....	104

	Seite
Bild 1 – Tätigkeiten im System-Sicherheitslebenszyklus (nach IEC 61513)	19
Bild 2 – Softwarebezogene Tätigkeiten im System-Sicherheitslebenszyklus.....	20
Bild 3 – Entwicklungstätigkeiten im Software-Sicherheitslebenszyklus nach dieser Norm	22
Bild 4 – Übersicht des Qualifizierungsvorgangs von vorgefertigter Software (PDS)	59
Bild 5 – Beziehung der Bewertung und Analyse vorgefertigter Software mit dem Qualifizierungsplan des Systems, in das die vorgefertigte Software integriert ist.....	60
Bild C.1 – Lebenszyklus für anwendungsorientiertes Software-Engineering	85
 Tabelle 1 – Prozess- und Produktaspekte von Auslegung und Implementierung	 33
Tabelle J.1 – Zusammenhang zwischen IEC 61513 und dieser Norm.....	105
Tabelle J.2 – Abschnitte aus IEC 60880, die bei der nächsten Überarbeitung von IEC 61513 zu berücksichtigen sind	108

Einleitung

a) Technischer Hintergrund, hauptsächliche Inhalte und Organisation der Norm

Die Konstruktion softwarebasierter Leittechnik zum Einsatz für die Reaktorsicherheit ist in Bezug auf die zu erfüllenden Sicherheitsanforderungen eine besondere Herausforderung. Die in Kernkraftwerken für sicherheitstechnische Zwecke verwendete Software, deren unterstützte Funktionen oft nur in Notfällen erforderlich sind, muss vor Verwendung im Betrieb vollständig validiert und qualifiziert werden. Um die geforderte hohe Zuverlässigkeit zu erreichen, muss während des gesamten Lebenszyklus von den grundsätzlichen Anforderungen über die verschiedenen Entwicklungsphasen bis zu den Verifizierungs- und Validierungs-Prozeduren für Betrieb und Wartung große Sorgfalt aufgewendet werden. Es ist das Ziel dieser Norm, die diesbezüglichen Sicherheitsaspekte anzusprechen und Anforderungen zur Erreichung der notwendigen hohen Software-Qualität bereitzustellen.

Die erste Ausgabe dieser Norm von 1986 hat die bisher bei Hardware-Systemen angewendeten grundsätzlichen Sicherheitsprinzipien für die Nutzung digitaler Systeme – verteilte Mehrfachprozessor-Systeme und Systeme mit großem Zentralprozessor – in Sicherheitssystemen von Kernkraftwerken umgesetzt.

Die Ausgabe 1986 wurde von der Nuklearindustrie umfassend genutzt, um Anleitungen und Anforderungen für Software in Leittechnik von Sicherheitssystemen bereitzustellen.

Obwohl viele Anforderungen in der ersten Ausgabe nach wie vor Gültigkeit haben, gibt es signifikante Faktoren, die die Herausgabe einer überarbeiteten Fassung rechtfertigen:

- Seit 1986 wurde eine Anzahl von Normen veröffentlicht, die im Detail auf allgemeine Systemanforderungen (IEC 61513) und Hardware-Anforderungen (IEC 60987) eingehen, sowie eine Norm über Software für sicherheitstechnisch wichtige leittechnische Systeme, die Funktionen der Kategorie B oder Kategorie C (IEC 62138) durchführen. Zusätzlich erschien Teil 2 von IEC 60880 (IEC 60880-2). Schließlich wurde die Sicherheitsleitlinie 50-SG-D3 der IAEA überarbeitet und als NS-G-1.3 veröffentlicht.
- Die Softwaretechnologie hat inzwischen wesentliche Fortschritte gemacht.

Für diese Norm wurde größte Sorgfalt darauf verwendet, Transparenz in Bezug auf die erste Ausgabe sicherzustellen. Wo möglich wurde die Formulierung der Anforderungen beibehalten, anderenfalls wurden sie in nachvollziehbarer Weise erweitert. In gleicher Weise wurde IEC 60880-2, die sich mit Softwareaspekten zur Vermeidung von Ausfällen gemeinsamer Ursache, Verwendung von Software-Werkzeugen und vorgefertigter Software beschäftigt, integriert, sodass die nun vorliegende Norm alle anzusprechenden Software-Sicherheitsfragen abdeckt.

Diese Norm ist für die Anwendung durch Systementwickler, Anwender (EVU), Gutachter und Genehmigungsbehörden gedacht.

b) Einordnung der vorliegenden Norm in die Struktur der Normen des IEC/SC 45A

Auf IEC 60880 wird in IEC 61513 direkt Bezug genommen. IEC 61513 beschäftigt sich mit den Systemaspekten von hochzuverlässiger rechnerbasierter Leittechnik, die in Sicherheitssystemen von Kernkraftwerken verwendet wird.

IEC 60880 ist eine Norm der zweiten Ebene der Normen des IEC/SC 45A und behandelt Softwareaspekte leittechnischer Systeme, die Funktionen der Kategorie A ausführen.

Software für Funktionen der Kategorien B und C wird in IEC 62138 behandelt.

IEC 60880 und IEC 62138 decken zusammen das Gebiet von Softwareaspekten rechnerbasierter Systeme ab, die in Kernkraftwerken für die Durchführung sicherheitstechnisch wichtiger Funktionen verwendet werden.

Diese zweite Ausgabe der IEC 60880 ist in Zusammenhang mit IEC 60987 und IEC 61226 zu lesen, den einschlägigen Normen des SC 45A für Rechnerhardware und für Kategorisierung.

Näheres zur Struktur der Normen des IEC/SC 45A ist unter d) dieser Einleitung zu finden.

c) Empfehlungen und Einschränkungen der Anwendung der Norm

Es ist wichtig, darauf hinzuweisen, dass in dieser Norm keine zusätzlichen funktionalen Anforderungen für Sicherheitssysteme definiert werden.

Hinsichtlich folgender Aspekte wurden spezielle Anforderungen und Empfehlungen erstellt:

- 1) allgemeine Vorgehensweise zur Software-Entwicklung, um die Erstellung hochzuverlässiger Software unter Berücksichtigung der Wechselwirkung zwischen Hardware und Software sicherzustellen;
- 2) allgemeine Vorgehensweise zur Software-Verifizierung und zu den Softwareaspekten bei der Validierung eines rechnerbasierten Systems;
- 3) Prozeduren zur Änderungs- und Konfigurationskontrolle;
- 4) Anforderungen für die Verwendung von Werkzeugen;
- 5) Prozeduren für die Qualifizierung vorgefertigter Software.

Es ist zu berücksichtigen, dass sich die Softwaretechnologie nach wie vor rasch entwickelt und dass es in einer Norm wie dieser nicht möglich ist, Verweisungen auf alle modernen Entwicklungstechniken mit einzuschließen.

Um sicherzustellen, dass die Norm über längere Zeit unverändert gültig bleibt, wurde der Schwerpunkt nicht auf spezielle Software-Techniken gelegt, sondern auf die Festlegung von Prinzipien.

Wenn neu entwickelte Techniken zur Verfügung stehen, sollte es möglich sein, ihre Eignung durch die in dieser Norm enthaltenen Sicherheitsprinzipien zu beurteilen.

d) Beschreibung der Struktur der Normen des IEC/SC 45A und deren Beziehungen zu anderen Dokumenten einschließlich der Dokumente anderer Organisationen (IAEA, ISO)

Das Eingangsdokument der Normen des IEC/SC 45A stellt die Norm **IEC 61513** dar. Diese Norm befasst sich mit Anforderungen an sicherheitstechnisch wichtige leittechnische Systeme von Kernkraftwerken und strukturiert die Normen des IEC/SC 45A.

In **IEC 61513** wird direkt auf andere Normen des IEC/SC 45A verwiesen, die allgemeine Themen wie die Kategorisierung von Funktionen und die Klassifizierung von Systemen, Qualifizierung, Trennung von Systemen, Maßnahmen gegen Versagen gemeinsamer Ursache, Softwareaspekte rechnerbasierter Systeme, Hardwareaspekte rechnerbasierter Systeme und Auslegung von Warten behandeln. Die Normen, auf die direkt verwiesen wird, bilden die zweite Ebene und sollten zusammen mit **IEC 61513** als ein in sich konsistenter Satz von Dokumenten betrachtet werden.

Die SC-45A-Normen der dritten Ebene, auf die in **IEC 61513** nicht direkt verwiesen wird, beziehen sich auf spezielles Gerät, technische Methoden oder spezifische Aktivitäten. Üblicherweise können diese Dokumente, in denen auf Normen der Ebene 2 verwiesen wird, für sich allein benutzt werden.

Auf der vierten Ebene finden sich als Erweiterung der SC-45A-Normenreihe Technische Berichte, die der Information dienen und keine Normen darstellen.

Für **IEC 61513** wird ein Darstellungsformat verwendet, das dem der grundsätzlichen Sicherheitsnorm IEC 61508 ähnlich ist, mit einem Sicherheits-Lebenszyklus als Gesamtrahmen und einem System-Lebenszyklus als weiteren wichtigen Rahmen. Mit **IEC 61513** werden die allgemeinen Anforderungen (für Sicherheitssysteme unterschiedlicher industrieller Anwendungen) von **IEC 61508-1**, **-2** und **-4** für die Anwendung auf nuklearem Sektor umgesetzt. Die Einhaltung der **IEC 61513** erleichtert die Übereinstimmung mit IEC 61508, weil die Anforderungen der **IEC 61513** aus den Anforderungen der IEC 61508 für die Anwendung in der nuklearen Industrie entwickelt wurden. In diesem Rahmen entsprechen IEC 60880 und **IEC 62138** für den nuklearen Sektor der Norm **IEC 61508-3**.

IEC 61513 stützt sich für Themen der Qualitätssicherung sowohl auf ISO als auch auf IAEA 50-C-QA ab.

In den Normen des IEC/SC 45A werden die Prinzipien und grundsätzlichen Sicherheitsaspekte des „IAEA Code for the safety of nuclear power plants“ und der „IAEA safety series“, insbesondere der Anforderungen des NS-R-1 „Safety of Nuclear Power Plants: Design“ und des Safety Guide NS-G-1.3 „Instrumentation and control systems important to safety in Nuclear Power Plants“, umgesetzt und detailliert. Terminologie und Definitionen in den Normen des IEC/SC 45A sind mit den von der IAEA verwendeten konsistent.

1 Anwendungsbereich

Diese Norm stellt Anforderungen für die Software rechnerbasierter leittechnischer Systeme in Kernkraftwerken bereit, die entsprechend der Definition in IEC 61226 Funktionen der Kategorie A ausführen.

Nach den Definitionen in IEC 61513 sind leittechnische Systeme der Klasse 1 grundsätzlich dafür gedacht, Funktionen der Kategorie A auszuführen, sie können jedoch auch für Funktionen einer niedrigeren Kategorie verwendet werden. Die Systemanforderungen werden jedoch immer von den implementierten Funktionen bestimmt, die der höchsten Kategorie angehören.

Für Software leittechnischer Systeme, die entsprechend Definition in IEC 61226 nur Funktionen der Kategorie B oder C ausführen, sind die Anforderungen und Anleitungen in IEC 62138 anwendbar.

Die Anforderungen in dieser Norm dienen der Erstellung einer hochzuverlässigen Software. Dabei wird jede Stufe der Software-Entwicklung und Dokumentation angesprochen, einschließlich Anforderungsspezifikation, Auslegung, Realisierung, Verifizierung, Validierung und Betrieb.

Die in der Entwicklung dieser Anforderungen angewendeten Prinzipien umfassen:

- beste verfügbare Praxis;
- Top-Down-Auslegungsmethoden;
- Modularität;
- Verifizierung jeder Phase;
- klare Dokumentation;
- überprüfbare Dokumente;
- Validierungstests.

Zusätzliche Anleitungen und Informationen, wie die Anforderungen des Hauptteils dieser Norm erfüllt werden können, sind in den [Anhängen A bis I](#) zu finden.

2 Normative Verweisungen

Die folgenden zitierten Dokumente sind für die Anwendung dieses Dokuments erforderlich. Bei datierten Verweisungen gilt nur die in Bezug genommene Ausgabe. Bei undatierten Verweisungen gilt die letzte Ausgabe des in Bezug genommenen Dokuments (einschließlich aller Änderungen).

IEC 60671, *Periodic tests and monitoring of the protection system of nuclear reactors*

IEC 61069-2:1993, *Industrial-process measurement and control – Evaluation of system properties for the purpose of system assessment – Part 2: Assessment methodology*

IEC 61226, *Nuclear power plants – Instrumentation and control systems important for safety – Classification of instrumentation and control functions*

IEC 61508-4, *Functional safety of electrical/electronic/programmable electronic safety-related systems – Part 4: Definitions and abbreviations*

IEC 61513, *Nuclear power plants – Instrumentation and control for systems important to safety – General requirements for systems*

ISO/IEC 9126-1, *Software Engineering – Product quality*

IAEA Guide NS-G-1.2, *Safety Assessment and Verification for Nuclear Power Plants*

IAEA Guide NS-G-1.3, *Instrumentation and Control Systems Important to Safety in Nuclear Power Plants*

3 Begriffe

Für die Anwendung dieses Dokuments gelten folgende Begriffe.

3.1

Animation

(en: animation)

Prozess, durch den das in einer Spezifikation festgelegte Verhalten mit aktuellen Werten angezeigt wird, abgeleitet aus den Gleichungen für dieses Verhalten und einigen Eingangswerten

3.2

Anwendungsfunktion

(en: application function)

Funktion eines leittechnischen Systems, die eine Aufgabe in Verbindung mit dem gesteuerten Prozess und nicht mit dem Funktionieren des Systems selbst erfüllt

[IEC 61513, 3.1]

3.3

anwendungsorientierte Sprache

(en: application-oriented language)

Rechnersprache, die speziell für bestimmte Anwendungsarten entwickelt und von Personen verwendet wird, die Spezialisten in diesen Anwendungsarten sind

[IEC 62138, 3.3]

ANMERKUNG 1 Gerätefamilien benutzen üblicherweise anwendungsorientierte Sprachen, um damit eine leicht nutzbare Möglichkeit für die Anpassung des Geräts an spezifische Anforderungen zur Verfügung zu stellen.

ANMERKUNG 2 Anwendungsorientierte Sprachen können verwendet werden, um spezielle funktionale Anforderungen an ein leittechnisches System festzulegen oder Anwendungssoftware zu spezifizieren und/oder zu entwickeln.

ANMERKUNG 3 Beispiele: Sprachen für Funktionsblockdiagramme, Sprachen nach IEC 61131-3.

3.4

Anwendungssoftware^{N1)}

(en: application software)

Teil der Software eines leittechnischen Systems, durch den Anwendungsfunktionen realisiert werden

[IEC 61513, 3.2]

3.5

automatische Code-Erzeugung

(en: automated code generation)

Funktion automatischer Werkzeuge, die eine Transformation der anwendungsorientierten Sprache in eine für die Kompilierung oder Programmausführung geeignete Form ermöglicht

3.6

Kanal

(en: channel)

Anordnung von miteinander verbundenen Komponenten innerhalb eines Systems, die ein Ausgangssignal erzeugen. Die Definition Kanal gilt bis an die Stelle, wo das einzelne Ausgangssignal mit Signalen anderer Kanäle verknüpft wird, wie z. B. einem Anzeige Kanal oder einem Betätigungskanal.

[IAEA NS-G-1.3, Glossary]

3.7

Code-Kompaktierung

(en: code compaction)

zweckmäßige Reduzierung des Speicherbedarfs für ein Programm durch Eliminierung von redundanten oder unwesentlichen Instruktionen

^{N1)} Nationale Fußnote: Siehe auch Systemsoftware (3.41) und Betriebssoftware (3.26).

3.8

Versagen gemeinsamer Ursache

CCF

(en: common cause failure – CCF)

Versagen von zwei oder mehreren Strukturen, Systemen oder Komponenten infolge eines einzelnen spezifischen Ereignisses oder Grundes

[IAEA NS-G-1.3, Glossary]

3.9

Rechner

(en: computer)

programmierbare Funktionseinheit, die aus einem oder mehreren Prozessoren und peripherem Gerät besteht und die durch interne Programme gesteuert wird und wesentliche Rechnungen durchführen kann, einschließlich umfangreicher arithmetischer oder logischer Rechengänge, ohne menschlichen Eingriff während eines Rechenlaufs

[ISO 238/1]

ANMERKUNG Ein Rechner kann sowohl ein einzelner Rechner sein als auch eine Zusammenschaltung verschiedener Einheiten.

3.10

Rechnerprogramm

(en: computer program)

Satz geordneter Instruktionen und Daten, in dem Rechengänge in einer für den Rechner geeigneten Form spezifiziert werden

3.11

rechnerbasiertes System

(en: computer-based system)

leittechnisches System, dessen Funktionen meistens von Mikroprozessoren, programmierten elektronischen Einheiten oder Rechnern abhängen oder durch die Verwendung derartiger Gerätschaften zur Gänze durchgeführt werden

[IEC 61513, 3.10]

ANMERKUNG Entspricht den Begriffen „digitales System“, „softwarebasiertes System“ und „programmierbares System“.

3.12

Daten

(en: data)

Darstellung von Information oder Anweisungen in einer für die Übertragung, Auswertung oder Verarbeitung mittels Rechnern geeigneten Weise

[IEEE 610, modifiziert]

ANMERKUNG Daten, die zur Definition von Parametern und zur Durchführung von Anwendungs- und Servicefunktionen im System dienen, werden „Applikationsdaten“ genannt.

3.13

Konzept der gestaffelten Verteidigung^{N2)}

DiD

(en: defence in depth – DiD)

Anwendung von mehr als einer Schutzmaßnahme zum Erreichen eines gegebenen Sicherheitsziels, sodass dieses erreicht wird, auch wenn eine der Schutzmaßnahmen versagt

[IAEA Safety Glossary]

3.14

Diversität^{N3)}

(en: diversity)

Vorhandensein von zwei oder mehreren unterschiedlichen Verfahren oder Mitteln, um ein bestimmtes Ziel zu erreichen. Diversität wird insbesondere als Schutzmaßnahme gegen CCF-Versagen eingesetzt. Sie kann erreicht werden, indem physikalisch unterschiedliche Systeme eingesetzt werden oder durch funktionale Diversität, bei der gleichartige Systeme ein bestimmtes Ziel über unterschiedliche Verfahren erreichen.

3.15

dynamische Analyse

(en: dynamic analysis)

Prozess der Bewertung eines Systems oder einer Komponente aufgrund des dynamischen Verhaltens im Betrieb. Gegenstück zur statischen Analyse

[IEEE 610]

3.16

Versagen

Ausfall^{N4)}

(en: failure)

Abweichen des vorliegenden Verhaltens von dem beabsichtigten Verhalten

[IEC 61513, 3.21, modifiziert]

3.17

Fehler

(en: fault)

Mangel an einer Hardware-, Software- oder Systemkomponente

[IEC 61513, 3.22]

3.18

Fehlertoleranz

(en: fault tolerance)

Fähigkeit eines Systems, trotz des Vorhandenseins einer begrenzten Anzahl von Hardware- oder Softwarefehlern kontinuierlich korrekt weiterzuarbeiten

^{N2)} Nationale Fußnote: Für dieses Auslegungsprinzip hat sich inzwischen der englische Begriff „defence in depth“ (DiD) etabliert. Im Folgenden wird die Abkürzung „DiD“ verwendet. Das DiD-Konzept ist durch die Anwendung gestufter Maßnahmen gegen die Auswirkungen versagenauslösender Ereignisse gekennzeichnet. In IAEA 50-C-D sind vier Stufen definiert, die von Maßnahmen bei der Abweichung vom Normalbetrieb (Stufe 1) bis zur Verminderung der Konsequenzen schwerer Störfälle (Stufe 4) reichen. Je nach Anforderung können innerhalb einer Maßnahmenstufe redundante oder auch diversitäre Maßnahmen getroffen werden.

^{N3)} Nationale Fußnote: Auch „diversitäre Redundanz“ genannt.

^{N4)} Nationale Fußnote: Wenn Hardware oder Teilsysteme (z. B. ein Strang eines redundanten Systems) betroffen sind, wird „failure“ mit „Ausfall“ übersetzt.

3.19

funktionale Diversität

(en: functional diversity)

Anwendung der Diversität auf der Funktionsebene (z. B. Ableitung eines Abschaltkriteriums sowohl aus Druck- als auch aus Temperaturgrenzwerten)

3.20

allgemeine Sprache

(en: general-purpose language)

Rechnersprache, die für alle Arten von Anwendungen ausgelegt ist

[IEC 62138, 3.17]

ANMERKUNG 1 Die Systemsoftware von Gerätefamilien wird üblicherweise über die Anwendung allgemeiner Sprachen realisiert.

ANMERKUNG 2 Beispiele: ADA, C, Pascal.

3.21

menschliches Fehlverhalten

(en: human error)

Handlung eines Menschen, die eine unbeabsichtigte Folge hat

3.22

initialisieren

(en: initialise)

Setzen von Zählern, Schaltern, Adressen oder Inhalten von Speichern auf Null oder andere Startwerte, zu Beginn oder an bestimmten Punkten des Betriebs eines Rechnerprogramms

3.23

Integrationstests

(en: integration tests)

Tests, die während des Integrationsvorgangs von Hardware/Software vor Validierung des Rechnersystems durchgeführt werden, um die Kompatibilität der Software und Hardware des Rechners zu verifizieren

3.24

Bibliothek

(en: library)

Sammlung von Software-Elementen, die miteinander in Beziehung stehen und gruppiert sind, aber zur Integration in das endgültige Software-Produkt individuell ausgewählt werden

3.25

Multiversions-Software

(en: n-version software)

(en: multi-version software)

Satz unterschiedlicher Programme, die entwickelt wurden, um einer gemeinsamen Anforderung und einer gemeinsamen Abnahmeprüfung zu genügen. Diese Versionen laufen gleichzeitig und voneinander unabhängig ab, üblicherweise in redundanter Hardware. Identische Eingangssignale von Prüfsystemen oder sich entsprechende Eingangssignale in redundanten Systemen werden verwendet. Zur Entscheidung bei unterschiedlichen/widersprüchlichen Ausgangssignalen der verschiedenen Versionen werden vorbestimmte Strategien eingesetzt, wie z. B. Mehrheitsentscheidungen.

3.26

Betriebssoftware

(en: operational system software)

Software, die auf dem Zielprozessor während dessen Betriebs läuft, wie z. B. Eingangs-/Ausgangs-Treiber und -Dienste, Interrupt-Management, Scheduler, Kommunikationstreiber, anwendungsorientierte Bibliotheken, Online-Diagnostik, Redundanz-Management, „Graceful-Degradation-Management“^{N5)}

3.27

anzunehmende versagenauslösende Ereignisse

AVE

(en: postulated initiating events – PIE)

Ereignisse, die zu den zu betrachtenden betrieblichen Vorkommnissen oder Auslegungsstörfällen und deren Folgewirkungen führen

[IEC 61513, 3.41]

3.28

vorgefertigte Software

(en: pre-developed software – PDS)

Software, die bereits vorgefertigt, als kommerzielles oder gesetzlich geschütztes Produkt verfügbar und für den Einsatz in einem rechnerbasierten System vorgesehen ist

[IEC 62138, 3.24 modifiziert]

3.29

Redundanz

(en: redundancy)

Beistellung unterschiedlicher (identischer oder verschiedener) Strukturen, Systeme oder Komponenten, so dass jedes, unabhängig vom Betriebszustand oder Versagen irgendeines anderen, die geforderte Funktion ausüben kann

[IAEA NS-G-1.3, Glossary]

3.30

aufgabenbasierte Zugriffskontrolle

(en: role-based access control)

auf Anweisungen basierende Zugriffskontrolle, durch die erlaubter Zugriff von Benutzern auf Objekte (Funktionen, Daten) für Benutzergruppen mit identischen Aufgaben (anstelle einzelner Benutzer) geregelt wird

3.31

Sicherheitsfunktion

(en: safety function)

Funktion mit speziellem für die Sicherheit zu erreichenden Ziel

[IAEA NS-R-1, Glossary]

3.32

Sicherheitssystem

(en: safety system)

sicherheitstechnisch wichtiges System, das die Sicherheitsabschaltung des Reaktors oder die Nachwärmeabfuhr aus dem Kern oder die Begrenzung der Folgen von anzunehmenden betrieblichen Ereignissen und Auslegungsstörfällen sicherstellt

[IAEA NS-R-1, Glossary]

^{N5)} Nationale Fußnote: Unter „Graceful Degradation“ wird die Minderung der Leistungsfähigkeit eines Systems verstanden, die derart wirkt, dass es statt komplett zu versagen mit reduzierter Leistungsfähigkeit weiter für den Betrieb zur Verfügung steht (nach ATIS Telecom Glossary 2000 T1.523-2001).

3.33**Signaltrajektorie**

(en: signal trajectory)

Zeitabläufe aller Gerätezustände, interner Zustände, Eingangssignale und Bedienereingaben, die den Ausgangszustand eines Systems bestimmen

3.34**Software**

(en: software)

Programme (d. h. Sätze geordneter Instruktionen), Daten, Regeln und zugehörige Dokumentation betreffend den Betrieb eines rechnerbasierten leittechnischen Systems

[IEC 62138, 3.27]

3.35**Software-Entwicklung**

(en: software development)

Phase des Software-Lebenszyklus, die zur Erzeugung der Software für ein leittechnisches System oder eines anderen Software-Produkts führt. Sie umfasst alle Tätigkeiten von der Anforderungsspezifikation bis zur Installation in der Anlage.

[IEC 62138, 3.30]

3.36**Software-Modifizierung**

(en: software modification)

Änderung eines bereits abgestimmten Dokuments, die zu einer Veränderung des ablauffähigen Codes führt

ANMERKUNG Software-Modifizierungen können während der Entwicklung von Software auftreten (z. B. bei der Behebung von Fehlern, die in einem späteren Stadium der Entwicklung gefunden wurden) oder nachdem die Software bereits in Betrieb genommen wurde.

3.37**Sicherheitslebenszyklus der Software**

(en: software safety life cycle)

notwendige Tätigkeiten im Zusammenhang mit Entwicklung und Betrieb von Software eines sicherheitstechnisch wichtigen leittechnischen Systems in einer Zeitspanne beginnend mit der Anforderungsspezifikation der Software in der Konzeptphase bis zu dem Zeitpunkt, zu dem die Software nicht mehr für die Nutzung zur Verfügung steht

[IEC 62138, 3.31]

3.38**Software-Version**

(en: software version)

jeweilige Ausgabe eines Software-Produkts, die durch Änderung oder Korrektur eines vorhergehenden Software-Produkts erstellt wurde

[IEEE 610, modifiziert]

3.39**Spezifikation**

(en: specification)

Dokument, in dem die Anforderungen, Funktionsweisen oder andere Eigenschaften eines Systems oder einer Komponente, oft auch die Verfahren zur Überprüfung der Einhaltung dieser Angaben, in kompletter, präziser und verifizierbarer Weise angegeben sind

[IEEE 610]

ANMERKUNG Es gibt verschiedene Arten von Spezifikationen, z. B. Anforderungs- oder Systemspezifikationen.

3.40

statische Analyse

(en: static analysis)

Prozess der Bewertung eines Systems oder einer Komponente aufgrund von Form, Struktur, Inhalt oder Dokumentation. Gegenstück zur dynamischen Analyse

3.41

Systemsoftware

(en: system software)

Teil der Software eines leittechnischen Systems, der für ein bestimmtes Rechnersystem oder eine Rechnerfamilie erstellt wurde, um Entwicklung, Betrieb und Modifizierung dieser Einrichtungen und der zugehörigen Programme zu erleichtern

[IEC 62138, 3.33]

3.42

Systemvalidierung

(en: system validation)

Bestätigung durch Prüfung und Beibringen anderer Nachweise, dass ein System zur Gänze die Anforderungsspezifikation erfüllt (Funktionalität, Ansprechzeit, Fehlertoleranz, Robustheit)

3.43

Verifizierung

(en: verification)

durch Prüfen und Vorlegen objektiver Beweise bestätigen, dass die Ergebnisse einer Tätigkeit den Zielen und Anforderungen entsprechen, die für diese Tätigkeit definiert wurden

[IEC 62138, 3.35]

4 Formelzeichen und Abkürzungen

Englisch	Deutsch
CASE Computer Aided Software Engineering	– rechnerunterstütztes Software-Engineering
CCF Common Cause Failure (siehe 3.8)	CCF Versagen gemeinsamer Ursache (siehe 3.8)
I&C Instrumentation and Control	LT Leittechnik
PDS Pre-Developed Software (siehe 3.28)	PDS vorgefertigte Software (siehe 3.28)
PIE Postulated Initiating Event (siehe 3.27)	AVE anzunehmendes versagensauslösendes Ereignis (siehe 3.27)
QA Quality Assurance	QS Qualitätssicherung
V&V Verification and Validation	V&V Verifizierung und Validierung

5 Allgemeine Anforderungen für Software-Projekte

5.1 Einleitung

Die Vorgänge zur Herstellung von leittechnischen Systemen für Kernkraftwerke sind in [IEC 61513](#) dargelegt, einer Norm, die das Konzept des System-Sicherheitslebenszyklus als Mittel vorstellt, mit dem der Entwicklungsvorgang kontrolliert werden kann und dessen Übernahme auch in der Evidenz resultieren sollte, den betrieblichen Einsatz von Sicherheitssystemen zu rechtfertigen. Der in [IEC 61513](#) beschriebene Sicherheitslebenszyklus enthält Anforderungen an die Phasen des Systemprojekts, schreibt aber nicht vor, welche Maßnahmen zu verwenden sind (siehe Bild 1).

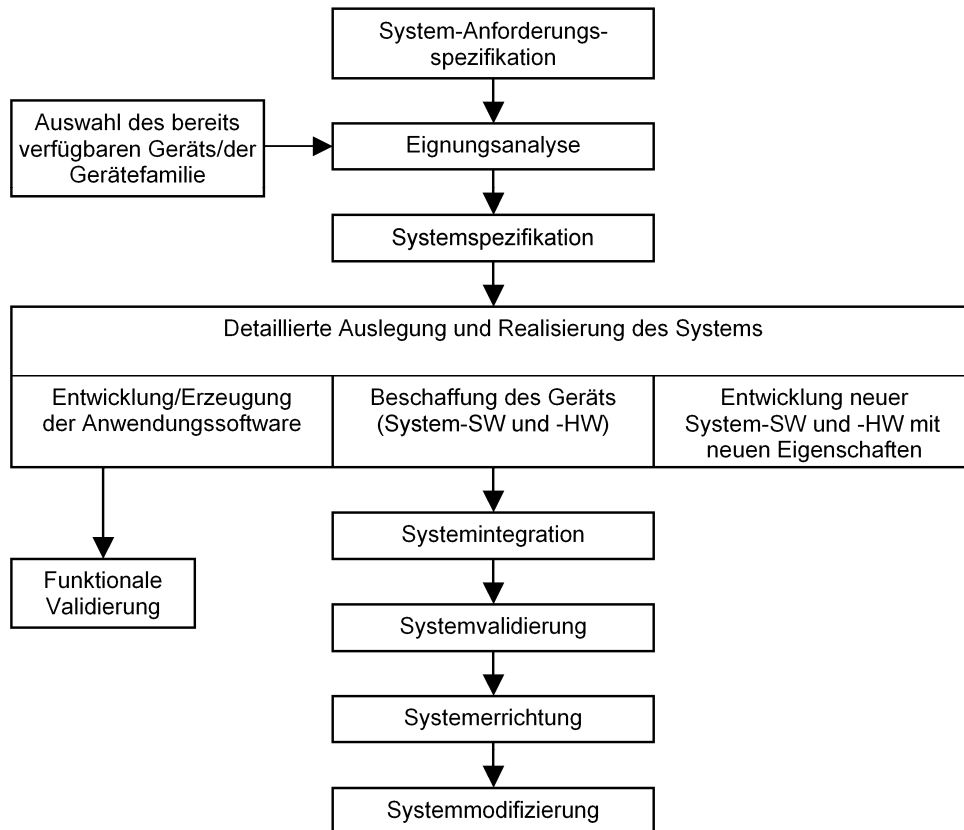


Bild 1 – Tätigkeiten im System-Sicherheitslebenszyklus (nach IEC 61513)

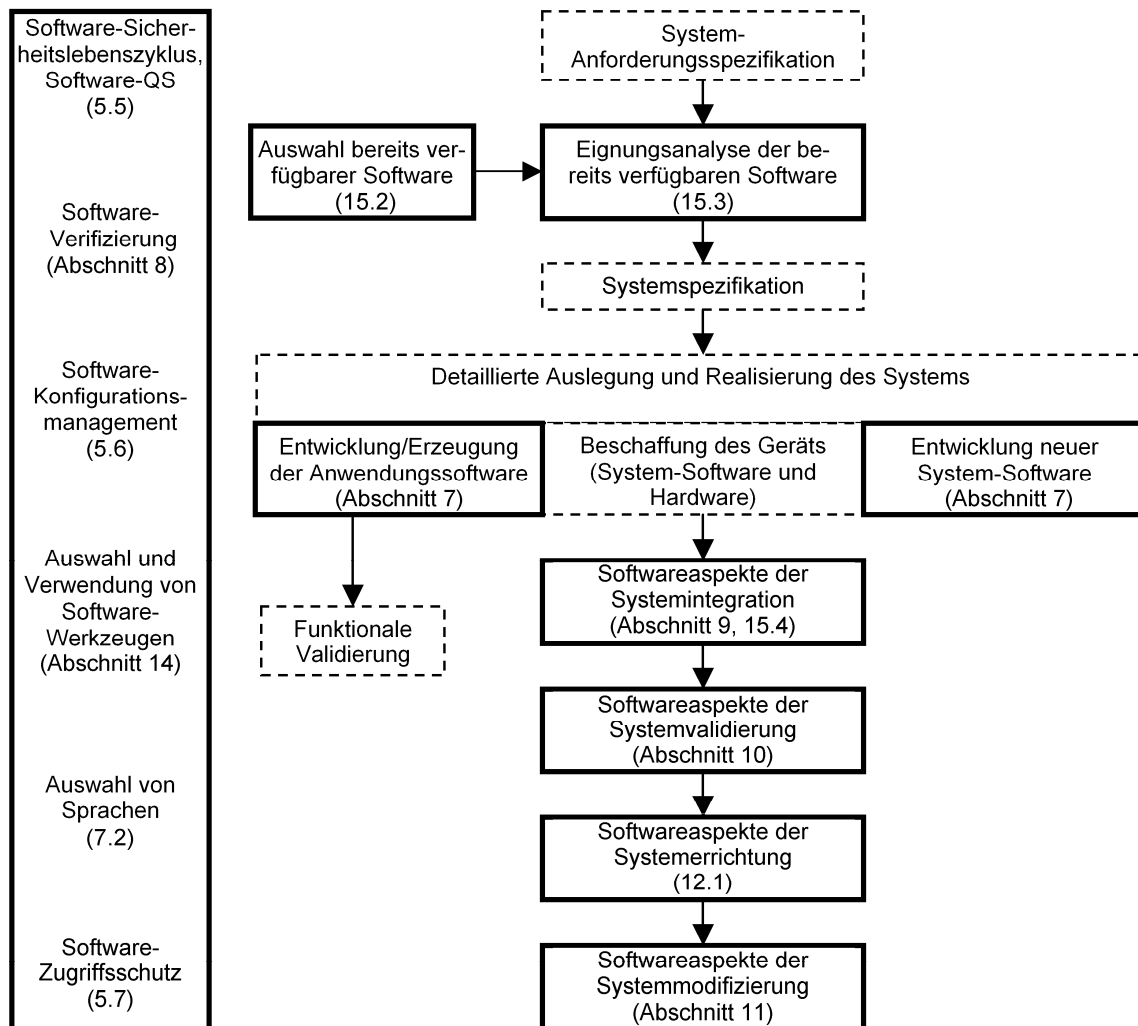
Für rechnerbasierte (d. h. digitale) Systeme wird mit der Einführung des Software-Sicherheitslebenszyklus das Konzept des System-Sicherheitslebenszyklus weiterentwickelt (siehe [Bild 2](#) bezüglich der Tätigkeiten). Aus beiden Bildern wird klar, dass die Hardware- und Software-Entwicklung ausgehend von einer gemeinsamen Spezifikation parallel vorgenommen wird, aber in den Integrations- und Installationsphasen des Lebenszyklus wieder zusammenläuft.

Durch folgende Vorgänge wird der phasenweise Entwicklungsprozess zur Erstellung der Software unterstützt:

- Software-Projektmanagement ([5.4](#));
- Software-Qualitätssicherung und -Qualitätskontrolle ([5.5](#));
- Software-Konfigurationsmanagement ([5.6](#));
- Software-Zugriffsschutz ([5.7](#));
- Software-Verifizierung ([Abschnitt 8](#)).

Weiter gibt es Tätigkeiten im Zusammenhang mit Auswahl der Rechnersprachen (7.2, Anhang D), Auswahl von Software-Werkzeugen zur Unterstützung der Entwicklung (Abschnitt 14), Vorkehrungen gegen CCF (Abschnitt 13) und Erstellung der Dokumentation (7.4, Anhang F).

Die sich für den Software-Sicherheitslebenszyklus ergebenden Tätigkeiten und die unterstützenden Vorgänge sind unten in Bild 2 gezeigt (Kästen mit vollen Linien enthalten in Klammern Hinweise auf die zugehörigen Abschnitte).



ANMERKUNG Gestrichelte Kästen stellen Tätigkeiten am System dar, die in dieser Norm nicht angesprochen werden.

Bild 2 – Softwarebezogene Tätigkeiten im System-Sicherheitslebenszyklus

Die Vorgehensweise bei der Software-Entwicklung sollte auf dem traditionellen V-Modell beruhen, da diese in anderen Normen, namentlich IAEA NS-G-1.3, aufgegriffen und verbreitet wurde, außerdem notwendige Anpassungen erlaubt, in Berücksichtigung der Tatsache, dass einige Phasen der Entwicklung mit Hilfe von Werkzeugen automatisch ablaufen können und dass die Software-Entwicklung iterativ sein kann.

In 5.2 und 5.3 werden die verschiedenen Softwaretypen und der in dieser Norm betrachtete Entwicklungsprozess vorgestellt.

5.2 Softwaretypen

Die Softwarekomponenten eines Systems werden häufig als entweder der Betriebssoftware (Kommunikation, Eingangs-/Ausgangs-Management, Standardfunktionen, Selbstüberwachung usw.) oder Anwendungssoftware (Verriegelungslogik, Steuerungskreise, Darstellungsformate, Alarmlogik usw.) zugehörig unterteilt.

Anwendungssoftware nutzt im Allgemeinen die von der Betriebssoftware bereitgestellten Funktionen und vermeidet damit die Notwendigkeit, Code innerhalb von Modulen zu duplizieren, wodurch der Gesamtumfang der Software reduziert wird.

Anwendungssoftware ist üblicherweise projektspezifisch.

Betriebssoftware kann in verschiedenen Projekten verwendet werden.

In vielen Systemauslegungen wird extensiv von Konfigurationsdaten Gebrauch gemacht. Konfigurationsdaten können Betriebssoftware oder Anwendungssoftware zugeordnet werden. Konfigurationsdaten für Anwendungssoftware bestehen hauptsächlich aus technischen Anlagendaten, die aus der Auslegung der Anlage resultieren und somit häufig von den damit befassten Ingenieuren zur Verfügung gestellt werden, von denen Software-Erfahrung nicht gefordert wird.

Konfigurationsdaten können unterteilt werden in

- Daten, deren Online-Änderung durch die Betriebsmannschaft nicht vorgesehen ist und die denselben Anforderungen unterliegen wie die sonstige Software;
- Parameter, d. h. Daten, die durch die Betriebsmannschaft während des Anlagenbetriebs geändert werden können (z. B. Alarmwerte, Grenzwerte, zur Kalibrierung der Instrumentierung benötigte Daten) und die spezifischen Anforderungen unterliegen.

Viele moderne leittechnische Plattformen sind mit extensiven Entwicklungswerkzeugen versehen, die es den Systemingenieuren erlauben, ablauffähigen Code zu erzeugen.

Beispielweise schließt ein typisches leittechnisches System, das in Hinblick auf die Verwendung von Komponenten einer Gerätefamilie entwickelt wurde, Folgendes ein:

- vorgefertigte Softwarekomponenten, wie etwa den Kern der Betriebssoftware und die Bibliothek der Anwendungsfunktionen;
- Konfigurationsdaten, die zur Anpassung des Kerns der Betriebssoftware an die Eingangs-/Ausgangs-Umgebung und an die für die Anwendung erforderlichen Dienste benötigt werden;
- Anwendungssoftware, die unter Verwendung einer anwendungsorientierten Sprache entwickelt wurde.

5.3 Vorgehensweise zur Software-Entwicklung

Software trägt üblicherweise wesentlich zur Erzeugung der Funktionen bei, die von einem leittechnischen System ausgeführt werden. Sie kann auch zusätzliche Funktionen unterstützen, die über die Systemauslegung eingeführt wurden (z. B. Initialisierung und Überwachung von Hardware, Kommunikation zwischen und Synchronisierung von Teilsystemen). Daher ist der Software-Sicherheitslebenszyklus in den meisten Fällen voll in den System-Sicherheitslebenszyklus integriert. Insbesondere ist die Software-Anforderungsspezifikation ein Teil der Systemspezifikation und Systemauslegung oder direkt davon abgeleitet.

Obwohl die Verifizierung neuer Software definitiv ein Teil des Software-Sicherheitslebenszyklus ist, gibt es oft keine separate und wohldefinierte Grenze zwischen Software-Integration und Systemintegration. In dieser Norm wird die Software-Integration als Teil der Systemintegration betrachtet. Auch Software-Validierung ist keine reine Softwaretätigkeit. In dieser Norm wird sie als Teil der Systemintegration und/oder Systemvalidierung angesehen.

In dieser Norm wird davon ausgegangen, dass der Software-Lebenszyklus, ursprünglich für die Entwicklung von Software mit allgemeinen Sprachen vorgesehen, auch für anwendungsorientierte Sprachen und die Konfiguration vorgefertigter Software verwendbar ist.

DIN IEC 60880 (VDE 0491-3-2):2007-08

Jedoch werden Unterschiede im Entwicklungsvorgang berücksichtigt, indem bei der Realisierung jedem Softwaretyp zugeordnete Teilvorgänge eingeführt werden:

- Realisierung unter Verwendung allgemeiner Sprachen;
- Realisierung unter Verwendung anwendungsorientierter Sprachen mit zugehörigen Codegeneratoren;
- Auswahl, Verwendung und Konfiguration von vorgefertigter Software.

Da die Kästen „Entwicklung von Anwendungssoftware/Erzeugung“ und „Entwicklung neuer Betriebs-Software“ in [Bild 2](#) einen großen und wesentlichen Teil des Software-Sicherheitslebenszyklus ausmachen, ist in [Bild 3](#) eine vergrößerte Darstellung gewählt, die eine detailliertere Beschreibung der Tätigkeiten zwischen Software-Anforderungsspezifikation und Validierung ermöglicht. Dabei wird auch eine klare Unterscheidung der drei Realisierungspfade getroffen. In diesem Bild sind Hinweise auf die zugehörigen Abschnitte dieser Norm in Klammern angegeben.

In [Anhang B](#) dieser Norm finden sich auch zusätzliche Anforderungen für Software.

Die Prinzipien, die sich in den Anforderungen dieser Norm widerspiegeln, sind auf die Qualität des endgültigen Codes ausgerichtet und sind unabhängig davon anwendbar, ob der Code unter der Verwendung allgemeiner Sprachen, anwendungsorientierter Sprachen mit automatischer Code-Erzeugung oder Konfiguration entwickelt wird.

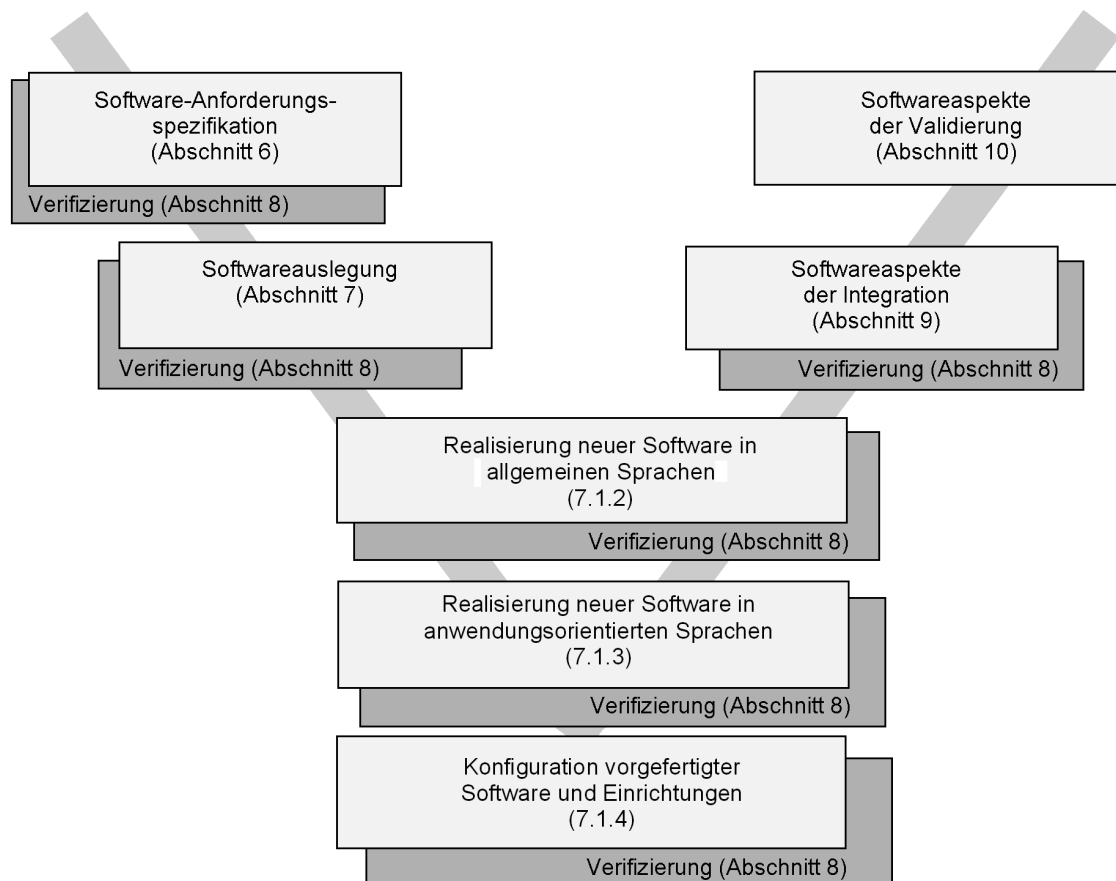


Bild 3 – Entwicklungstätigkeiten im Software-Sicherheitslebenszyklus nach dieser Norm

5.4 Software-Projektmanagement

5.4.1 Jedes Software-Projekt ist in mehreren Phasen zu strukturieren.

Jede Phase ist in gewissem Maß in sich geschlossen, hängt aber von anderen Phasen ab und beeinflusst andere Phasen. Diese Phasen sind auch ohne Formalismus durch die speziell mit ihnen zusammenhängenden Tätigkeiten erkennbar.

Die Phasen und zugehörigen Tätigkeiten, die für ein Software-Projekt definiert werden, beschreiben einen Vorgang, der in dieser Norm „Software-Entwicklung“ genannt wird. Dieser Vorgang kann iterativ sein, vorausgesetzt, die Anforderung im letzten Absatz der Einleitung zu [Abschnitt 6 der IEC 61513](#) ist erfüllt.

Folgende allgemeine Faktoren sind bei der Realisierung eines Software-Projekts für die Tätigkeiten und Phasen bestimmend:

5.4.2 Die Tätigkeiten, die in den Phasen des Entwicklungsvorgangs auszuführen sind, müssen entsprechend der für das Projekt gewählten Vorgehensweise zur Software-Entwicklung festgelegt werden (siehe [5.2](#) und [5.3](#)).

5.4.3 Die Tätigkeiten der Software-Entwicklung müssen den gesamten Software-Sicherheitslebenszyklus einbeziehen.

5.4.4 Jede in 5.4.1 festgelegte Phase der Software-Entwicklung muss in gut definierte Tätigkeiten unterteilt werden.

5.4.5 Die Phasen der Software-Entwicklung müssen formalisiert sein und keine der definierten Phasen darf ausgelassen werden.

5.4.6 Sollten Tätigkeiten zur Software-Entwicklung durch Verwendung von Werkzeugen automatisiert sein, müssen die automatisierten Vorgänge dokumentiert sein, einschließlich der für die Phasen maßgebenden Eingänge und Ausgänge.

5.4.7 Die Ein- und Ausgänge jeder Phase müssen definiert und dokumentiert sein.

5.4.8 Das Ergebnis jeder Phase muss systematisch überprüft werden (siehe [B1.c](#) und [B4.g](#)).

5.4.9 Jede Phase muss eine hinreichende Dokumentation mit einschließen ([Anhang F](#)).

5.4.10 Jede Phase muss systematisch durch eine Überprüfung abgeschlossen werden, einschließlich Überprüfung relevanter Dokumente.

5.4.11 Eine Liste der Dokumente, die im Zuge des Software-Sicherheitslebenszyklus verwendet wurden, muss während der Software-Entwicklung angefertigt werden. Ein typisches Beispiel für eine derartige Liste findet sich in [Anhang F](#).

5.5 Software-Qualitätssicherungsplan

5.5.1 Ein Qualitätssicherungsplan muss vorhanden sein oder in einem frühen Stadium des Software-Sicherheitslebenszyklus aufgestellt werden.

Es dürfen spezielle Qualitätssicherungspläne für individuelle Produktphasen oder einzelne Softwarekomponenten entsprechend nationaler Normen oder Firmenstandards übernommen werden, vorausgesetzt, die in dieser Norm enthaltenen Prinzipien werden angesprochen.

5.5.2 Jede Abweichung von den Anforderungen dieser Norm und ihrer normativen Anhänge muss gekennzeichnet und gerechtfertigt werden.

5.5.3 Wenn Praktiken angewendet werden, die von den in den normativen Anhängen genannten abweichen, müssen sie dokumentiert werden und nach den Anforderungen in den Hauptteilen dieser Norm überprüfbar sein.

5.3.4 Insbesondere muss der Einfluss dieser Praktiken auf das leittechnische System und die Software in Betracht gezogen werden.

5.5.5 Alle technischen Prozeduren, die in den verschiedenen Phasen des Software-Sicherheitslebenszyklus benötigt wurden, müssen im Qualitätssicherungsplan angesprochen werden.

5.5.6 Im Qualitätssicherungsplan muss gefordert werden, dass die Aktivitäten im Rahmen der Phasen kompetenten Personen zugeordnet werden, die über geeignete Ressourcen verfügen.

5.5.7 Im Qualitätssicherungsplan muss verlangt werden, dass Modifizierungen in bereits genehmigten Dokumenten zusammengestellt, geprüft und von autorisierten Personen genehmigt werden.

5.5.8 Im Qualitätssicherungsplan muss gefordert werden, dass die verwendeten Methoden, Sprachen, Werkzeuge, Regeln und Normen festgelegt und dokumentiert werden, dem betroffenen Personenkreis bekannt sind und von ihm beherrscht werden.

5.5.9 Im Qualitätssicherungsplan muss verlangt werden, dass bei Verwendung mehrerer Methoden, Sprachen, Werkzeuge, Regeln und/oder Normen eindeutig klar ist, welche für die jeweiligen Aktivitäten zum Einsatz kam.

5.5.10 Im Qualitätssicherungsplan muss verlangt werden, dass projektspezifische Ausdrücke, Abkürzungen und Konventionen explizit definiert sind.

5.5.11 Im Qualitätssicherungsplan muss verlangt werden, dass alle Qualitätskomplexe angesprochen, verfolgt und gelöst werden.

5.5.12 Im Qualitätssicherungsplan muss verlangt werden, dass von jeder qualitätssichernden Maßnahme Aufzeichnungen gemacht werden.

5.5.13 Jeder Verifizierungsschritt oder jede Überprüfung muss in einem Bericht über die durchgeführte Analyse, die erzielten Schlussfolgerungen und die vereinbarten Lösungen resultieren.

5.5.14 Jede Abweichung vom Qualitätssicherungsplan muss dokumentiert und gerechtfertigt werden.

5.6 Konfigurationsmanagement

In **6.2.1.2** (System-Konfigurationsmanagementplan) der **IEC 61513** sind Anforderungen für das Konfigurationsmanagement auf Ebene des leittechnischen Systems zu finden. Die in diesem Abschnitt folgenden Ausführungen enthalten zusätzliche Anforderungen, die von spezifischer Bedeutung für Software sind.

5.6.1 Das Konfigurationsmanagement für Software muss nach den Vorgaben des Konfigurationsmanagementplans oder des Qualitätssicherungsplans erfolgen.

5.6.2 Diese Vorgaben müssen konsistent mit jenen sein, die für das Konfigurationsmanagement auf Systemebene gelten.

5.6.3 Dokumentierte Vorgehensweisen zum Software-Konfigurationsmanagement müssen frühzeitig im Lebenszyklus des Software-Projekts festgelegt werden, um die nachfolgenden Anforderungen ansprechen zu können.

5.6.4 Jede erzeugte Version jeder Softwareeinheit muss eindeutig festgelegt sein.

5.6.5 Es muss möglich sein, die relevanten Versionen der gesamten Software-Dokumentation zu identifizieren, die zu den jeweiligen Softwareeinheiten gehören.

5.6.6 In Entwicklung befindliche Software muss von Software deutlich getrennt sein, die bereits freigegeben oder verifiziert wurde.

5.6.7 Es muss möglich sein, die Versionen aller Softwareeinheiten zu identifizieren, die zusammen eine komplette Version des endgültigen Produkts ausmachen.

- 5.6.8** Es muss möglich sein, die Integrität der Softwareeinheiten zu verifizieren.
- 5.6.9** Es muss möglich sein, die Software-Version des Zielsystems festzustellen.
- 5.6.10** Es muss möglich sein, im Nachhinein alle Softwareeinheiten festzustellen, die von der Realisierung einer Modifizierung betroffen sind.
- 5.6.11** Der Zugang zu allen Softwareeinheiten, die der Konfigurationskontrolle unterliegen, muss in geeigneter Weise kontrolliert werden, um sicherzustellen, dass die Software von nicht autorisierten Personen nicht modifiziert wird und der Zugriffsschutz der Software aufrechterhalten bleibt.
- 5.6.12** Es muss möglich sein, alle Übersetzungswerkzeuge und Werkzeugversionen festzustellen, die für die Erzeugung der jeweiligen ablauffähigen Softwareeinheit verwendet wurden (siehe 14.3.3).

5.7 Zugriffsschutz

Zweck des Zugriffsschutzes ist der Schutz von Software und Daten, sodass nicht autorisierte Personen und Systeme sie nicht lesen oder ändern können, aber autorisierten Personen und Systemen der Zugriff nicht verwehrt wird.

IEC 61513 enthält in 5.4.2 (Gesamt-Sicherungsplan gegen Zugriff) und 6.2.2 (System-Sicherungsplan gegen Zugriff) Anforderungen für den Zugriffsschutz auf der Ebene der leittechnischen Architektur und des einzelnen leittechnischen Systems.

Obwohl die Verwendung von Software ein gewisses Bedrohungspotenzial für den Zugriffsschutz darstellt, so ist zu berücksichtigen, dass die hauptsächlichen Gegenmaßnahmen üblicherweise auf Systemebene getroffen werden, z. B. physikalische Schutzmaßnahmen, festverdrahtete Verriegelungseinrichtungen. Die Anforderungen des Zugriffsschutzes für Software können helfen, die Verwundbarkeit zu minimieren und die Schutzmaßnahmen auf Systemebene zu unterstützen.

Dieser Abschnitt enthält Anforderungen des Zugriffsschutzes, die von spezifischer Bedeutung für Software sind^{N6)}.

5.7.1 Zugriffsschutz-Analyse

5.7.1.1 Es muss eine Analyse der möglichen Bedrohung des Zugriffsschutzes bezüglich der Software durchgeführt werden. Diese Analyse muss die relevanten Phasen des System- und Software-Sicherheitslebenszyklus berücksichtigen. Sie muss die Anforderungen hinsichtlich Schutz von und Zugriffsmöglichkeit auf Daten und Software basierend auf den Vorgaben in diesem Abschnitt ermitteln.

5.7.1.2 Die Analyse zum Zugriffsschutz für Software muss im Software- oder System-Qualitätssicherungsplan oder im Software- oder Systemplan für den Zugriffsschutz berücksichtigt werden.

5.7.1.3 Wenn die Analyse zeigt, dass die Gegenmaßnahmen auf Systemebene nicht ausreichend sind, müssen mit ihrer Hilfe Anforderungen für Gegenmaßnahmen durch Software-Auslegung festgelegt werden.

5.7.2 Zugriffsschutz-Auslegung

5.7.2.1 Die Anforderungen für Auslegungs-Gegenmaßnahmen in der Software, die in der Zugriffsschutz-Analyse festgelegt wurden, müssen in die Anforderungen der Systemauslegung integriert werden.

5.7.2.2 Jede neue Software sollte so ausgelegt sein, dass die Verwundbarkeit des Systems minimiert wird.

5.7.2.3 Jede vorgefertigt Software sollte so konfiguriert und parametrisiert sein, dass die Verwundbarkeit des Systems minimiert wird, z. B. durch Minimierung der Funktionen im notwendigen Ausmaß oder durch Verwendung von bestehenden Funktionen des Zugriffsschutzes der Software.

^{N6)} Nationale Fußnote: Zum Zugriffsschutz sind zusätzlich nationale Festlegungen zu beachten.

5.7.2.4 Es muss verhindert werden, dass der Operateur gespeicherte Programme abändert.

5.7.2.5 Wenn Zugriff des Operators notwendig ist, um zur Bedienung leittechnischer Funktionen Daten zu ändern, dann muss der Zugriff durch Einrichtungen der Mensch-Maschine-Schnittstelle auf das notwendige Maß beschränkt werden.

5.7.2.6 Wo es notwendig ist, mögliche Bedrohungen des Zugriffsschutzes abzuwenden, müssen wirksame Schutzmaßnahmen in die Auslegung, Konfiguration und/oder die Parameter-Zuordnung der Software mit eingeschlossen werden, und zwar betreffend

- benutzerselektive Zugriffskontrolle für die Softwarefunktionen,
- Datenverbindungen zu Systemen mit geringerer Bedeutung für die Sicherheit,
- Verfolgbarkeit von Software- und Parameter-Änderungen.

5.7.2.7 In der Auslegungsdokumentation müssen die für den Zugriffsschutz kritischen Funktionen und die Merkmale des Zugriffsschutzes, die mit Hilfe der Software realisiert wurden, dargelegt und beschrieben werden.

5.7.2.8 Während der Verifizierung der Software muss die Wirksamkeit der implementierten Funktionen des Zugriffsschutzes bestätigt werden.

5.7.2.9 Während der Validierung des leittechnischen Systems muss die Wirksamkeit der in die Software implementierten Funktionen des Zugriffsschutzes durch geeignete Tests nachgewiesen werden.

5.7.3 Benutzerzugang

5.7.3.1 Wo notwendig, muss die Software technische Maßnahmen für einen effektiven Authentifizierungsvorgang unterstützen, bevor der Zugriff des Benutzers erlaubt wird.

5.7.3.2 Wo Benutzerzugang ein kritisches Merkmal für den in die Software implementierten Zugriffsschutz darstellt, sollte der in der Software implementierte Authentifizierungsvorgang technische Mittel für eine Kombination von Wissen (z. B. Kennwort), Besitz (Schlüssel, Smart Card), persönliche Merkmale (z. B. Fingerabdruck) bereitstellen, und nicht nur für ein Kennwort.

5.7.3.3 Die Software und die rollenbezogene Zugangskontrolle müssen so konfiguriert und parametrisiert sein, dass der erlaubte Benutzerzugang zu den Funktionen auf das notwendige Ausmaß minimiert wird.

5.7.3.4 Der Benutzerzugang muss in geeignetem Ausmaß in Hinblick auf Gelegenheiten und Folgen einer möglichen Bedrohung des Zugriffsschutzes geschützt werden.

Die Anwendung von kryptographischen Methoden kann eine der Möglichkeiten sein, die oben genannte Anforderung zu realisieren.

5.7.3.5 Ein Zugang von außerhalb des technischen Umfelds der Anlage (z. B. von den Verwaltungsgebäuden oder von außerhalb der Anlage), durch den Softwarefunktionen oder Daten beeinflusst werden können, darf nicht realisiert werden.

5.7.4 Zugriffsschutz während der Entwicklung

5.7.4.1 Im Software-Sicherheitslebenszyklus der Entwicklung sollten mögliche Bedrohungen des Zugriffsschutzes während der Entwicklungs- und Instandhaltungstätigkeiten angesprochen werden.

5.7.4.2 Es muss Vorkehrungen gegen versteckte Funktionen in der Anwendungssoftware oder Systemsoftware geben (z. B. Code-Verifizierung), weil sie möglichen nicht autorisierten Zugriff unterstützen könnten.

5.7.4.3 Wenn keine Vorkehrungen für vorgefertigte Software realisiert werden können, muss die Verwendung derartiger Software gerechtfertigt werden, wobei die mögliche Bedrohung des Zugriffsschutzes, die sicherheitstechnische Bedeutung der betroffenen leittechnischen Funktionen und die Eigenschaften des Systems und der Software in Betracht zu ziehen sind.

5.7.4.4 Mögliche Mittel zur vorsätzlichen Änderung von Software, die durch gewisse zeitliche oder Daten-Randbedingungen getriggert fehlerhaftes Verhalten der Software verursachen können, müssen identifiziert werden und es muss begründet werden, dass sie bei den Verifizierungstätigkeiten auffindbar sind.

6 Software-Anforderungen

6.1 Spezifikation der Software-Anforderungen

6.1.1 Software-Anforderungen müssen von den Anforderungen der Sicherheitssysteme abgeleitet sein, sie sind Teil der Systemspezifikation des Rechners.

6.1.2 Die Software-Anforderungen müssen beschreiben, was die Software zu tun hat, und nicht, wie sie es tun muss.

6.1.3 Die Software-Anforderungen müssen Folgendes spezifizieren:

- die von der Software bereitzustellenden Anwendungsfunktionen;
- die verschiedenen Verhaltensarten der Software und die entsprechenden Übergangsbedingungen;
- die Schnittstellen und Wechselwirkungen der Software mit ihrem Umfeld (z. B. mit den Operateuren, dem übrigen leittechnischen System, anderen Systemen, mit denen sie in Wechselwirkung tritt oder Ressourcen teilt), einschließlich der Aufgaben, Typen, Bereiche und Randbedingungen der Eingänge und Ausgänge;
- die Parameter der Software, die während des Betriebs von Hand geändert werden können (wenn vorhanden), ihre Aufgaben, Typen, Formate, Bereiche und Randbedingungen, und die Überprüfungen, die durchzuführen sind, wenn die Software geändert wird;
- die geforderte Leistungsfähigkeit der Software, insbesondere Anforderungen hinsichtlich der Ansprechzeiten;
- was die Software nicht tun darf oder verhindern muss, wenn zutreffend;
- die Anforderungen der Software oder die durch Software getroffenen Annahmen, in Hinblick auf ihr Umfeld, wenn zutreffend;
- die Anforderungen der Standard-Softwarepakete, wenn zutreffend.

6.1.4 Wegen der Bedeutung dieser Phase der Software-Entwicklung müssen die Vorgänge zur Erstellung der Software-Anforderungen sehr streng gehandhabt werden.

6.1.5 Die Software-Anforderungsspezifikation muss so abgefasst sein, dass die Übereinstimmung des leittechnischen Systems mit den Anforderungen von **IEC 61513** gezeigt werden kann.

Zusätzliche Anforderungen für die Software-Anforderungsspezifikation sind in [Anhang A](#) zu finden.

6.1.6 Die Randbedingungen zwischen Hardware und Software müssen beschrieben sein ([A.2.1](#)).

6.1.7 Ein Hinweis auf die Hardware-Anforderungsspezifikation muss in der Software-Anforderungsspezifikation für jegliche Rückwirkung auf die Hardware gemacht werden.

6.1.8 Spezielle Betriebsbedingungen wie Inbetriebnahme der Anlage oder Brennstoffwechsel müssen für die betroffenen Funktionen bis auf die Softwareebene hinunter beschrieben werden.

6.2 Selbstüberwachung

6.2.1 Die Software des rechnerbasierten Systems muss während des Betriebs die Hardware in spezifizierten Zeitabständen und das Verhalten der Software überwachen ([A.2.2](#)).

Das wird als primärer Faktor für die Erzielung hoher Zuverlässigkeit des Gesamtsystems betrachtet.

6.2.2 Jene Teile des Speichers, die Code oder unveränderliche Daten enthalten, müssen in Hinblick auf unbeabsichtigte Änderungen überwacht werden.

6.2.3 Die Selbstüberwachung sollte in der Lage sein, in praktikablem Ausmaß

- Zufallsausfälle von Hardwarekomponenten,
 - fehlerhaftes Verhalten der Software (z. B. Abweichungen von spezifizierten Software-Abläufen und Betriebsbedingungen oder Datenbeschädigung),
 - fehlerhafte Datenübertragung zwischen verschiedenen Prozessoren
- zu entdecken.

6.2.4 Wenn während des Anlagenbetriebs durch die Software irgendein Fehler entdeckt wird, muss die Software in geeigneter Weise zeitgerecht reagieren. Die Reaktionsmaßnahmen müssen nach den in der Spezifikation und in den Auslegungsregeln von [IEC 61513](#) geforderten Systemreaktionen realisiert werden.

Dies kann geeignete Überlegungen zur Vermeidung fälschlicher Reaktionen erfordern.

6.2.5 Es muss sichergestellt sein, dass Selbstüberwachung die geplanten Systemfunktionen nicht beeinträchtigt.

6.2.6 Es sollte möglich sein, alle nützlichen diagnostischen Informationen zu sammeln, die sich aus der Selbstüberwachung durch Software ergeben.

6.3 Periodische Prüfungen

6.3.1 Bei rechnerbasierten Sicherheitssystemen sollten die allgemeinen Prinzipien von [IEC 60671](#) für jene Komponenten angewendet werden, die durch Selbstüberwachung nicht ausreichend abgedeckt sind.

6.3.2 Die Software muss so ausgelegt sein, dass die Anforderungen periodischer Prüfungen, die innerhalb festgelegter maximaler Zeitabstände stattfinden, erfüllt werden (z. B. Zeitabstände regulärer Abschaltungen).

- 1) Jede Sicherheitsfunktion muss durch periodisches Prüfen abdeckbar sein.
- 2) Jedes Versagen der Sicherheitsfunktionen muss aufgedeckt werden.

6.3.3 Es sollte möglich sein, alle während der periodischen Prüfungen gesammelten nützlichen diagnostischen Informationen automatisch zu sammeln.

6.3.4 Die Qualität der Software von Hilfseinrichtungen zum Prüfen sollte der Qualität von Verifizierungswerkzeugen nach [Abschnitt 10](#) entsprechen.

Die Software für Hilfseinrichtungen zum Prüfen von Systemen der Klasse 1 muss nicht so ausgelegt werden, dass sie allen Software-Sicherheitsanforderungen der vorliegenden Norm entspricht.

6.4 Dokumentation

6.4.1 Der Hauptzweck der Software-Anforderungsspezifikation besteht darin, eine Grundlage für Software-Entwicklung zu bilden. Jedoch sollten Genehmigungsaspekte nicht vernachlässigt werden, da es sein kann, dass dieses Dokument im Genehmigungsverfahren vorgelegt wird. Deshalb kann es auch Gesichtspunkte von geringerer Relevanz für die Software-Entwicklung enthalten, die jedoch eine wichtige Hintergrundinformation für die Genehmigung darstellen.

Solche wichtigen Genehmigungsaspekte können sein:

- Risiko-Überlegungen;
- Empfehlungen für Funktionen oder sicherheitstechnische Merkmale;
- Informationen, die den Hintergrund für spezifische Anforderungen darstellen;
- spezielle genehmigungstechnische Anforderungen an die Softwarestruktur, Code-Analyse, V&V usw.

6.4.2 Die Software-Anforderungsspezifikation muss in ihrer Darstellungsweise einem Standard entsprechen, dessen Formalitäten die Lesbarkeit nicht behindern sollten (A.2.3).

6.4.3 Die Software-Anforderungsspezifikation muss eindeutig, prüfbar oder verifizierbar, und umsetzbar sein.

Eine formale oder eine anwendungsorientierte Sprache kann verwendet werden, um die Schlüssigkeit und Vollständigkeit von Aspekten der Software-Anforderungsspezifikation zu verbessern.

Für diesen Zweck können automatische Werkzeuge verwendet werden.

6.4.4 Die Software-Anforderungsspezifikation muss den relevanten Teilnehmern am Auslegungsprozess zur Verfügung gestellt werden.

7 Auslegung und Implementierung

Der folgende Abschnitt basiert auf erprobter Praxis zur Software-Entwicklung mit entsprechenden Sicherheitsmerkmalen, nach Möglichkeit mit Hilfe eines Fehlerbaums, und einer Verifizierung gut zugänglich.

Die Software-Anforderungsspezifikation muss verfügbar sein, bevor die Auslegungs- und Implementierungsphase der Programmentwicklung beginnt.

In der Auslegungs- und Implementierungsphase wird die Software-Anforderungsspezifikation realisiert und die Basis für die Verifizierung der Auslegung und Realisierung der Software gelegt.

7.1 Prinzipien der Auslegung und Realisierung

7.1.1 Allgemeines

7.1.1.1 Die Software-Auslegung muss eine Selbstüberwachung einschließen (A.2.2).

7.1.1.2 Bei Entdeckung eines Ausfalls müssen entsprechende Maßnahmen nach 6.2 ergriffen werden.

7.1.1.3 Die Programmstruktur sollte auf einer Zerlegung in Module basieren.

7.1.1.4 Die Programmstruktur sollte einfach und leicht verständlich sein, und zwar sowohl hinsichtlich ihrer Gesamtauslegung als auch ihrer Details.

7.1.1.5 Tricks, rekursive Strukturen und Code-Kompaktierung sollten vermieden werden.

7.1.1.6 Das Quellprogramm sollte für erfahrene Ingenieure verständlich sein, die nicht in den Entwicklungsvorgang der Software eingeschaltet waren.

7.1.1.7 Das Quellprogramm sollte dokumentierten Regeln folgen, die der Verbesserung der Klarheit, Modifizierbarkeit und Prüfbarkeit dienen.

7.1.1.8 Unstimmigkeiten mit den Auslegungsregeln sollten gerechtfertigt werden.

7.1.1.9 Es muss eine umfassende und klar formulierte Dokumentation bereitgestellt werden.

7.1.1.10 Kommunikationsverknüpfungen müssen in Übereinstimmung mit den Anforderungen für Datenkommunikation nach 5.3.1.3 von IEC 61513 ausgelegt werden.

7.1.1.11 Kommunikationsverknüpfungen innerhalb eines redundanten Stranges müssen deterministisch sein.

7.1.1.12 Von diesen Prinzipien sind folgende Empfehlungen abgeleitet:

- 1) Vorkehrungen zur Realisierung der Software-Sicherheitsanforderungen einschließlich Selbstüberwachung sollten zu Beginn der Auslegung getroffen werden (B.3);
- 2) für die Software-Auslegung sollte eine Top-Down-Vorgehensweise der Bottom-Up-Vorgehensweise vorgezogen werden (B.1);
- 3) zu Beginn jedes Software-Projekts sollte man ein konzeptionelles Modell für die Software-Architektur vorsehen (B.2);
- 4) das Programm sollte so geschrieben sein, dass eine einfache Verifizierung ermöglicht wird (B.4, B.5);
- 5) wenn Standardsoftware eines Herstellers oder einer Lieferfirma verwendet wird, gelten die Anforderungen von [Abschnitt 15](#) zusätzlich zu B2.c;
- 6) anwendungsorientierte Sprachen sollten gegenüber maschinenorientierten Sprachen bevorzugt verwendet werden (B5.e).

7.1.2 Realisierung neuer Software in allgemeinen Sprachen

Dieser Abschnitt betrachtet Fälle, in denen ein Teil oder alle Sicherheitsfunktionen der Kategorie A durch die Entwicklung neuer Softwarekomponenten unter Verwendung allgemeiner Sprachen bereitgestellt werden.

Allgemeine Sprachen sind üblicherweise entweder Hochsprachen wie Ada, C, Pascal oder der verwendeten Plattform zugeordnete Assemblersprachen. Sie können verwendet werden, um jede Art von Funktion zu realisieren, vorausgesetzt, es werden geeignete Auslegungs- und Codierungsregeln befolgt.

7.1.2.1 In der Software-Auslegungsphase müssen die Softwarekomponenten festgelegt werden, die mit allgemeinen Sprachen entwickelt werden sollen.

7.1.2.2 Für diese Komponenten sollte im Entwicklungsvorgang eine detaillierte Auslegungsphase und eine Codierungsphase festgelegt werden.

7.1.2.3 Durch Detailarbeit in der Auslegungsphase müssen die Ergebnisse dieser Phase bis zu dem Punkt gebracht werden, an dem die Codierung in systematischer Form mit der ausgewählten Sprache durchgeführt werden kann (z. B. durch Definition der notwendigen Algorithmen, Datenstrukturen, Funktions-Schnittstellen, Randbedingungen usw.).

7.1.2.4 Der Detaillierungsgrad der Information, die in der detaillierten Auslegungsphase bereitzustellen ist, hängt von der verwendeten allgemeinen Sprache ab. Wird eine Assemblersprache benutzt, so müssen in der Auslegungsphase detailliert strukturierte Algorithmen und Datendarstellungen bereitgestellt werden.

7.1.2.5 In der Codierungsphase muss die detaillierte Auslegung nach vorgegebenen Programmierungsregeln entsprechend den Anforderungen in [Anhang B](#) in den Quellcode umgesetzt werden.

7.1.3 Realisierung neuer Software in anwendungsorientierten Sprachen

Dieser Abschnitt betrachtet Fälle, in denen ein Teil oder alle Sicherheitsfunktionen durch die Entwicklung neuer Software bereitgestellt werden, indem anwendungsorientierte Sprachen verwendet werden.

Anwendungsorientierte Sprachen unterstützen Formalismen (wie logische Diagramme oder Funktionsblock-Diagramme etc.), die verwendet werden können, um ganz oder teilweise die Software-Anforderungsspezifikation und/oder teilweise die Software-Auslegungsspezifikation auszudrücken. Diese Teile können als Eingangsinformation zur automatischen Erzeugung von ablauffähigem Code benutzt werden.

7.1.3.1 Die verwendeten Formalismen sollten folgende Eigenschaften haben: geringe Komplexität, Klarheit und standardisierte Auslegung und Darstellung, Modularität, Vorhandensein zutreffender Kommentare, Vermeidung unsicherer Merkmale. Diese Eigenschaften erleichtern im Allgemeinen das Verständnis, die Verifizierung, Prüfungen und spätere Modifizierungen.

7.1.3.2 Anwendungsorientierte Sprachen sollten ein Format aufweisen, das für Ingenieure verständlich ist, die für eine Überprüfung der Softwarespezifikation verantwortlich sind, z. B. die Verfahrenstechniker und Leittechniker, die sich mit den Systemen beschäftigen, für die die leittechnischen Funktionen spezifiziert sind.

7.1.3.3 Die anwendungsorientierte Sprache sollte eine einfache Softwarestruktur unterstützen, z. B. lineare Programme.

7.1.3.4 Die anwendungsorientierte Sprache sollte es den Entwicklern erlauben, die Spezifikation der leittechnischen Systemarchitektur zu berücksichtigen, d. h. die Zuordnung von leittechnischen Funktionen zu Systemkomponenten ermöglichen und die Entwicklung fehlertoleranter Auslegungsmerkmale unterstützen.

7.1.4 Konfiguration vorgefertigter Software

Dieser Abschnitt betrachtet Fälle, in denen Sicherheitsfunktionen der Kategorie A durch Software bereitgestellt werden, die mit Hilfe von Anwendungsdaten konfiguriert wird, d. h. mit für die geplante Anwendung spezifischen Daten.

Vorgefertigte Software kann zu einer Gerätefamilie gehören oder ein unabhängiges Produkt sein, das dann in eine ausgewählte Hardwareplattform integriert wird.

7.1.4.1 Wenn vorentwickelte Software verwendet wird, müssen die Fähigkeiten der Software evaluiert und analysiert werden (siehe [15.3](#)), um sicherzustellen, dass sie für den geplanten Einsatz geeignet ist.

Die Anforderungen für die Eignungsanalyse werden in [15.3.1.2](#) angesprochen. Soll die Verwendung der Software beschränkt werden, so kann dies durch Verwendung einer die vorgefertigte Software umschließenden Software erreicht werden.

Für die Konfiguration von Software ist eine werkzeuggestützte Vorgehensweise vorzuziehen, um den Bereich menschlicher Fehler einzuschränken.

7.1.4.2 Alle für die Daten relevanten Randbedingungen müssen dokumentiert werden, z. B. erlaubte Datenformate, Bereiche, Rechenregeln.

7.1.4.3 Die Konfigurationsdaten müssen dokumentiert werden.

7.1.4.4 Für die verwendeten Datenwerte müssen ausreichende Begründungen vorgelegt werden, einschließlich Querverweisen auf auslegungsbezogene Eingangsinformationen.

7.1.4.5 Verfolgbarkeit: Es sollte möglich sein, zu erkennen, wann und von wem Konfigurationsdaten modifiziert wurden.

7.1.4.6 Wartbarkeit: Im Entwicklungsprozess muss sichergestellt werden, dass durch eine strukturierte Vorgehensweise zusammen mit entsprechender Kommentierung in der Datendokumentation und/oder unterstützenden Dokumentation die Auslegung der Konfigurationsdaten verstanden wird und über den geplanten Lebenszyklus des Systems (zu dem die Konfigurationsdaten gehören) beibehalten wird.

Für weitergehende Anleitungen betreffend die Verwendung von Werkzeugen für Anwendungsdaten siehe [14.3.5](#).

7.2 Sprachen und zugehörige Übersetzer und Werkzeuge

7.2.1 Allgemeine Anforderungen

Obwohl die Verwendung spezieller Sprachen nicht verlangt werden kann, sind folgende Ausführungen als allgemeine grundsätzliche Regeln für Sprachen zu betrachten, die für Auslegung und Realisierung von Software für Systeme der Klasse 1 benutzt werden.

7.2.1.1 Die anzuwendende Sprache muss strikten (oder gut definierten) semantischen und Syntaxregeln folgen.

DIN IEC 60880 (VDE 0491-3-2):2007-08

7.2.1.2 Die Syntax der Sprache muss klar und vollständig definiert und dokumentiert werden.

7.2.1.3 Die Anwendung der Sprache sollte, wo zutreffend, auf eine „sichere“ Untermenge beschränkt werden, z. B. auf Primitive, die für die Spezifikation der notwendigen Funktionen geeignet sind.

7.2.1.4 Es sollten Sprachen mit einem gründlich getesteten Übersetzer verwendet werden.

7.2.1.5 Wenn kein gründlich getesteter Übersetzer verwendet wird, muss durch zusätzliche Verifizierung gezeigt werden, dass die Ergebnisse der Übersetzung korrekt sind.

7.2.1.6 Werkzeuge für automatisches Prüfen sollten verfügbar sein.

Die Verwendung automatischer Werkzeuge wird empfohlen. Die dafür maßgeblichen Anforderungen sind in [Abschnitt 14](#) angegeben.

7.2.2 Allgemeine Sprachen

Allgemeine Sprachen für Systeme der Klasse 1 und ihre Übersetzer sollten nicht daran hindern, fehlerbegrenzende Konstrukte zu verwenden wie:

- Überprüfung der Variablen-Typen während der Übersetzungszeit, Überprüfung der Parameter von Subroutinen;
- Überprüfung der Feldgrenzen während der Laufzeit.

Anleitungen für die Auswahl der Sprache, des Übersetzers usw. finden sich in [Anhang D](#).

7.2.3 Anwendungsorientierte Sprachen und zugehörige automatische Code-Erzeugung

7.2.3.1 Anwendungsorientierte Sprachen sollten durch automatische Werkzeuge in eine allgemeine Sprache transformiert werden (z. B. durch einen Codegenerator), und zwar vor dem Kompilieren in eine ablauffähige Form.

7.2.3.2 Die Übereinstimmung des erzeugten Codes mit den Anforderungen an Software-Auslegung und Software-Codierung dieser Norm muss untersucht und Abweichungen müssen gerechtfertigt werden.

7.2.3.3 Die zu erzeugende Programmstruktur muss in generischer Form definiert werden, z. B. die Position der Deklarationen im Verhältnis zu den Code-Statements.

7.2.3.4 Der erzeugte Code darf nicht durch direkte Eingriffe von Hand geändert werden.

7.2.3.5 Der Code muss neu erzeugt werden, wenn die Eingangsspezifikation modifiziert wird (z. B. in Hinblick auf Ergebnisse der V&V-Tätigkeiten).

Zusätzliche Empfehlungen zur automatischen Code-Erzeugung sind in [B5.f](#) angegeben.

7.3 Detaillierte Empfehlungen

7.3.1 Allgemeines

Eine Reihe von Empfehlungen ist in [Anhang B](#) enthalten, in dem die in [7.1](#) genannten Aspekte detailliert spezifiziert werden.

Die Überschriften der individuellen Empfehlungen in [Anhang B](#) sind für die zwei hauptsächlichen Phasen der Software-Entwicklung anwendbar, wie in [Tabelle 1](#) gezeigt.

Tabelle 1 – Prozess- und Produktaspekte von Auslegung und Implementierung^{N7)}

Software-Entwicklungsphase	Prozessaspekte	Abschnitt in Anhang B	Produktaspekte	Abschnitt in Anhang B
Auslegung	– Modifizierbarkeit – Top-Down-Vorgehensweise – Verifizierung von Zwischenprodukten der Auslegung – Änderungskontrolle während der Entwicklung	B1.a	– Kontroll- und Zugriffsstrukturen	B2.a
		B1.b	– Module	B2.b
		B1.c	– Betriebssoftware	B2.c
		B1.d	– Ausführungszeit	B2.d
			– Interrupts	B2.e
			– Arithmetische Ausdrücke	B2.f
			– Plausibilitätsprüfungen	B3.a
			– Sicherer Ausgang	B3.b
			– Zweige und Schleifen	B4.a
			– Subroutinen	B4.b
			– Eingebettete Strukturen	B4.c
			– Datenstrukturen	B4.e
			– Anwendungsorientierte Sprachen	B5.e
Implementierung	– Verifizierung von Zwischenprodukten der Auslegung – Änderungskontrolle während der Entwicklung – Einzel- und Integrationsprüfungen – Programmierungsregeln	B1.c	– Module	B2.b
		B1.d	– Ausführungszeit	B2.d
		B4.g	– Interrupts	B2.e
		B5.d	– Arithmetische Ausdrücke	B2.f
			– Plausibilitätsprüfungen	B3.a
			– Sicherer Ausgang	B3.b
			– Speicherinhalt	B3.c
			– Fehlerprüfung	B3.d
			– Zweige und Schleifen	B4.a
			– Subroutinen	B4.b
			– Eingebettete Strukturen	B4.c
			– Adressierung und Felder	B4.d
			– Datenstrukturen	B4.e
			– Dynamische Änderungen	B4.f
			– Abfolgen und Anordnungen	B5.a
			– Kommentare	B5.b
			– Assembler	B5.c
			– Automatische Code-Erzeugung	B5.f

^{N7)} Nationale Fußnote: In der IEC-Publikation sind die Kleinbuchstaben der Spalte „Abschnitt in Anhang B“ kursiv geschrieben, entsprechend dem jeweils ersten Punkt des Abschnitts (z. B. B1.a). Gemeint ist aber immer der gesamte Abschnitt (also für das gewählte Beispiel: B1.a).